

WinNonlin®

User's Guide

WinNonlin Version 5.3

WinNonlin copyright ©1998-2009 Tripos, L.P. All rights reserved. This software and the accompanying documentation are owned by Tripos, L.P. Pharsight is authorized to distribute and sublicense the material contained herein with the express written permission of Tripos, L.P. The software and the accompanying documentation may be used only as authorized in the license agreement controlling such use. No part of this software or the accompanying documentation may be reproduced, transmitted, or translated, in any form or by any means, electronic, mechanical, manual, optical, or otherwise, except as expressly provided by the license agreement or with the prior written permission of Tripos, L.P.

This product may contain the following software that is provided to Pharsight Corporation under license: Actuate™ Formula One® copyright 1993-03 Actuate, Inc. All rights reserved. Sentinel RMS 8.1.1 copyright 2006 SafeNet, Inc. All rights reserved. Microsoft® XML Parser version 3.0 copyright 1998-2005 Microsoft Corporation. All rights reserved. IMSL® copyright 1970-2008 Visual Numerics, Inc. All rights reserved.

Information in the documentation is subject to change without notice and does not represent a commitment on the part of Pharsight Corporation or Tripos, L.P. The documentation contains information proprietary to Tripos, L.P. and is for use by Pharsight Corporation, and its affiliates' and designates' customers only. Use of the information contained in the documentation for any purpose other than that for which it is intended is not authorized. NONE OF PHARSIGHT CORPORATION, TRIPOS, L.P., NOR ANY OF THE CONTRIBUTORS TO THIS DOCUMENT MAKES ANY REPRESENTATION OR WARRANTY, NOR SHALL ANY WARRANTY BE IMPLIED, AS TO THE COMPLETENESS, ACCURACY, OR USEFULNESS OF THE INFORMATION CONTAINED IN THIS DOCUMENT, NOR DO THEY ASSUME ANY RESPONSIBILITY FOR LIABILITY OR DAMAGE OF ANY KIND WHICH MAY RESULT FROM THE USE OF SUCH INFORMATION

Destination Control Statement

All technical data contained in the software and documentation are subject to the export control laws of the United States of America. Disclosure to nationals of other countries may violate such laws. It is the reader's responsibility to determine the applicable regulations and to comply with them.

United States Government Rights

The software and documentation constitute “commercial computer software” and “commercial computer software documentation” as such terms are used in 48 CFR 12.212 (Sept. 1995). United States Government end users acquire the software under the following terms: (i) for acquisition by or on behalf of civilian agencies, consistent with the policy set forth in 48 CFR 12.212 (Sept. 1995); or (ii) for acquisition by or on behalf of units of the Department of Defense, consistent with the policies set forth in 48 CFR 227.7202-1 (June 1995) and 227.7202-3 (June 1995). The manufacturer is Pharsight Corporation, 5625 Dillard Drive, Suite 205, Cary, NC 27518.

Trademarks

AutoPilot, Pharsight Knowledgebase Server (PKS), Trial Simulator, PKS Reporter, Pharsight, WinNonlin, WinNonMix, Drug Model Explorer, DMX, and Phoenix are trademarks or registered trademarks of Tripos, L.P. and are licensed to Pharsight Corporation as provided above. S-PLUS is a registered trademark of Insightful Corporation. SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. Sentinel RMS is a trademark of SafeNet, Inc. SigmaPlot is a registered trademark of Systat Software, Inc. NONMEM is a registered trademark of The Regents of the University of California. Actuate is a trademark and Formula One is a registered trademark of Actuate Corporation. Microsoft, MS, the Internet Explorer logo, MS-DOS, the Office logo, Microsoft Word, Microsoft Excel, Microsoft PowerPoint, Visual C++, Visual Studio, Windows, Windows 2000, Windows XP, Windows Vista, the Windows logo, the Windows Start logo, and the XL design (the Microsoft Excel logo) are trademarks or registered trademarks of Microsoft Corporation. Pentium and Pentium III are trademarks or registered trademarks of Intel Corporation. Adobe, Acrobat, Acrobat Reader and the Adobe PDF logo are registered trademarks of Adobe Systems Incorporated. IMSL is a registered trademark of Visual Numerics, Inc.

Compaq Visual Fortran is trademark of Compaq Computer Corporation. All other brand or product names mentioned in this documentation are trademarks or registered trademarks of their respective companies or organizations.

Pharsight Corporation
1699 S. Hanley Road, St. Louis, MO 63144 USA
Telephone: +1-919-859-6868 • Fax: +1-919-859-6871
<http://www.pharsight.com> • support@pharsight.com

Contents

Chapter 1	Introduction	1
	WinNonlin	1
	WinNonlin editions	1
	<i>In-vitro in-vivo</i> correlation toolkit	2
	WinNonlin user documentation	2
	Modeling and nonlinear regression	3
	Model fitting algorithms and features of WinNonlin	5
	Summary of regression methods	8
	Pharsight Corporation	9
	Products	9
	Scientific and consulting services	9
	Training	10
	Pharsight contact information	10
	Technical support	10
	Licensing and product upgrades	10
	Customer feedback	11
Chapter 2	Data Management	13
	WinNonlin windows and files	13
	File types	14
	WinNonlin and WinNonMix file compatibility	17
	Hidden windows	18
	Workspaces	19
	Printing	21
	Page setup	21
	Page setup for workbooks	21
	Page setup for text windows	24
	Page setup for charts	25
	Print preview	25

Print	25
Printing workbooks	26
Printing text objects and models	26
Print all	27
Print all options	27
WinNonlin options	28
Directories	28
Workbooks	28
Models	29
Tables	31
Units	31
Licensing	31
Data dependencies and origins	33
Saving dependencies	35
Refreshing results	35
Unlocking derived data	36
Object origins	36
Data history worksheets and log files	37
History worksheet	37
WinNonlin Log Viewer	39

Chapter 3	Workbooks	41
	Editing data	41
	Inserting or deleting worksheets	42
	Inserting and deleting rows, columns or cells	43
	Using formulas and functions	44
	Filling adjacent cells	46
	Formatting cells	46
	Clearing cell values and formats	49
	Undo	50
	Find, find next and replace	50
	Go to	52
	Column names and units	53
	Units builder	55
	Data manipulation	56
	Freezing rows and columns	56
	Sorting	57
	Merge	57
	Data exclusion and inclusion	60
	Re-including data	62
	Transformations	62

Multiple transformation	64
Transform action	66
Multi-transformation filter	66
Multi-transformation equation manager	67
Rank transformations	68
Status code transformations	69
Rule sets	71
Creating a rule set	73
 Chapter 4 Data import and export	75
Microsoft Word export	76
Word export document options	77
Word export chart options	78
Word export workbook options	78
SAS Transport file export	78
NONMEM export	80
Data file structure	80
Exporting to NONMEM	82
Files	82
Dependent variables	83
Other data items	83
Dose information	84
Occasions	85
SigmaPlot export	85
S-PLUS export	86
ODBC import	87
Creating an import	87
Final settings	88
Connection	90
Creating a new connection	90
Loading a saved connection	92
Queries	92
Building a query	93
Schema filter	93
Query builder	94
Loading a saved query	96
Loading an existing import	96
Refreshing an imported data set	97
The custom query builder	97
Select a builder	98
Study selection	99

ODBC export	102
Creating an export	102
Export definition	103
Building an export definition	104
Field selection	105
Loading a saved export definition	106
Loading a saved export	107
The Enterprise Software Development Kit	107
Import file format	108
Usage.	108
Export file format	109
Custom query builder interface.	109
VB example.	111

Chapter 5 Charts 115

Chart Wizard	115
X-Y variables	116
Scatter plots and bar charts	117
Histograms	118
Sort and group variables	119
Error bars	120
Chart titles	122
Axis options.	122
Chart Designer.	123
Frequently used chart formatting options	124
Changing the plot type	124
Changing line style and markers.	125
Changing axis scaling	125
Creating a semi-log plot	126
Setting grid lines	127
Removing or adding walls	128
Chart templates	129
Copy and paste.	130

Chapter 6 Tables 131

Table Wizard	131
Missing and excluded data	133
WinNonlin table templates	134
Joining data sets	143
Summary statistics.	144

Significant digits and column alignments	145
Table formatting	146
Table definition files	148
Creating a table definition file	148
Loading a table definition file	149
Chapter 7 Descriptive Statistics	151
Computing summary statistics	151
Units	153
Output and computational formulae	153
Weighted summary statistics	157
Output and computational formulae for weighted calculations	158
Chapter 8 Noncompartmental Analysis	161
NCA model selection	162
Model variables	163
Urine models	163
Lambda Z or slope estimation settings	164
Lambda Z or slope range selection	164
Calculation of Lambda Z	165
Calculation of slopes for effect data	165
Setting Lambda Z or PD slope ranges	166
Partial areas	169
Therapeutic response	170
Dosing regimen	173
Time deviations in steady-state data	174
Model options	175
Output options	175
Weight	177
Title	178
NCA settings	179
Units	180
Parameter names	181
Computational rules for WinNonlin's noncompartmental analysis engine	183
Data checking and pre-treatment	183
AUC calculation and interpolation formulas	185
Partial areas	186
Therapeutic response windows	188
Sparse sampling computations	190

Chapter 9	Modeling	195
	Loading the model	195
	Model properties	198
	Data variables	198
	Model parameters	199
	Dosing regimen	202
	Dosing data variables	204
	Model options	205
	Output options	206
	Weight	207
	Title	208
	PK settings	208
	Units	210
	Running the model	211
	Stop modeling	211
	Troubleshooting modeling problems	211
	Errors	212
	Output	212
	Text (ASCII) output	213
	Workbook output	216
	Chart output	217
 Chapter 10	 User Models	 219
	ASCII user models	219
	Creating a new ASCII model	220
	User model parameters	221
	Model structure	222
	ASCII model examples	228
	WinNonlin variables	233
	Temporary variables	234
	The WinNonlin programming language	235
	Statements	235
	Comparison operators	237
	Functions	237
	Bioassay functions	238
	Compiled user models	238
	Using FORTRAN	239
	Example using Digital (now Compaq) Visual FORTRAN 6.0	239
	Using C++	240
	Example using Microsoft Visual C++ 6.0	241

Chapter 11	Weighting	243
	Weighted least squares	243
	Iterative reweighting	244
	Reading weights from the data file	244
	Scaling of weights	244
	Weights in ASCII models	245
Chapter 12	The WinNonlin Toolbox	247
	Nonparametric superposition	247
	Data and assumptions	248
	Computation method	248
	Performing nonparametric superposition	250
	Output	251
	Semicompartmental modeling	252
	Data and assumptions	252
	Computation method	252
	Performing semicompartmental modeling	254
	Output	255
	Crossover design	256
	Data and assumptions	257
	Descriptive analysis	258
	Hypothesis testing	259
	Using the Crossover Design tool	260
	Output	262
	Numeric deconvolution	262
	Deconvolution methodology	263
	Using deconvolution	271
	Deconvolution variables	273
	Deconvolution dosing	274
	Deconvolution settings	275
	Deconvolution output	277
Chapter 13	The IVIVC Toolkit	279
	Wagner-Nelson and Loo-Riegelman methods	280
	Wagner-Nelson inputs and calculations	280
	Loo-Riegelman inputs and calculations	281
	Using the Wagner-Nelson or Loo-Riegelman method	282
	Data variables	283
	Dosing regimen	284
	Loo-Riegelman model parameters	285

Lambda Z ranges	286
Output options	289
Weighting	290
Title	292
Settings	293
Convolution	295
Data variables	297
Convolution settings	299
Convolution output	300
Convolution units	301
Levy plots	302
Output	304
The IVIVC Wizard	304
Minimizing the IVIVC Wizard	306
Project Status	306
Saving and loading IVIVC projects	307
In-vitro tab	308
In-vivo tab	309
Correlation tab	313
ASCII user model for correlations	315
Prediction tab	316
Options	318
IVIVC Wizard Output Objects	319
Using the IVIVC Wizard with PKS	323

Chapter 14 Linear Mixed Effects Modeling327

An example	327
The linear mixed effects model	330
Construction of the X matrix	332
Linear combinations of elements of β	336
Determining estimability numerically	337
Substitution of estimable functions for non-estimable functions	338
Fixed effects	340
Output parameterization	340
Variance structure	342
Repeated effect	343
Random effects	345
Least squares means	348

Contrasts	350
Joint versus single contrasts	351
Nonestimability of contrasts.	352
Degrees of freedom	353
Other options	353
Estimates	353
LinMix Wizard	354
LinMix limits and constraints.	354
Fixed effects	356
Variance structure	357
Random coefficients	359
Repeated measurements	360
Contrasts	360
Estimates	362
Least squares means	363
LinMix general options	364
LinMix output	365
Linear mixed effects text	365
Linear mixed effects workbook	365
Tests of hypotheses.	366
Akaike's Information Criterion (AIC)	367
Schwarz's Bayesian Criterion (SBC)	367
Hessian eigenvalues	368
Predicted values and residuals	368
Fixed effects parameter estimates	368
Coefficients.	369
Iteration history	369
Parameter key	369
Computational details	370
Restricted maximum likelihood	370
Newton's Algorithm for maximization of restricted likelihood.	372
Starting values	373
Convergence criterion	373
Satterthwaite's approximation for degrees of freedom	374
Comparing LinMix to ANOVA and SAS's PROC MIXED	376
LinMix and ANOVA	376
LinMix and PROC MIXED	377

Chapter 15 Bioequivalence	379
--	------------

Introduction to bioequivalence	379
Basic terms	379
Bioequivalence	380
The model and data	382
Linear model	382
Variance structures	382
Missing data	382
Variable name limits	382
Average bioequivalence	383
Study designs	383
Recommended models for average bioequivalence	383
Least squares means	385
Classical and Westlake confidence intervals	387
Two one-sided T-tests	388
Anderson-Hauck test	390
Power	390
Individual and population bioequivalence	391
Introduction	391
Bioequivalence criterion	392
Details of computations	394
Bioequivalence Wizard	399
Using average bioequivalence	400
Fixed effects for average bioequivalence	400
Random effects	402
Repeated variance	403
Bioequivalence options	404
Bioequivalence general options	405
Using population/individual bioequivalence	405
Fixed effects for population/individual bioequivalence	406
Bioequivalence options for population/individual bioequivalence	406
Bioequivalence output	407
Average bioequivalence	407
Population/Individual bioequivalence	408
Ratio tables	410

Chapter 16 Simulation	413
Simulating responses	413

Simulation of a compiled library model	413
Creating the input data	414
Setting up the model	414
Data variables	415
Model parameters	416
Dosing regimen	417
Model options	418
Running the simulation	418
Comparing dosing schemes	419
Dosing regimen	420
 Chapter 17 Scripting	421
Terminology	421
Writing and editing a script	422
Executing a script	424
Within WinNonlin	424
From the command line	424
From file manager or explorer	425
Scripting object properties and methods	425
Objects	425
Engines	427
Script file components	430
Note on saving files	432
Processing multiple profiles in script files	432
Special file formats	434
Scripting the Pharsight Knowledgebase Server	444
 Chapter 18 Using the Pharsight Knowledgebase Server	449
The PKS information structure	450
Studies	450
Scenario	451
Other documents	452
Dosing	452
Dosing worksheet	452
Append dosing	453
Dosing dialog	453
Accessing the PKS from WinNonlin	453
PKS sessions	454
Connecting to the PKS	455
Logging in	456

Creating a study	456
Study definition	457
Study Creation Wizard	458
Subject identifiers	459
Subject ID attributes	459
Subject data	460
Subject data attributes	461
Time	461
Sample data	462
Sample attributes	462
Dosing	463
Dosing attributes	464
Data Collection Point	465
Treatment	465
Study creation	466
Loading a study or scenario	467
Select observations	468
Select dose data	468
Scenario search	468
Creating scenarios	469
Adding data columns to a scenario	470
Adding non-WinNonlin documents to a scenario	470
Optimizing PKS/WNL performance	471
Change tracking and audit information	474
Audit information	474
Update reason	475
Name objects	476
Dosing information	476
Changing the dose regimen	477
Appending or updating a study	478
PKS libraries and object shares	479
PKS share	479
Libraries	480
Loading a library from the PKS	481
Adding or updating a library in the PKS	481
Library and share status	482
PKS Information	483
WinNonlin and PKS differences	484
Units	484
Studies and scenarios	485

Appendix A	Glossary	487
Appendix B	WinNonlin Model Libraries	507
	Notation and conventions	508
	Noncompartmental analysis	509
	NCA output parameters and their computation formulas.	510
	Plasma or serum data	511
	Urine data	516
	Sparse sampling (pre-clinical) data	518
	Drug effect data (model 220)	519
	Pharmacokinetic models	523
	ASCII models and files	523
	Model 1	524
	Model 2	524
	Model 3	525
	Model 4	526
	Model 5	526
	Model 6	526
	Model 7	527
	Model 8	528
	Model 9	528
	Model 10	529
	Model 11	530
	Model 12	531
	Model 13	532
	Model 14	532
	Model 15	532
	Model 16	533
	Model 17	534
	Model 18	535
	Model 19	536
	Pharmacodynamic models	537
	ASCII library	537
	Model 101	538
	Model 102	538
	Model 103	539
	Model 104	539
	Model 105	539
	Model 106	540
	Model 107	540
	Model 108	540

PK/PD link models	540
Notation.	541
Linking PK and PD models	541
Output parameters	542
Indirect response models	543
Models	544
Model 51	544
Model 52	544
Model 53	545
Model 54	545
Running an indirect response model	545
Michaelis-Menten models	546
Model 301	547
Model 302	548
Model 303	548
Model 304	549
Simultaneous PK/PD link models	549
Data	549
ASCII library.	549
Linear models	550
Sigmoidal/dissolution models	551
Model 601	552
Model 602	553
Model 603	553
Model 604	554
Appendix C Worksheet Functions	555
Appendix D Commands, Arrays and Functions	561
Appendix E Scripting Commands	583
Appendix F Warnings and Error Messages	641
Compartmental modeling	641
LinMix and Bioequivalence wizards	653
Table Wizard	656
Descriptive statistics	656

Noncompartmental analysis	657
Error messages	657
Warnings.	657
IVIVC	658
Scripting language.	659
 Appendix G References	661
Bioequivalence	661
<i>In vitro-in vivo</i> correlation and formulation science.	662
Pharmacokinetics	663
Regression and modeling	664
Sparse data analysis methods	666
Statistics	667
 Index	669

Introduction

WinNonlin editions, user documentation, nonlinear modeling and Pharsight contact information

Welcome. This chapter introduces WinNonlin, nonlinear modeling and the Pharsight suite of products and services under the following headings.

- “WinNonlin” on page 1
- “Modeling and nonlinear regression” on page 3
- “Pharsight Corporation” on page 9

WinNonlin

WinNonlin is a sophisticated tool for nonlinear modeling. It is particularly suited to pharmacokinetic and pharmacodynamic (PK/PD) modeling and non-compartmental analysis to evaluate data from bioavailability and clinical pharmacology studies. WinNonlin includes a large library of pharmacokinetic, pharmacodynamic, noncompartmental, PK/PD link, and indirect response models. It also supports creation of custom models to enable fitting and analysis of virtually any kind of data.

WinNonlin editions

WinNonlin is distributed in two editions, WinNonlin *Professional* and WinNonlin *Enterprise*. WinNonlin Professional is made for scientists involved with nonlinear modeling. It supports the analyses and generates the figures, tables, and listings required for regulatory submission. WinNonlin Enterprise provides all the features of WinNonlin Professional, and in addition, shares data with ODBC-compliant databases, and serves as a portal to the Pharsight® Knowledgebase Server™ (PKS), Pharsight's solution for secure storage, management and tracking of clinical and pre-clinical data. The

combination of WinNonlin Enterprise and PKS supports compliance with the Food and Drug Administration's (FDA) 21 CFR part 11 regulations.

In-vitro in-vivo correlation toolkit

With purchase of a special license, WinNonlin can be upgraded to include a suite of tools for analysis of in-vitro in-vivo correlations (IVIVC). The IVIVC suite includes a variety of correlation models, dissolution models, enhanced tools for convolution, deconvolution and plotting as well as a wizard to streamline setup of common IVIVC work flows.

WinNonlin user documentation

WinNonlin includes the following user documentation.

Table 1-1. WinNonlin user documentation

Document	Default location	Description
<i>Getting Started Guide</i>	...\WINNONLIN\USER DOCS and hardcopy with CD shipment	Installation and licensing instructions, and a step-by-step test to confirm that installation is complete and functional.
<i>User's Guide</i>	...\WINNONLIN\USER DOCS	Operating procedures for each software feature, information on the calculations performed by WinNonlin, and appendices detailing WinNonlin functions.
<i>Examples Guide</i>	...\WINNONLIN\USER DOCS	Step-by-step examples illustrating use of WinNonlin features in realistic scenarios.
Online help	Accessed from WinNonlin via the Help menu, F1 key, or Help button	Operating procedures for the software features and reference material on WinNonlin's computational engines.
Release notes	...\WINNONLIN\USER DOCS	Known issues and work-arounds in the current version.
Addendum to release notes	http://www.pharsight.com/support/support_sflogin.php	If issues and work-arounds are discovered after product release, an addendum to the release notes will be posted to the customer support web site with any software patches.
FAQ	http://www.pharsight.com/support/support_sflogin.php	Frequently asked questions and solutions for WinNonlin.

The manuals are not intended as a comprehensive text on nonlinear estimation, noncompartmental analysis, nor statistical analysis. See “References” on page 661 for a bibliography of references on the underlying theory.

Modeling and nonlinear regression

The value of mathematical models is well recognized in all sciences—physical, biological, behavioral, and others. Models are used in the quantitative analysis of all types of data, and with the power and availability of computers, mathematical models provide convenient and powerful ways of looking at data. Models can be used to help interpret data, to test hypotheses, and to predict future results. The concern here is “fitting” models to data—that is, finding a mathematical equation and a set of parameter values such that values predicted by the model are in some sense “close” to the observed values.

Most scientists have encountered linear models, models in which the dependent variable can be expressed as the sum of products of the independent variables and parameters. The simplest example is a line (linear regression):

$$Y = A + BX + \varepsilon$$

where Y is the dependent variable; X, the independent variable; A, the intercept and B, the slope. Two other common examples are polynomials such as:

$$Y = A_1 + A_2X + A_3X^2 + A_4X^3 + \varepsilon$$

and multiple linear regression.

$$Y = A_1 + A_2X_1 + A_3X_2 + A_4X_3 + \varepsilon$$

These examples are all “linear models” since the parameters appear only as coefficients of the independent variables.

Note: WinNonlin’s LinMix module handles linear regression, as well as Analysis of Variance and Analysis of Covariance.

In nonlinear models, at least one of the parameters appears as other than a coefficient. A simple example is the decay curve:

$$Y = Y_0 \exp(-BX)$$

This model can be linearized by taking the logarithm of both sides, but as written it is nonlinear.

Another example is the Michaelis-Menten equation:

$$V = \frac{V_{max}C}{K_m + C}$$

This example can also be linearized by writing it in terms of the inverses of V and C , but better estimates are obtained if the nonlinear form is used to model the observations (Endrenyi, ed. (1981), pages 304-305).

There are many models that cannot be made linear by transformation. One such model is the sum of two or more exponentials, such as

$$Y = A_1 \exp(-B_1 X) + A_2 \exp(-B_2 X)$$

All such models are called nonlinear models, or nonlinear regression models. This manual is not intended to be a comprehensive text on nonlinear estimation theory. However, two good references for the topic of general nonlinear modeling are Draper and Smith (1981) and Beck and Arnold (1977). The books by Bard (1974), Ratkowsky (1983) and Bates and Watts (1988) give a more detailed discussion of nonlinear regression theory. Modeling in the context of kinetic analysis and pharmacokinetics is discussed in Endrenyi, ed. (1981) and Gabrielsson and Weiner (2001).

All pharmacokinetic models derive from a set of basic differential equations. When the basic differential equations can be integrated to algebraic equations, it is most efficient to fit the data to the integrated equations. But it is also possible to fit data to the differential equations by numerical integration. WinNonlin can fit models defined in terms of algebraic equations, differential equations or a combination of the two types of equations.

Given a model of the data, and a set of data, how does one “fit” the model to the data? Some criterion of “best fit” is needed, and many have been suggested, but only two are much used. These are *maximum likelihood* and *least squares*. With certain assumptions about the error structure of the data (no random effects), the two are equivalent. A discussion of using nonlinear least squares to obtain maximum likelihood estimates can be found in Jennrich and Moore (1975). In least squares fitting the “best” estimates are those that minimize the sum of the squared deviations between the observed values and the values predicted by the model.

The sum of squared deviations can be written in terms of the observations and the model. In the case of linear models it is easy to compute the least squares estimates. One equates to zero the partial derivatives of the sum of squares with respect to the parameters. This gives a set of linear equations with the estimates as unknowns; well-known techniques can be used to solve these linear equations for the parameter estimates.

This method will not work for nonlinear models, because the system of equations that results by setting the partial derivatives to zero is a system of non-

linear equations without general solution methods. Consequently, any method for computing the least squares estimates in a nonlinear model must be an iterative procedure. That is, initial estimates of the parameters are made and then in some way modified to give better estimates, i.e., estimates that result in a smaller sum of squared deviations. The iteration continues until hopefully the minimum (or least) sum of squares is reached.

In the case of models with random effects, such as population PK/PD models, the more general method of maximum likelihood (ML) must be employed. In ML, the likelihood of the data is maximized with respect to the model parameters. WinNonMix fits models of this variety.

Since it is not possible to know exactly what the minimum is, some stopping rule must be given at which point it is assumed that the method has converged to the minimum sum of squares. A more complete discussion of the theory underlying nonlinear regression can be found in the books cited previously.

Model fitting algorithms and features of WinNonlin

WinNonlin is a computer program for estimating the parameters in a nonlinear model, using the information contained in a set of observations. WinNonlin can also be used to simulate data from a model. That is, given a set of parameter values and a set of values of the independent variables in the model, WinNonlin will compute values of the dependent variable and also estimate the variance-inflation factors for the estimated parameters.

A computer program such as WinNonlin is only a tool for the modeling process. As such it can only take the data and model supplied by the researcher and find a “best fit” of the model to the data. The program cannot determine the correctness of the model nor the value of any decisions or interpretations based on the model. The program does, however, provide some information about the “goodness of fit” and about how well the parameters are estimated.

It is assumed that WinNonlin users have some knowledge of nonlinear regression. WinNonlin provides the “least squares” estimates of the model parameters, as discussed above, with a choice of three algorithms for minimizing the sum of squared residuals: the simplex algorithm of [Nelder and Mead \(1965\)](#), the Gauss-Newton algorithm with the modification proposed in [Hartley \(1961\)](#), and a Levenberg-type modification of the Gauss-Newton algorithm ([Davies and Whitting \(1972\)](#)).

The simplex algorithm is a very powerful minimization routine; its usefulness has formerly been limited by the extensive amount of computation it requires.

The power and speed of current computers make it a more attractive choice, especially for those problems where some of the parameters may not be well-defined by the data and the sum of squares surface is complicated in the region of the minimum. The simplex method does not require the solution of a set of equations in its search and does not use any knowledge of the curvature of the sum of squares surface. When the Nelder-Mead algorithm converges, WinNonlin restarts the algorithm using the current estimates of the parameters as a “new” set of initial estimates and resetting the step sizes to their original values. The parameter estimates that are obtained after the algorithm converges a second time are treated as the “final” estimates. This modification helps the algorithm locate the global minimum (as opposed to a local minimum) of the residual sum of squares for certain difficult estimation problems.

The Gauss-Newton algorithm uses a linear approximation to the model. As such it must solve a set of linear equations at each iteration. Much of the difficulty with this algorithm arises from singular or near-singular equations at some points in the parameter space. WinNonlin avoids this difficulty by using singular value decomposition rather than matrix inversion to solve the system of linear equations. (See [Kennedy and Gentle \(1980\)](#).) One result is that the iterations do not get “hung up” at a singular point in the parameter space, but rather move on to points where the problem may be better defined. To speed convergence, WinNonlin uses the modification to the Gauss-Newton method proposed by Hartley (1961) and others; with the additional requirement that at every iteration the sum of squares must decrease.

Many nonlinear estimation programs, including the original NONLIN, have found Hartley's modification to be very useful. [Beck and Arnold \(1977\)](#) compare various least squares algorithms and conclude that the Box-Kanemasu method (almost identical to Hartley's) is the best in many circumstances.

As indicated, the singular value decomposition algorithm will always find a solution to the system of linear equations. However, if the data contain very little information about one or more of the parameters, the adjusted parameter vector may be so far from the least squares solution that the linearization of the model is no longer valid. Then the minimization algorithm may fail due to any number of numerical problems. One way to avoid this is by using a 'trust region' solution; that is, a solution to the system of linear equations is not accepted unless it is sufficiently close to the parameter values at the current iteration. The Levenberg and Marquardt algorithms are examples.

Note: The Gauss-Newton method with the Levenberg modification is WinNonlin's default estimation method.

Trust region methods tend to be very robust against ill-conditioned data sets. There are two reasons, however, why one may not want to use them. (1) They require more computation, and thus are not efficient with data sets and models that readily permit precise estimation of the parameters. (2) More importantly, the trust region methods obtain parameter estimates that are often meaningless because of their large variances. Although WinNonlin gives indications of this, it is possible for users to ignore this information and use the estimates as though they were really valid. For more information on trust region methods, see Gill, Murray and Wright (1981) or Davies and Whitting (1972).

In WinNonlin the partial derivatives required by the Gauss-Newton algorithm are approximated by difference equations. There is little, if any, evidence that any nonlinear estimation problem is better or more easily solved by the use of the exact partial derivatives.

To fit those models that are defined by systems of differential equations, the RKF45 numerical integration algorithm is used (Shampine et. al., 1976). This algorithm is a 5th order Runge-Kutta method with variable step sizes.

It is often desirable to set limits on the admissible parameter space, resulting in “constrained optimization.” For example, the model may contain a parameter that must be non-negative, but the data set may contain so much error that the actual least squares estimate is negative. In such a case it may be preferable to give up some properties of the unconstrained estimation in order to obtain parameter estimates that are physically realistic. At other times, setting reasonable limits on the parameter may prevent the algorithm from wandering off and getting lost. For this reason, it is recommended that users always set limits on the parameters (the means of doing this is discussed in the modeling chapters of this document). In WinNonlin two different methods are used for bounding the parameter space. When the simplex method is used, points outside the bounds are assigned a very large value for the residual sum of squares. This sends the algorithm back into the admissible parameter space.

With the Gauss-Newton algorithms, two successive transformations are used to affect the bounding of the parameter space. The first transformation is from the bounded space as defined by the input limits to the unit hypercube. The second transformation uses the inverse of the normal probability function to go to an infinite parameter space. This method of bounding the parameter space has worked extremely well with a large variety of models and data sets.

Bounding the parameter space in this way is in effect a transformation of the parameter space. It is well known that a reparameterization of the parameters will often make a problem more tractable. (See Draper and Smith (1981) or Ratkowsky (1983) for discussions of reparameterization.) When encountering

a difficult estimation problem, the user may want to try different bounds to see if the estimation is improved. It must be pointed out that it may be possible to make the problem worse with this kind of transformation of the parameter space. However, experience suggests that this will rarely occur. Reparameterization to make the models more linear also may help.

Because of WinNonlin's flexibility and generality, and the complexity of nonlinear estimation, many options and specifications may be supplied to the program. To make WinNonlin easy to use, many of the most commonly used options are set as WinNonlin defaults.

WinNonlin is capable of estimating the parameters in a very large class of nonlinear models. Fitting pharmacokinetic and pharmacodynamic models is a special case of nonlinear estimation, but it is important enough that WinNonlin is supplied with a library of the most commonly used pharmacokinetic and pharmacodynamic models. To speed execution time, the main WinNonlin library has been built in, or compiled. However, ASCII library files corresponding to the same models as the built-in models, plus an additional utility library of models, are also provided. The model libraries and their uses are described in "[WinNonlin Model Libraries](#)" on page 507. Users can also create custom libraries, as explained in "[User Models](#)" on page 219.

Summary of regression methods

Method 1: Nelder-Mead simplex

Method 1 is a very powerful algorithm. Since it doesn't require the estimated variance-covariance matrix as part of its algorithm, it often performs very well on ill-conditioned data sets. *Ill-conditioned* indicates that for the given data set and model, there isn't enough information contained in the data to precisely estimate all of the parameters in the model or that the sum of squares surface is highly curved. Although the procedure works well, it can be slow. When used to fit a system of differential equations, it can be extremely slow.

Method 2: Gauss-Newton with Levenberg and Hartley modification

Method 2 is the default, and also performs well on a wide class of problems, including ill-conditioned data sets. Although the Gauss-Newton method does require the estimated variance-covariance matrix of the parameters, the Levenberg modification suppresses the magnitude of the change in parameter values from iteration to iteration to a reasonable amount. This enables the method to perform well on ill-conditioned data sets.

Method 3: Gauss-Newton with Hartley modification

Method 3 is another Gauss-Newton method. It is not as robust as Methods 1 and 2 but is extremely fast. Method 3 is recommended when speed is a consideration or when maximum likelihood estimation or linear regression analysis is to be performed. It is not recommended for fitting complex models.

Pharsight Corporation

Pharsight, a Certara company, is a market-leading provider of software products and scientific consulting services to help pharmaceutical and biotechnology companies improve their drug development process, regulatory compliance and strategic decision-making. Established in 1995, the company's goal is to help customers reduce the time, cost and risk of drug development, as well as optimize the post-approval marketing and use of pharmaceutical products. Pharsight leverages expertise in its software tools and in the disciplines of pharmacology, drug and disease modeling, human genetics, biostatistics, strategic decision-making, and regulatory strategy.

Headquartered in St. Louis, Missouri, with more than 1200 customers worldwide, Pharsight products and services are used by all of the world's top 50 pharmaceutical firms. More information is available on the web at www.pharsight.com.

Products

Pharsight offers a comprehensive suite of software products, including WinNonlin[®], WinNonMix[®], Trial Simulator[™], Drug Model Explorer[™] and the Pharsight Knowledgebase Server[™] suite of products. For further information, please visit Pharsight on the web at www.pharsight.com.

Scientific and consulting services

Our scientific consultants can design and optimize clinical trials and clinical development programs to meet an individual company's objectives. Pharsight Scientific Services are project-centered, planned and delivered in accordance with specific needs and agreed-upon objectives. Consultants can facilitate:

- Gathering and interpreting data
- Building drug and population models

- Guiding variability testing and trial optimization
- Addressing specific development program objectives
- Process automation

Training

Pharsight Corporation offers workshops and training courses at regular intervals. On-site training is available by request. Contact Pharsight for schedules, pricing and availability. The training schedule is also available on Pharsight's corporate web site at http://www.pharsight.com/training/training_home.php.

Pharsight contact information

Technical support

Please consult the documentation (see “WinNonlin user documentation” on page 2) to address questions. If further assistance is needed, please contact Pharsight technical support by e-mail or web (preferred), phone, fax or post.

E-mail: support@pharsight.com (fastest response time)
Web: http://www.pharsight.com/support/support_sflogin.php
Phone/voice mail: +1-919-852-4620
Fax: +1-919-859-6871
Post: Pharsight Corporation
5625 Dillard Drive, Suite 205
Cary, North Carolina 27518

For the most efficient service, please e-mail a complete description of the problem, including copies of input data, model file and instructions to WinNonlin, as well as the ASCII output, if applicable, detailing the error.

Licensing and product upgrades

For license renewals and product upgrades, please contact Pharsight sales:

E-mail: sales@pharsight.com
Telephone: +1-888-708-7444 (from US only)
Fax: +1-650-314-3811

Request assistance with Pharsight software licensing by e-mail or telephone:

E-mail: license@pharsight.com
Telephone: +1-919-852-4620

Customer feedback

Please submit requests for product enhancements and defect corrections by e-mail, fax or through Pharsight's customer support web site, as follows.

E-mail: support@pharsight.com
Web: http://www.pharsight.com/support/support_sflogin.php
Fax: +1-919-859-6871

Data Management

Working with WinNonlin data windows and files

This chapter covers data handling in WinNonlin, under the following topics.

- “[WinNonlin windows and files](#)” on page 13: WinNonlin workbook, text and chart windows and their supported file types.
- “[Workspaces](#)” on page 19: saving and reloading WinNonlin analyses.
- “[Printing](#)” on page 21
- “[WinNonlin options](#)” on page 28: default file locations, worksheet behaviors, and modeling and analysis default settings.
- “[Data dependencies and origins](#)” on page 33: refreshing analysis output when source data change, and unlocking output (removing dependency on source data).
- “[Data history worksheets and log files](#)” on page 37: tracking changes to data and operations in WinNonlin.

See also: “[Data import and export](#)” on page 75.

WinNonlin windows and files

WinNonlin provides three window types for managing data objects:

- Workbooks hold tabular data in one or more *worksheets* that support spreadsheet operations and formats. See “[Workbooks](#)” on page 41.
- Chart windows contain plots. See “[Charts](#)” on page 115 for instructions on creating and formatting plots.
- Text windows contain ASCII text representing a model, script, or modeling output.

To create a new window:

1. Click the **New** (left-most) button on the WinNonlin tool bar and select **New Workbook**, **New Text** or **New Chart** from the drop-down menu that appears. (Before using New Chart, open the workbook window containing data to be plotted. New Chart will open the [Chart Wizard](#) to create the plot(s).)

or

1. Choose **File>New** from the menus (**Alt+F, N**). The New dialog appears.
2. Select the appropriate radio button.
3. Click **OK**. The new window opens.

Any window can be hidden to reduce clutter in the WinNonlin parent window (see “[Hidden windows](#)” on page 18).

To create a copy of an existing workbook:

1. Open the workbook in WinNonlin.
2. Choose **Tools>Copy Workbook** from the WinNonlin menus.

File types

Each window type can load and save the file types listed in [Table 2-1](#). In addition, analysis settings entered in the WinNonlin user interface can be saved to file, as noted in the last row of [Table 2-1](#).

Table 2-1. Supported file formats

Window	Example	File formats: load/import	File formats: save/export
Workbooks	Subject data Modeling output Dosing data Model parameters	ASCII (*.DAT, *.DTA, *.PRN, *.CSV) Microsoft Excel 4.0, 5.0/95, 97, 2000 (*.XLS) Pharsight Workbook Object (*.PWO) SAS® Transport (*.XPT, *.STX) WinNonlin Data Object (*.WDO)	ASCII data (*.DAT, *.PRN, *.CSV) ASCII model parameters (*.DTA) Microsoft Excel 5.0/95, 97 (*.XLS) Pharsight Workbook Object (*.PWO) SAS® Transport (*.XPT, *.STX)
Chart windows		Pharsight Chart Object (*.PCO) WinNonlin Graph Object (*.WGO)	Pharsight Chart Object (*.PCO) Bitmap image (*.BMP) Vector graphic (*.JPG) Windows Metafile (*.WMF)

Table 2-1. Supported file formats (continued)

Window	Example	File formats: load/import	File formats: save/export
Text windows	Modeling output ASCII models Scripts Notes	ASCII text (*.TXT) Library model(s) (*.LIB) Rich text (*.RTF) Pharsight Script (*.PSC) Pharsight Text Object (*.PTO) WinNonlin Script (*.WSC) WinNonlin Text Object (*.WTO)	ASCII text (*.TXT) Library model(s) (*.LIB) Rich text (*.RTF) Pharsight Script (*.PSC) Pharsight Text Object (*.PTO)
Model settings (Load/Save in modeling dialogs)	ANOVA settings Bioequivalence Model settings	ANOVA Command File (*.ANV) LinMix or Bioequivalence Command File (*.LML) Pharsight Model Object (*.PMO) WinNonlin Model Object (*.WMO) WinNonlin Command File (*.CMD)	LinMix or Bioequivalence Command File (*.LML) Pharsight Model Object (*.PMO)

Note: Excel 4.0 files and WinNonlin object files (*.WDO, *.WTO, *.WGO, *.WMO), command files (*.CMD) and script files (*.WSC) are import-only. Changes must be saved to a more recent Excel file type or Pharsight file (*.PWO, *.PTO, *.PCO, *.PMO, *.PSC).

Pharsight objects (*.PWO, *.PTO, *.PCO) preserve all formatting, data settings, sorting and units applied in WinNonlin. Pharsight chart and workbook objects can save multiple worksheets in a workbook, or multi-chart windows.

A Pharsight Workbook Object (*.PWO) saves all information in a workbook, including all worksheets, data sort order, units, inclusions/exclusions and formatting.

A Pharsight Model Object (*.PMO) contains the current model (the ASCII code or the number of the compiled model) and any associated data, including values for the model parameters, constants (dosing), data variables, and modeling options. When an ASCII model is used with a Pharsight Model Object, the source ASCII model is updated to include any changes in WinNonlin.

A Pharsight Chart Object (*.PCO) saves all charts in a chart window, as well as the graph attributes, including formatting and links to the graph data.

A Pharsight Text Object (*.PTO) saves all text and formats in a text window.

Non-Pharsight file formats may limit the information saved, as follows.

ASCII data files save the text of the active worksheet or text window, without any formatting, data inclusions/exclusions, or units. Multiple worksheets cannot be saved to a single ASCII file.

Rich text files (*.RTF) preserve text and formatting from text windows.

Excel files preserve data, formatting, data exclusions, and multiple worksheets in a workbook. However, units and column headers are not saved.

When a chart is saved to a .WMF, .JPEG or .BMP file, a snapshot of the plot that appears on-screen is saved. Saving a chart window as a Pharsight Chart Object saves all plots in the window and their attributes (formatting, etc.).

Note: When a chart is saved as a Windows Metafile (*.WMF) WinNonlin prompts whether to “Include placeable header information.” Some applications that read Metafiles will expect this information; others will not. If you will open the chart in a Microsoft application, such as Word or Excel, click **Yes**.

Troubleshooting: loading tabular data

Non-numeric data

Occasionally, when transferring data from other packages, non-numeric data may appear in workbook cells that should contain numeric data. This may be caused by leading or trailing blank spaces in some fields. One indicator that the data are non-numeric is that non-numeric data will be left justified rather than centered in the cells. Non-numeric data may cause problems in WinNonlin calculations, owing in part to the different sort order inherent in non-numeric data.

To convert non-numeric data to numeric data; create a new variable using a formula or function, such as $New = Old + 0$. Copy the new column, then paste only the values (not the formula) to another column. Last, delete the other two columns.

Missing data

If a data set includes missing values, and the cells with the missing values do not have a gray background, then check that the character denoting missing values was specified when the file was opened (as set in the Options dialog accessed from the Open dialog).

Column problems

If all of the data appear in the first column, check the Options (accessed from the Open dialog) to ensure that the correct delimiter is specified when loading the data.

If a number of blank columns or cells appear where they do not belong, check the Treat Consecutive Delimiters as One option on the Options dialog.

WinNonlin and WinNonMix file compatibility

The following table outlines file types that can and cannot be shared between WinNonMix and WinNonlin.

Table 2-2. WinNonlin and WinNonMix file compatibility

File Type	WinNonlin	Compatibility	WinNonMix
ASCII Data	*.dat, *.dta, *.prn, *.csv	Identical	*.dat, *.dta, *.prn, *.csv
Excel Data files	Excel 4.0 (*.xls) Excel 5.0/95 (*.xls) Excel 97 (*.xls)	Identical	Excel 4.0 (*.xls) Excel 5.0/95 (*.xls) Excel 97 (*.xls)
Text files	*.txt, *.lib	Identical	*.txt, *.lib
Rich text files	*.rtf	Identical	*.rtf
WinNonlin files: Data Text Graph Model	*.wdo *.wto *.wgo *.wmo	Identical Identical Identical NA	*.wdo *.wto *.wgo Not available
Pharsight files: Data Text Chart Model	*.pwo *.pto *.pco *.pmo	Identical Identical Identical Not compatible	*.pwo *.pto *.pco *.pmo
SAS Transport files	*.xpt, *.stx	NA	Not available
ANOVA command files ^a	*.anv	Identical	*.anv
LinMix command files	*.lml	NA	Not available

Table 2-2. WinNonlin and WinNonMix file compatibility (continued)

File Type	WinNonlin	Compatibility	WinNonMix
Bioequivalence command files	*.lml	NA	Not available
Model object files ^b	*.pmo	Not compatible	*.pmo
Dosing files	ASCII data: *.dat, *.dta	Not compatible	ASCII data: *.dat, *.dta
Model parameters	ASCII data: *.dat, *.dta	Not compatible	ASCII data: *.dat, *.dta
Lambda Z range	ASCII data: *.dat, *.dta	NA	Not available
Partial areas	ASCII data: *.dat, *.dta	NA	Not available
Graph template	*.gtp	Identical	*.gtp
Table definition files	*.tdf	Identical	*.tdf
Command files	*.cmd (import only)	NA	Not available
Script files	*.psc, *.wsc	NA	Not available
ODBC import and export files	*.imp, *.exp	Identical	*.imp, *.exp
PKS files	*.pks	NA	Not available

- a. ANOVA command files can be imported into the Linear Mixed Effects Wizard or the Bioequivalence Wizard in WinNonlin. New *.ANV files cannot be created.
- b. WinNonMix model object files (*.PMO) include more information than WinNonlin model object files.

Hidden windows

Windows of all types can be hidden so as not to clutter the parent window. The Hide Window command leaves the active window open, but hidden from the WinNonlin workspace and commands. Hidden windows do not appear in list boxes (for modeling, Descriptive Statistics, Table Wizard, and Chart Wizard, for example). Hidden windows cannot be saved to files or workspaces.

To hide a window:

1. Open or select the window to be hidden.
2. Choose **Window>Hide Window** from the WinNonlin menus.

To un-hide a window:

1. Choose **Window>Unhide Windows...** from the WinNonlin menus.
2. Check the window(s) to unhide and click **OK**.

Workspaces

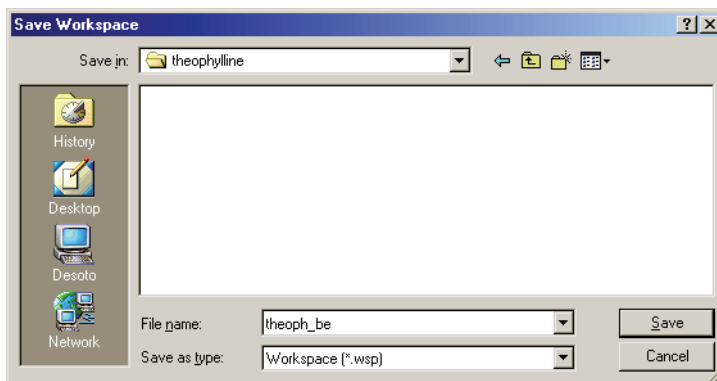
A *workspace* is a snapshot of all current dose and model settings, and all data object windows that are open in WinNonlin. This information can be saved to a workspace file (*.wsp), to be reloaded in future WinNonlin sessions. All workbooks, text windows, charts, and model settings that have not yet been saved to disk are implicitly saved to separate files, which are named after the workspace. Model settings, if not previously-saved, are saved to a Pharsight Model Object (*.PMO). Unsaved workbook data, such as modeling output, are saved to a Pharsight Workbook Object (*.PWO). The workspace file contains the relative paths to these objects.

Note: Pharsight Model Object (*.PMO) files may use absolute paths to data files. If a workspace containing model information is moved, then loaded, WinNonlin may prompt the user for the location of some source data files.

To save current work to a workspace:

1. Choose **File>Save Workspace (Alt+F, V)** from the WinNonlin menus.

If the workspace is new, the **Save Workspace** dialog will appear.



2. Navigate to the appropriate directory and enter a name for the workspace.
Limit the name to alphanumeric, dash and space characters.
3. Click **Save**.

To create a new, empty workspace:

1. Choose **File>New Workspace (Alt+F, W)** from the WinNonlin menus.
WinNonlin will prompt the user to save any unsaved work, close all windows, and start a new, empty, workspace.

To save an existing workspace to a new workspace name or location:

1. Select **File>Save Workspace As (Alt+F, K)** from the WinNonlin menus.
The Save Workspace dialog appears.
2. Navigate to the appropriate directory and enter a name for the workspace.
Limit the name to alphanumeric, dash and space characters.
3. Click **Save**.

To load a saved workspace:

1. Choose **File>Load Workspace (Alt+F, L)** from the WinNonlin menus.
2. In the Load Workspace dialog, navigate to the appropriate directory and double-click the desired workspace.

Printing

Customize print output using the following features:

- “Page setup” on page 21 to set formatting and layout options.
- “Print preview” on page 25 to check print output before printing.
- “Print” on page 25 to print the current window.
- “Print all” on page 27 to print selected items from any open windows.

Page setup

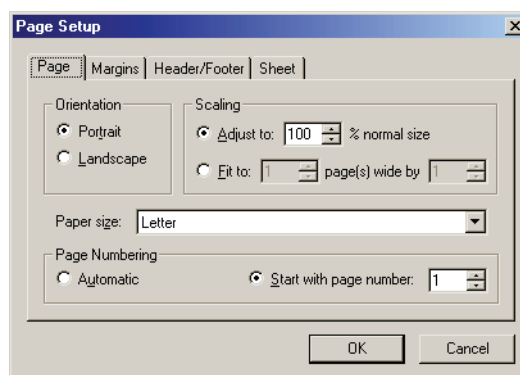
To set page settings for print output:

1. Select **Page Setup** from the **File** menu (**Alt+F, G**). The page setup dialog appropriate to the active object appears. See:
 - “Page setup for workbooks”
 - “Page setup for text windows” on page 24 (including model windows)
 - “Page setup for charts” on page 25

Page setup for workbooks

Page setup options for workbook windows are specified on four tabs:

Page Tab



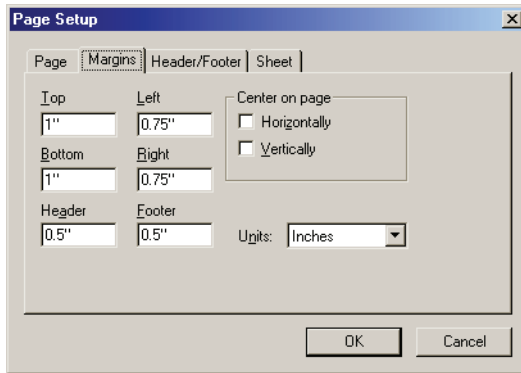
Orientation: Select **Portrait** (vertical) or **Landscape** (horizontal) page layout.

Scaling: The workbook can be scaled as a percentage of the normal size or select **Fit to** in order to make it fit on a certain number of pages.

Paper Size: Select the size paper to use.

Page Numbering: Select **Automatic** to start numbering pages at 1, or check the **Start with page number** check box and enter the first page number.

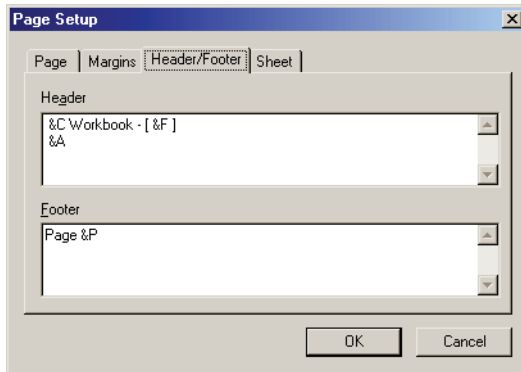
Margins Tab



Enter margins for the worksheet, including the header (top) and footer (bottom) margins, in inches or centimeters, as selected under Units.

Check **Horizontally** and/or **Vertically** to center the output on the page horizontally (across), vertically (up and down) or both.

Headers and Footers Tab



Enter the alignment, formatting and content for page headers and footers using the codes listed in the tables below and/or text to be printed verbatim. Any entry not preceded with the ampersand "&" will be reproduced as

entered. Codes must be space-separated. Alignment codes must precede font-related codes. Special character codes for printing worksheet-specific information such as worksheet titles must appear last. If the codes are entered in the wrong order WinNonlin may ignore some of them. Codes and text are centered unless &L or &R is specified.

Table 2-3. Alignment codes

&L	Left-aligns the characters that follow.
&C	Centers the characters that follow.
&R	Right-aligns the characters that follow.

Table 2-4. Font codes

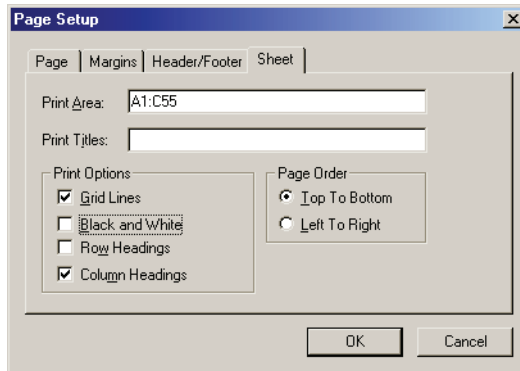
&B	Uses a bold font.
&I	Uses an italic font.
&U	Underlines the header.
&S	Strikeout font.
&"fontname"	Uses the specified font.
&nn	Uses the specified font size - must be a two digit number.

Table 2-5. Special codes for workbook information

&F	Prints the Workbook Name.
&A	Prints the Worksheet Name.
&P	Prints the Page Number.
&D	Prints the Current Date.
&T	Prints the Current Time.
&L	Left-aligns the characters that follow.
&C	Centers the characters that follow.
&R	Right-aligns the characters that follow.
&P + number	Prints the page number plus number.
&P - number	Prints the page number minus number
&&	Prints an ampersand

Table 2-5. Special codes for workbook information

&N	Prints the total number of pages in the document
----	--

Sheet Tab

Print Area: Enter the range of columns and rows to print using absolute or relative cell references. Separate non-contiguous ranges with commas.

Print Titles: Specify rows or columns for which titles should be printed on each page, as absolute or relative cell references. Separate non-contiguous ranges with commas. If a row is entered, the title is printed at the top of each page. If a column is selected, the title is printed at the left edge of each page. Multiple rows or columns may be selected, but they must be adjacent.

Print Options: Set whether or not grid lines and row and column headers will print. Output can be specified to print in black and white, so that items with shading or color do not use grayscale (gray “missing” cells will appear white).

Page Order: Specify the print order: **Top to Bottom** (left columns print before those farther right) or **Left to Right** (top rows print before the lower rows).

Page setup for text windows

Page setup options for text windows sets the page orientation and margins.

Orientation: Select **Portrait** (vertical) or **Landscape** (horizontal) page layout.

Margins: enter the page margins in inches.

Page setup for charts

Page setup for charts sets chart size and the number of charts per page.

Orientation: Select whether the graph is to be printed on the page vertically (**Portrait**) or horizontally (**Landscape**).

Scaling: Select **Actual Size** to print the chart at its original size. Select **Best Fit** to scale the graph proportionally to fit the page (recommended).

Margins: Enter a value in inches or centimeters, depending on the Windows settings, to specify the top, bottom, left and right margins of the page.

Layout: Select **Screen Layout** to print the graph at the size it displays on the screen. Select **Layout for Printer** to expand the scale of the graph to the size and layout of the selected paper orientation (similar to the Scaling function).

Multiple charts per page: If multiple charts are to be printed per page, select this check box and select the format from the list box. Once this option is selected, the controls for scaling and layout are disabled, as charts will be scaled to fit the selected number across and down the page.

Print preview

To preview print output:

1. Make sure that the object to print is the active window in WinNonlin.
2. Select **File>Print Preview** from the WinNonlin menus.
3. A Print Preview window appears with the layout from the Page Setup dialog.
4. Click **Next** to scroll through multiple pages.
5. Click **Close** to close the window and return to WinNonlin.

Print

To print the active window using the default print settings:

1. Click the **Print** button in the WinNonlin tool bar.

To print the active window:

1. Select the window to be printed.
2. Choose **File>Print (Alt+F, P)** from the WinNonlin menus. The Print dialog appears.
3. Set options as appropriate for the object type (workbook, chart, text). See also: “[Printing workbooks](#)” on page 26 and “[Printing text objects and models](#)” on page 26.
4. Click **Preview** to view the expected print output or **OK** or **Print**.

Note: When printing charts, use the [Page setup](#) dialog first in order to specify multiple charts per page and chart size.

If lines appear fuzzy in printed plots, change line colors to black.

Printing workbooks

1. Select the Print Range:
 - The entire worksheet: Select **All** in the Print Range box.
 - Selected Pages: Select **Pages** in the Print Range box and enter the page number(s) as a comma-separated list. Use a hyphen to indicate ranges.
2. Under Print What, select whether to print all worksheets (**Entire Workbook**) or only the currently selected worksheet(s) (**Selected Sheet(s)**).

Note: To select multiple sheets in a workbook, hold down the **Ctrl** key and click on each worksheet tab.

Printing text objects and models

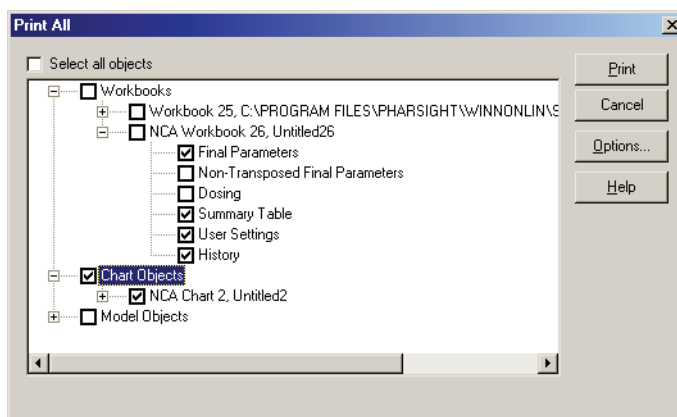
1. Select the appropriate printer and paper size.
2. Enter the number of copies to print.

Note: Page numbers are printed for text objects only when the printing is initiated via the print button on the tool bar.

Print all

To print a selection from the open windows:

1. Open the window(s) to print.
2. Choose **File>Print All** from the WinNonlin menus.
3. To print all open windows, including all worksheets and charts, click the **Print** button in the Print All dialog.



4. To select specific items to print:
 - a. Uncheck **Select all objects**.
 - b. Click the “+” symbols to view individual windows, worksheets or charts.
 - c. Check the box next to the items to print.
5. Use the **Options** button to set up the page layout for charts. (See “**Print all options**” below.)
6. Click **Print**.

Print all options

Multiple charts per page: Check this box and select the layout below to put more than one chart on a page, with page breaks between sets of charts.

WinNonlin options

WinNonlin provides options to set default file locations, worksheet behaviors, and modeling and analysis default settings as described in [Table 2-6](#).

Table 2-6. WinNonlin options

Category	Options
Directories	<ul style="list-style-type: none"> default directory for WinNonlin Library and User Library files
Workbooks	<ul style="list-style-type: none"> whether to display formulas in worksheets direction in which the Enter key moves the cursor in a worksheet value used to denote missing data in worksheets
Models	<ul style="list-style-type: none"> default output: workbooks, charts, and/or text with or without page breaks default orientation of the Final Parameters output table default method for the calculation of AUC in noncompartmental analyses
Tables	<ul style="list-style-type: none"> treatment of group variables in the table generator default use of page breaks in tables
Units	<ul style="list-style-type: none"> display of units with column headers in worksheets, chart labels and tables
Licensing	<ul style="list-style-type: none"> disable/enable warning message for impending license expiration

Directories

To set or change the default model library directories:

1. Choose **Tools>Options** from the WinNonlin menus and click on the **Directories** tab of the [WinNonlin options](#) dialog, if it is not already open.
2. To set the default file location for WinNonlin model library or user model library files, click the **Browse** button for that directory.
3. Select the desired drive and directory. The picture of the folder opens.
4. Click **Apply** to set these options and keep the dialog open or **OK** to keep these settings and close the [WinNonlin options](#) dialog.

Workbooks

To set or change default workbook options:

1. Choose **Tools>Options** from the WinNonlin menus and open the **Workbooks** tab of the [WinNonlin options](#) dialog.

2. Check the **Show Formulas** check box to display formulas, rather than the values computed by formulas, in individual worksheet cells. Note that the formula bar at the top of each worksheet displays any formula in the current cell.
3. Choose the direction of cursor movement between worksheet cells when the Enter key is pressed: check **Enter Moves Down** to move the cursor down the column; else, the cursor will move to the right when the Enter key is pressed.
4. To specify a character or string to represent missing data, enter it in the box labeled **Missing Value**. Any cell containing this string will be grayed and treated as missing in (excluded from) modeling and analyses.
5. Click **Apply** to set these options and keep the dialog open or **OK** to keep these settings and close the [WinNonlin options](#) dialog.

Models

To set default model options:

1. Choose **Tools>Options** from the WinNonlin menus and open the **Models** tab in the [WinNonlin options](#) dialog.
2. Use the check boxes to select the default form(s) of modeling output:
 - **Workbook**: check to include tabular results in a workbook.
 - **Charts**: check to include charts of model data and results.
 - **Text**: check to include text output listing model settings, fitting progress and final results. Check **Page breaks** for paginated output.
3. Set the layout for the Final Parameter output worksheet: Checking **Transpose Final Parameter Table** will present each parameter in a separate column. This layout is a useful for performing additional analyses, such as descriptive statistics, on modeling output. A second worksheet called Non-transposed Final Parameters will present parameters in separate rows. [Figure 2-1](#) and [Figure 2-2](#) present an example of each layout.

	Subject	Form	V_F	V_F_StdError	V_F_CV%	V_F_UnivarCI_Lower	V_F_UnivarCI_Upper
1	1	c	14.563861	21264.661884	146009.78	-50268.775001	50297.902723
2	1	t	30.111986	3043.156234	10106.13	-7416.228443	7476.452414
3	2	c	18.704861	1241486.068802	6637237.70	-2935652.928767	2935690.338488
4	2	t	6.775877	2334.505283	34453.18	-5705.556888	5719.108643
5	3	c	37.274154	48230.382309	129393.63	-114010.373082	114084.921389
6	4	t	47.894912	159.506270	333.03	-342.403152	438.192976

Figure 2-1. Example modeling output: transposed Final Parameters.

If the Transpose Final Parameters option is not checked, the Final Parameters worksheet will use the untransposed format, and the second worksheet, entitled Transposed Final Parameters, will contain the transposed version. This option is provided primarily for WinNonlin scripts that depend on the transposed format the Final Parameters worksheet.

	Subject	Form	Parameter	Estimate	StdError	CV%	UnivarCI_Lower	UnivarCI_Upper
1	1	c	V_F	14.563861	21264.661884	146009.78	-50268.775001	50297.902723
2	1	c	K01	0.062902	91.835226	145997.55	-217.094644	217.220447
3	1	c	K10	0.058889	90.658196	153946.91	-214.315399	214.433178
4	1	t	V_F	30.111986	3043.156234	10106.13	-7416.228443	7476.452414
5	1	t	K01	0.522895	52.479742	10036.39	-127.890501	128.936290
6	1	t	K10	0.532466	53.578722	10062.37	-130.570039	131.634971

Figure 2-2. Example modeling output: un-transposed Final Parameters.

4. Select the default NCA calculation method, to be used in computing area under the curve (AUC) in noncompartmental analyses, from among the methods below. See Section 3.7 of [Gabrielsson and Weiner \(2001\)](#), for detailed discussion of AUC computation methods.

- **Linear/log trapezoidal:** log-trapezoidal rule, using linear interpolation between log-transformed data points through Clast.
- **Linear trapezoidal (linear interpolation):** linear trapezoidal rule, using linear interpolation between untransformed data through Clast.
- **Linear up/log down:** linear interpolation between untransformed data up to Cmax, and between log-transformed data from Cmax through Clast.
- **Linear trapezoidal (linear/log interpolation):** trapezoidal rule with linear interpolation up to Cmax; log-linear interpolation between data points from Cmax, extrapolated to Tinf.

5. Click **Apply** to set these options and keep the dialog open or **OK** to keep these settings and close the [WinNonlin options](#) dialog.

Tables

To specify default table options:

1. Choose **Tools>Options** from the WinNonlin menus and open the **Tables** tab, or choose **File>Options** from the **Table Wizard** menus.
2. Use the check box to set the default for creating page breaks when the value of the group variable(s) changes.
3. To print group variables outside and above the table, rather than as a column in the table body, check **Place Page Break Data Outside Table Body**.
4. Click **Apply** to set these options and keep the dialog open or **OK** to keep these settings and close the [WinNonlin options](#) dialog.

Units

The display of units in workbooks, charts and tables is set on this tab of the [WinNonlin options](#) dialog. Units for specific output parameters can be changed for a modeling session (see “[Model options](#)” on page 205).

To specify default units display:

1. Choose **Tools>Options** from the WinNonlin menus and select the **Units** tab of the Options dialog.
2. Select the **Show Units with Column Headers** check box to include units in column headings of output workbooks.
3. Choose whether to have units displayed on chart labels and/or in tables. Note that these settings can be overridden in individual tables and charts.
4. Click **Apply** to set these options and keep the dialog open or **OK** to keep these settings and close the [WinNonlin options](#) dialog.

Licensing

By default, WinNonlin will generate a warning message upon launch if your license to use the software is within 30 days of expiration. The Licensing tab

of the [WinNonlin options](#) dialog provides a control to disable, delay or enable this warning message.

The Licensing tab also provides a means for floating license users to specify a preference to obtain a license for WinNonlin Enterprise edition, Professional, or the best available from the network license server. (See “[WinNonlin editions](#)” on page 1 for information about WinNonlin editions.)

To disable or enable license expiration warnings:

1. Choose **Tools>Options** from the WinNonlin menus and open the **License** tab of the Options dialog.
2. To allow WinNonlin to produce a warning message on launch within 30 days or license expiration, select **Enable** under Warnings.
3. To prevent WinNonlin from producing the license warning for a set number of days (beginning on the day this selection is made), select **Postpone for** under Warnings and select the appropriate number of days in the drop-down list.
4. To prevent WinNonlin from ever producing the warning, select **Disable**.
5. Click **OK** to apply your selection. The new setting will apply the next time you launch WinNonlin.

To set a preference for type of floating license:

1. Choose **Tools>Options** from the WinNonlin menus and open the **License** tab of the Options dialog.
2. Under Features, choose the license type that WinNonlin should request from your license server (see “[WinNonlin editions](#)” on page 1):
 - a. Best Available: WinNonlin will attempt to secure an Enterprise license, then, if no Enterprise license is available, a Professional license.
 - b. Enterprise: WinNonlin will only accept Enterprise edition licenses.
 - c. Professional: WinNonlin will accept only Professional edition licenses.
3. Click **OK** to apply your selection. The new setting will apply the next time you launch WinNonlin.

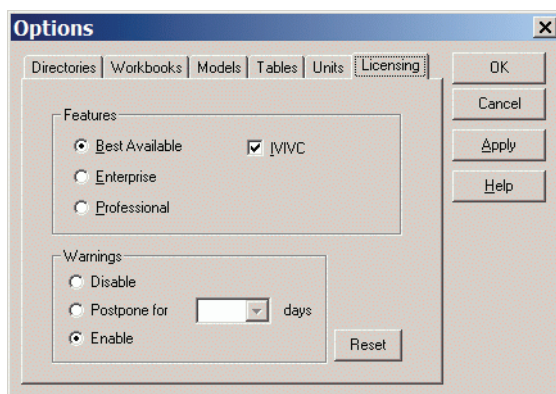
Installing an IVIVC license

Note: Installing an IVIVC License requires the same steps and listed descriptions as those for WinNonlin in the Getting Started Guide, pages 11-16.

1. To install a node license key code, follow the instructions in the WinNonlin Getting Started Guide under “Install a node license key code” on page 12.
2. To install a floating license key code, follow the instructions in the WinNonlin Getting Started Guide under “Install a floating license key code” on page 14.

After the license is installed, WinNonlin must be configured to recognize IVIVC.

1. Open WinNonlin.
2. Go to the main menu.
3. Click **Tools > Options > Licensing** tab.
4. **Check** the IVIVC Checkbox.
5. Click **OK**.



Data dependencies and origins

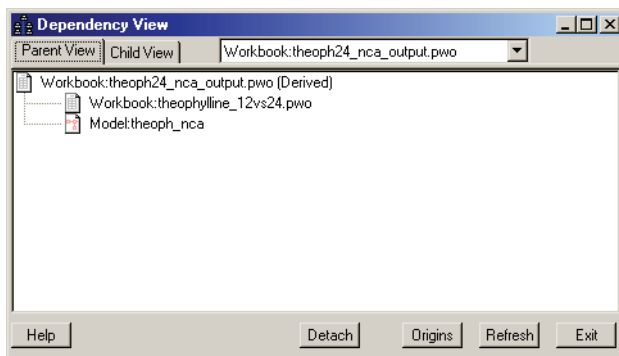
In order to protect the integrity of analysis results, WinNonlin locks derived data products—including modeling or analysis output in workbooks, text windows, and charts—against changes. If the source data changes, WinNonlin marks each derived product as out of date, by coloring it pink. You can refresh

the derived objects based on updated source data, or, if desired, make output independent of the source so that it is unlocked and editable.

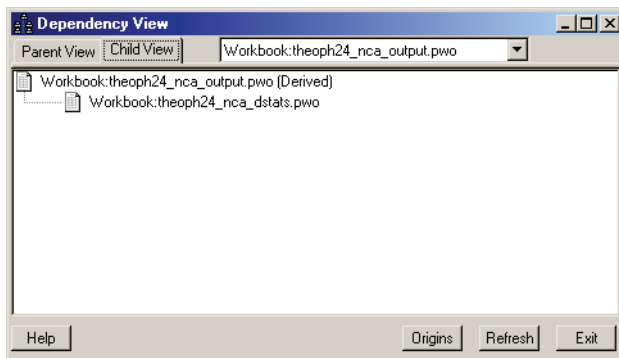
To view dependencies:

1. After an analysis, choose **File>Dependencies** from the WinNonlin menus.
2. Select an object from the drop-list at the top of the Dependency View dialog.

The Parent View (below) shows the items' source objects, if any, in a tree with source objects as branches. In this example, the noncompartmental analysis output THEOPH24_NCA_OUTPUT.PWO is dependent on its source workbook, THEOPH_12VS24.PWO, and the model THEOPH_NCA.PMO.



The Child View shows objects that are derived from a selected item. Its tree shows dependencies in reverse of the Parent View; with derived objects as branches. In the example below, the modeling output workbook THEOPH24_NCA_OUTPUT.PWO (top item) provided the source data for the descriptive statistics results in the workbook THEOPH24_NCA_DSTATS.PWO.



3. Click the **Origins** button to view pedigree information for derived objects.
See “Object origins” on page 36.
4. Click the **Refresh** button to view pedigree information for derived objects.
See “Refreshing results” on page 35.

Saving dependencies

Dependencies persist only while related objects remain open in WinNonlin, or when they are saved together in a workspace or PKS scenario. To save objects with dependency information, create the desired output, keep all objects open, and save the set to disk as a workspace (see “Workspaces” on page 19) or to the Pharsight Knowledgebase Server as a scenario (see “Using the Pharsight Knowledgebase Server” on page 449).

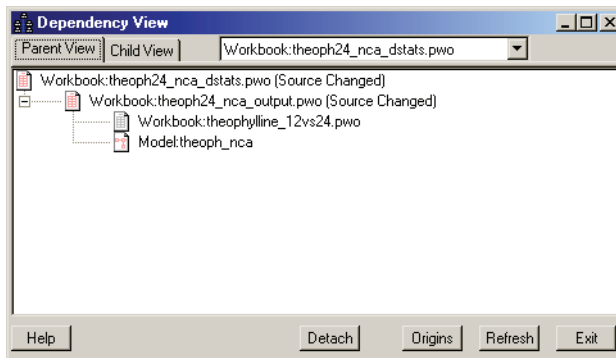
Refreshing results

If source data or analysis settings are modified while their derived objects (output) remain open in WinNonlin, the derived objects become out of date; each output window is shaded and its title bar includes: “Source Changed.”

Note: Opening and using the OK button to close any modeling or analysis dialog marks derived objects as out of date regardless of changes to analysis settings.

To update output to reflect changes to source data or analysis settings:

1. Open all input and output in WinNonlin, with dependencies intact (see “Data dependencies and origins” and “Saving dependencies” above).
2. Make the necessary edits to the source data and/or analysis settings.
3. Choose **File>Dependencies** from the WinNonlin menus.
4. Select an output item from the analysis to be refreshed at the top of the Dependency View dialog.



5. Click the **Refresh** button to re-generate all output from that analysis, and any derived objects that lie between the source data and the selected item.

In the example above, a change to data exclusions in THEOPH_12VS24.PWO deprecated the NCA chart and workbook output (THEOPH24_NCA_OUTPUT.PWO and THEOPH24_NCA_OUTPUT.PCO), and statistics on the NCA workbook (THEOPH24_NCA_DSTATS.PWO). Selecting either NCA output object for refresh would update both NCA objects but not the descriptive statistics. Selecting the descriptive statistics for refresh, as above, will update that output and the NCA chart and workbook on which it depends.

Unlocking derived data

Analysis and modeling output can be unlocked, making its data editable and removing object dependencies. Changes to the source data will no longer mark this derived work as out of date, and Refresh will no longer be available.

To unlock a derived product:

1. After running an analysis, choose **File>Dependencies** from the menus.
2. Select the item to unlock at the top of the Dependency View dialog.
3. Click the **Detach** button.

Object origins

The Origins button appears when an object that is dependent on any other object(s) is selected in the Dependencies View dialog. Click this button to view a pedigree for each child object open in WinNonlin, including:

- Object type and file or window name
- WinNonlin engine used to create the object, e.g., modeling or LinMix
- *Source ID*: identifier for any source data from which the object derives, with the worksheet and, if applicable, workbook name for source data
- For certain analyses, a report of analysis settings and variables.

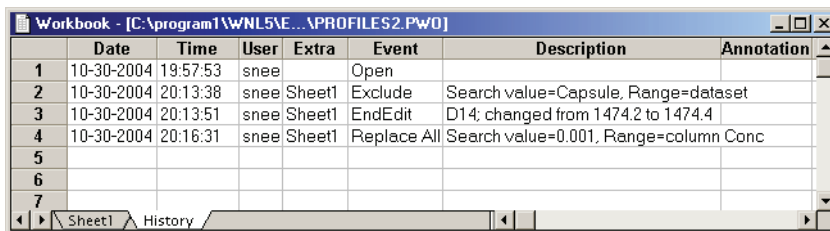
This pedigree information is saved as an XML dependencies file (*.DEP) when the source and derived objects are saved in a workspace. This file is always saved as part of a PKS scenario.

Data history worksheets and log files

WinNonlin provides two means to track data operations. Each WinNonlin workbook includes a *History* worksheet that logs most operations^A on data from that workbook. WinNonlin also includes a separate application, the WinNonlin Log Viewer, that provides access to a log file that tracks all data operations performed in WinNonlin. See “[WinNonlin Log Viewer](#)” on page 39.

History worksheet

The History worksheet logs operations made on the workbook data. It cannot be deleted, and only the Annotations column can be edited (enter notes here).



	Date	Time	User	Extra	Event	Description	Annotation
1	10-30-2004	19:57:53	snee		Open		
2	10-30-2004	20:13:38	snee	Sheet1	Exclude	Search value=Capsule, Range=dataset	
3	10-30-2004	20:13:51	snee	Sheet1	EndEdit	D14; changed from 1474.2 to 1474.4	
4	10-30-2004	20:16:31	snee	Sheet1	Replace All	Search value=0.001, Range=column Conc	
5							
6							
7							

The first line always notes creation of the workbook, and indicates the number of worksheets. The sheet then logs edits to the workbook, including application of units, all data manipulations, including sorts and transformations, PKS operations, and the analysis operations listed in [Table 2-7](#).

A. Two known cases that are not captured in the History page are filling a range of cells and moving a range of cells.

Table 2-7. Analysis operations logged in the History worksheet

Operation	Items logged
Deconvolution	<ul style="list-style-type: none"> • Source data workbook and worksheet • Missing/excluded values • Notification of any failed or invalid profiles • Success or failure of the deconvolution operation
Descriptive Statistics	<ul style="list-style-type: none"> • Source data workbook and worksheet • Missing/excluded values • For weighted analyses, notification of weights < 0 (all values for that profile will be missing) • Notification of any floating point error, and note that all values for that profile set to missing • Success or failure of the LinMix/BioEq operation
LinMix and Bioequivalence (BE)	<ul style="list-style-type: none"> • Source data workbook and worksheet • Missing/excluded values • In BE, if a least square mean is <0, notification that some calculation(s) couldn't be made • If Ln transformation included, notification if any value(s) of the dependent variable <= 0 • Notification of any regressor/covariate found to be alphanumeric • Warnings or errors from the analysis, and the profile number at which each occurred • If warnings generated, but no errors: "warnings generated on one or more profiles of data" • Success or failure of the LinMix/BioEq operation
Merge	<ul style="list-style-type: none"> • Source data workbook and worksheet #1 • Missing/excluded values from source worksheet #1 • Source data workbook and worksheet #2 • Missing/excluded values from source worksheet #2 • Sort variable used from each worksheet • Include variables from each worksheet • Success or failure of the Merge operation
PK and Non-compartmental analyses (NCA)	<ul style="list-style-type: none"> • Source data workbook and worksheet • Missing/excluded values • Profiles with less than 2 usable records noted: "cannot be processed" • For urine NCA, notification of any profile with all zero values for volume or concentration • For NCA, if only 1 non-zero concentration is found for any profile, and it occurs at time=0, notification that all parameters set to Missing • For NCA, lambda_z exclusions (or the lack of any) for each profile • Modeling halted using Stop button • Success or failure of the modeling process
Toolbox tools	<ul style="list-style-type: none"> • Source data workbook and worksheet • Success or failure of the toolbox engine operation

WinNonlin Log Viewer

WinNonlin has a separate application, the Log Viewer, for reviewing a record of operations performed in the application. WinNonlin writes this information to a separate log file. When the log file reaches its size limit, the file is purged to a backup file. This log file size can be specified or changed using View>Options in the Log Viewer application menus.

To run the WinNonlin Log Viewer:

1. From the Windows **Start** menu, choose **Programs>Pharsight>WinNonlin 5.3>Log Viewer**. The application log contains entries for the following.

• WinNonlin startup and finish	• PK/PD modeling errors
• Open/save workbook	• Cut/Copy/Paste operations
• Replace/Remove data (replace/remove parameters, not cell-by-cell values)	• Include/Exclude data operations (include/exclude parameters, not cell values)
• Changing WinNonlin registry settings	• Insert column/row
• Transform operations	• Delete column/row
• Specifying column units	• Undo operations
• Clearing units	• Units conversions, including messages of success/failure

Workbooks

Data editing, formatting, and manipulation in WinNonlin worksheets

Input and output data for WinNonlin analyses are stored and manipulated in Excel-compatible worksheets, shown as individual tabs within a workbook window. Data can be typed or pasted into a new worksheet, or imported (see [“Data import and export” on page 75](#)).

Every workbook contains at minimum one data worksheet and a History worksheet, which details many of the changes and operations made on the data (see [“Data history worksheets and log files” on page 37](#)).

Worksheet operations are covered in the following sections.

- [“Editing data” on page 41](#): editing and formatting worksheet data using formulas, functions, copy and paste and fill-down.
- [“Data manipulation” on page 56](#): sorting, excluding and including data, merging workbooks, and transforming non-numeric values.
- [“Transformations” on page 62](#): creating new workbook columns as functions of other columns.
- [“Status code transformations” on page 69](#): replacing non-numeric codes with specified values.

Editing data

WinNonlin worksheets support spreadsheet-like data entry and formatting. Numerical and text values may be entered directly into worksheet cells, and cut, copied, and/or pasted via the Windows Clipboard. Worksheet editing operations are detailed in the following sections.

- [“Inserting or deleting worksheets” on page 42](#)

- “Inserting and deleting rows, columns or cells” on page 43
- “Using formulas and functions” on page 44
- “Filling adjacent cells” on page 46
- “Formatting cells” on page 46
- “Clearing cell values and formats” on page 49
- “Undo” on page 50
- “Find, find next and replace” on page 50
- “Go to” on page 52

Inserting or deleting worksheets

Every workbook must contain at least two sheets: a data worksheet and a worksheet labeled Data History (see [“Data history worksheets and log files” on page 37](#)). These worksheets cannot be deleted. Other worksheets can be added and deleted.

To add a new worksheet:

1. Open the workbook to the insertion point for the new worksheet. The new worksheet will be inserted in front of the currently-selected worksheet.
2. Choose **Edit>Insert Sheet** from the WinNonlin menus. A new blank worksheet is inserted on top of the current worksheet.

To name a worksheet:

1. Select the worksheet to be named.
2. Double click on the tab for the sheet. The Sheet Name dialog appears.
3. Enter the name to be displayed on the tab and click **OK**.

To delete a worksheet:

1. Open the worksheet to be deleted.
2. Choose **Edit>Delete Sheet** from the WinNonlin menus.
3. Click **Yes** to confirm deletion of the current worksheet.

Inserting and deleting rows, columns or cells

To insert a row or column:

1. Highlight a row or column by clicking on the row or column label.
2. Select **Insert** from the **Edit** menu (**Alt+E, I**) or right-click and select **Insert** from the shortcut menu.

or

1. Highlight a cell within the row.
2. Select **Insert** from the **Edit** menu (**Alt+E, I**).
3. In the Insert dialog that appears, select **Entire Row** or **Entire Column**.

A row or column will be inserted above or left of the active row or column.

Note: To insert multiple row or columns quickly, select the same number of rows or columns as the number to be inserted, then choose **Edit>Insert**.

To insert a cell:

1. Select a cell with the mouse.
2. Select **Insert** from the **Edit** menu (**Alt+E, I**) or right-click the selection and choose **Insert** from the shortcut menu.
3. In the dialog that appears, select either **Shift Cells Right** or **Shift Cells Down**. A single cell is inserted.

To delete cells:

1. Select the cell(s) to be deleted.
2. Select **Delete** from the **Edit** menu (**Alt+E, D**) or right-click the selection and choose **Delete** from the shortcut menu.
3. From the Delete dialog, select how to move the surrounding cells.

To delete rows or columns:

1. Select the row(s) or column(s) to be deleted by highlighting the row or column label(s).
2. Select **Delete** from the **Edit** menu (**Alt+E, D**) or right-click the selection and choose **Delete** from the shortcut menu.

or

1. Select a cell in each row or column to be deleted.
2. Select **Delete** from the **Edit** menu (**Alt+E, D**).
3. From the Delete dialog, select **Entire Row** or **Entire Column**.

Using formulas and functions

Worksheet values may be entered as mathematical formulas, by beginning the cell's value with the equals sign, "=". This is useful for creating new columns as a function of other columns. See [Table 3-1](#) for a list of valid operators.

WinNonlin worksheets also support an extensive list of functions. A complete listing and syntax are provided in "[Worksheet Functions](#)" on page 555.

Cell references

Formulas and functions refer to worksheet cells by location in the worksheet grid, using row number and column letter. For example, A3 denotes the third cell in the first column. To identify a range of cells, enter the top-left and bottom-right cells, separated by a colon. For example, A1:B4 calls the first four cells in the first two columns (starting from the top left cell in the worksheet).

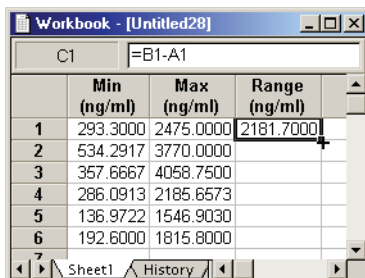
A "\$" makes a reference position absolute, rather than relative to the currently-selected cell, for copy and paste operations. For example, to create a new column of values equal to the difference between values in columns B and A, you could set cell C1 = B1-A1, then use the fill-down feature to paste this formula to all cells in column C. Cell C2 would equal B2-A2; C3 = B3-A3, etc. In contrast, to create a variable equal to the value in column B minus the value of cell A1, you would cell C1 = B1-\$A1. Pasting this formula down column C would cause C1 = B1-\$A1; C2 = B2-\$A1; C3 = B3-\$A1, etc.

Table 3-1. Workbook operators

Operator	Description
+	Addition
-	Subtraction
/	Division
*	Multiplication
^	Exponentiation
%	Percentage
=	Equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
<>	Not equal to

To create a new variable using a formula:

1. Using the mouse, select the first cell in the column of the new variable.
2. In the edit bar at the top of the worksheet, type the formula, including the operator and an expression representing the value, e.g., =SQRT (A1) or =A1+5. To enter cell references, either type them or click on the desired cell (drag to select multiple cells) to insert the reference into the formula.
3. Press the **Enter** key to enter the formula into the cell.
4. Select the cell and position the cursor over the small black square at the lower right hand corner of this cell until the cursor changes to a small black cross.



5. Press and hold the mouse button, and drag the cursor down the column to copy the formula.

	Min (ng/ml)	Max (ng/ml)	Range (ng/ml)
1	293.3000	2475.0000	2181.7000
2	534.2917	3770.0000	3235.7084
3	357.6667	4058.7500	3701.0834
4	286.0913	2185.6573	1899.5660
5	136.9722	1546.9030	1409.9308
6	192.6000	1815.8000	1623.2000

Filling adjacent cells

A value or formula from one cell can be filled to adjacent cells up, down, or to the left or right.

To copy to adjacent cells in a selected block of cells:

1. Select the cells to be filled, including the cells to be copied as the left-most cells in the selection.
2. Select **Edit>Fill>Down**, **Edit>Fill>Up**, **Edit>Fill>Right** or **Edit>Fill>Left** from the menus.

To copy to adjacent cells using the mouse:

1. If necessary, enter the value or formula to copy, and press the **Enter** key.
2. Select the cell to copy.
3. Position the cursor over the small black square at the lower right hand corner of this cell until the cursor changes to a small black cross.
4. Press and hold the mouse button, and drag the cursor down and/or to the right over the target cells to copy into them.

Formatting cells

Choose **Data>Format** (or click the Format tool bar button with a workbook in the active window) to apply formatting to the currently-selected worksheet cell(s). The Format Cells dialog has five tabs for specifying the following.

- Number format
- Alignment
- Font face and format
- Borders
- Fill color and pattern

Note: Data formatting is not saved in ASCII (text) files.

To set the format for a cell or selection of cells:

1. Select the cells to format. To select whole rows or columns, click on the row or column header(s).
2. Select **Data>Format** from the WinNonlin menus or click the **Format** tool bar button. The Format Cells dialog appears.
3. Choose format settings from the tabs detailed below.
4. Click **OK** to apply the changes or **Cancel** to close the dialog without changes.



Number format

The Number tab of the Format Cells dialog sets the handling of numbers, but not non-numeric entries, in the selected cells.

Category: Select a number category or All to show all types.

Type: Select a format here. The Sample field will display an example. Syntax in the Type box is as follows.

- **0** (zero) represents the a digit or exponent and decimal places, if any.
- **#** represents digits to the thousands, and indicates whether thousands are separated by commas.
- Symbols appearing after a semicolon indicate handling of negative numbers.
- An underscore “_” at the beginning of symbols indicates left-alignment of left parentheses. An underscore “_” at the end inserts a space, to align positive values with negative values enclosed in parentheses.

- ***** indicates that zero values will be replaced with a dash.
- **h** represents hours on a 24-hour clock.
- **m** represents minutes for times or months for dates.
- **s** represents seconds.
- **d** represents days.
- **y** represents years.
- **??** indicates display as fractions rather than decimals.
- **E** indicates scientific notation.

Note: Choose Category = All and Type = General to display entries as they are typed in, but using scientific notation for numbers \geq ten billion.

Alignment

The Alignment tab of the Format Cells dialog sets alignment and line wrap to the selected cells, and merges cells.

Horizontal: alignment from left to right can take the following values:

- General: left-aligns text and right-aligns numbers.
- Left: left-aligns cell content.
- Center: centers cell contents.
- Right: right-aligns cell contents.
- Fill: repeats cell contents enough time to fill the width of the cell.
- Center across cells: centers content in the left-most cell across a multi-column selection, assuming adjacent cells to the right are empty.

Vertical: aligns content at the top, center, or bottom of each cell.

Wrap Text: when text content is longer than the cell width allows, wraps text onto multiple lines to fit the cell width.

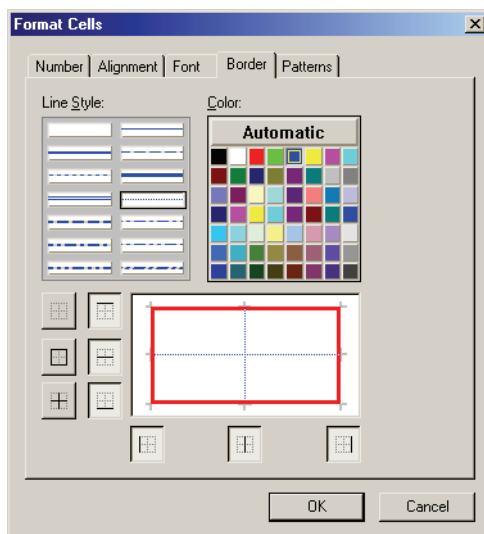
Merge Cells: joins selected cells into a single cell that spans multiple columns and/or rows.

Font face and format

Use this tab to assign a font face, color and format to the selected cell or cells.

Borders

Use this tab to create borders around the selected cell(s). First, select a line style and color. Next, use the buttons or click between the marks in the diagram below to indicate border locations. The example below will put a thick, red line around the outermost edges of the block of selected cells, and thin, blue lines between cells within the selection.



Fill color and pattern

Use this tab to set background color and pattern for the selected cell or cells.

Clearing cell values and formats

When cells are cleared, the values and/or formats are deleted, but the cells remain in the grid. To delete a cell from the grid, see [“Inserting and deleting rows, columns or cells”](#) on page 43.

To clear data values, but not formatting, from cells:

1. Select the cells to be cleared.

2. Choose **Edit>Clear>Values** from the menus. Alternately, right-click in the worksheet and select **Clear Values** from the shortcut menu, or press the **Delete** key.

To clear formatting from cells:

1. Select the cells to be cleared.
2. Select **Edit>Clear>Formats** from the menus.

To clear data values and formats from cells:

1. Select the cells to be cleared.
2. Select **Edit>Clear>All** from the menus.

This will delete the information from the cells, but will not remove the cells.

Undo

The Undo capability allows the single, most-recent action to be reversed, when editing workbooks, charts or text windows.

To undo an operation:

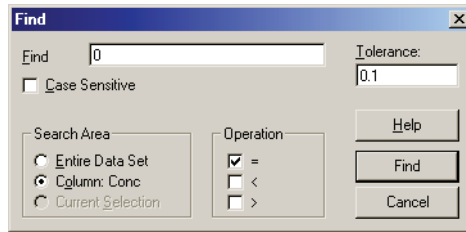
1. Select **Undo** from the **Edit** menu (**Alt+E, U**). Alternately, use the keyboard shortcut **Ctrl+Z** or right click in the window and select **Undo** from the shortcut menu.

When an operation has been performed that can be undone, Undo will appear as a normal selection on the Edit and shortcut menus. The Undo command is disabled if the most-recent action cannot be undone.

Find, find next and replace

To locate specific data values within a worksheet:

1. Choose **Edit>Find** from the WinNonlin menus (**Alt+e, f** or **Ctrl+f**).
2. Enter a numeric or text search string in the box labeled Find.
3. Select the appropriate options, below. The following example will find concentration values from -0.1 to 0.1.



Case Sensitive

Checking this box causes only occurrences with the exact combination of uppercase and lowercase letters to be found.

Search Area

Allows searching over the entire data set, the currently-selected column or a selection of cells. In this example, the column Conc is one potential search area because the a cell in that column was selected before opening this dialog.

Operation

This option searches for a value:

- Equal to the search string (=),
- Less than the search string (<),
- Less than or equal to the search string (<=),
- Greater than the search string (>),
- Greater than or equal to the search string (>=), or
- Not equal to the search string (<>).

Tolerance

Finds values not different from the (numeric) search string by more than the tolerance (optional).

4. Click **Find**.

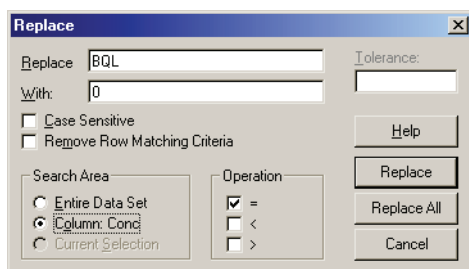
To repeat the designated search:

1. Select **Find Next** from the **Edit** menu (Alt+E, N).

To replace text or data in a grid:

1. Select **Replace** from the **Edit** menu (Alt+E, E). The Replace dialog appears.

2. Enter a numeric or text search string in the box labeled Replace.
3. Enter a numeric or text replacement string in the box labeled With.



4. Select the appropriate options. The example above will replace all concentrations with a value of “BQL” with zero. Note that, since the Case Sensitive option is checked, values of “bql” will not be replaced.

Options are the same as for Find, above, with one addition:

Remove Row Matching Criteria

Check this box to remove any row that contains the search string within the search area.

5. Click **Replace** to review each occurrence before replacement or click **Replace All** to replace all occurrences without reviewing each.

Go to

The Go to function jumps to a specific cell in any worksheet within the current workbook.

To move to a given cell using a cell reference:

1. Choose **Data>Goto** from the menus (**Alt+D, G**). The Goto dialog appears.
2. Select any worksheet in the active workbook under Sheet.
3. Enter the grid location of the cell. For example, for the third cell (3) in the first column (A), enter: A3.
4. Click **Goto**.

Column names and units

The Column Properties dialog sets column headers and units for the current worksheet or, for multiple transformations, for the new worksheet that will contain the transformed data (see “[Multiple transformation](#)” on page 64).

To specify column headers:



1. Highlight a cell in the column to be named.
2. Click the **Column Properties** tool bar button or double click a column header to open the Column Properties dialog.
3. Highlight the column to be named.
4. Click the **Edit Header** button.
5. Type the name. Note that long column names can affect performance.
6. Press **Enter** or click **Close**. The new column header appears in the worksheet.

Note: WinNonlin does not allow spaces in column headers. If a data set is loaded that has column names with spaces WinNonlin will ask to substitute an underscore (`_`) for the spaces (and offer the opportunity to save the data set to a different name); otherwise, the data cannot be used in analyses and models.

Each column in a worksheet can use different units. The units appear in the column header, in parentheses, and will be used in data calculations. If the units are not standard or recognizable, WinNonlin will display them in braces: { `furlongs` }. Nonstandard units—units enclosed in braces—will be carried throughout any operations, such as descriptive statistics, but not used in any calculations. See “[WinNonlin options](#)” on page 28 for units display and default options. Permissible units include the following.

Table 3-2. Unit types

Unit type	Unit	Abbreviation
Time	day	day
	hour	hr
	minute	min
	second	s
	millisecond	ms
	microsecond	us
	nanosecond	ns
Mass	gram	g
	mole	mol
	pound	lb
	IU	IU
Volume	Liter	L

Prefixes can be used for units of mass and volume.

Table 3-3. Prefixes

Prefix	Abbreviation		Prefix	Abbreviation
femto	f		milli	m
pico	p		centi	c
nano	n		deci	d
micro	u		deca	dk
			kilo	k

Note: Units are not case-sensitive.

To enter units in the Column Properties dialog:

1. Select the column name in the column header field.
2. Enter the units for that column in the New Units field.

3. Click the **Specify Units** button.
4. Click **Close** to close the dialog.

The Units Builder assists with units selection. See “[Units builder](#)” on page 55.

Note: Worksheet units can be saved only in Pharsight Workbook Objects (*.PWO). Other file types lose unit assignments when the file is closed.

To clear the units associated with a column:

1. Select a cell in the column to operate on.
2. Click the **Column Properties** tool bar button. The Column Properties dialog appears with the column selected. The current units appear in the list box.
3. Click the **Clear Units** button.
4. Click **Close** to close the dialog.

Converting units

Units can be converted automatically using the Column Properties dialog.

To convert units:

1. Select a cell in the column to be converted.
2. Click the **Column Properties** tool bar button. The Column Properties dialog appears with the column selected. The current units appears in the list box.
3. Enter the new units in the **Specify Units** field.
4. Click the **Convert Units** button. The data in the column are converted to use the new units.

Units builder

Units are edited in either the Column Properties dialog or the Units Builder.

To use the Units Builder:



1. With the data open in WinNonlin, click the **Column Properties** tool bar button or select **Data>Column Properties** from the WinNonlin menus.
2. In the Column Properties dialog click the **Units Builder** button. The Units Builder dialog appears.
3. Select the kind of unit (Time, Mass, or Volume) and the appropriate unit.
4. Prefixes (kilo, milli, etc.) can be selected from the **Prefix** field.
5. Once the unit (with appropriate prefix) displays correctly click the **Add** button to the right of that kind of unit. The appropriate abbreviation displays in the New Units field.
6. Click **OK** to close the Units Builder.
7. Click the **Specify Units** button to associate the units with the data column.
8. Click **Close** to close the Column Properties dialog.
9. To build units with operators (grams/ml, for example), enter the units and operators in order.

Data manipulation

WinNonlin worksheet operations include the following:

- “Freezing rows and columns” on page 56
- “Sorting” on page 57
- “Merge” on page 57
- “Data exclusion and inclusion” on page 60

Freezing rows and columns

Freezing one or more rows and/or columns creates a non-scrolling region that remains on the screen while you scroll through the remaining data.

To freeze a row or column:

1. Select the column or row, or one cell at the outer-most row/column to freeze.

2. Choose **Window>Freeze Panes** from the WinNonlin menus.

To unfreeze rows and/or columns:

1. Choose **Window>Unfreeze Panes** from the WinNonlin menus.

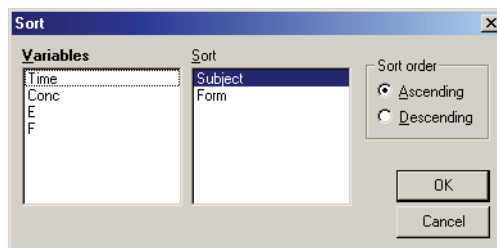
Sorting

Sorting provides a quick means of ordering worksheet data.

To sort the data in a worksheet:



1. Open the worksheet and choose **Tools>Sort** from the WinNonlin menus or click the **Sort** button on the tool bar. The Sort dialog appears.
2. Select a variable name and drag it to the **Sort** box, or highlight the variable then type the letter **S** while holding down the **SHIFT** key.
3. Repeat for any other sort variables.
4. Select the sort order (**Ascending** or **Descending**) and click **OK**.



If the data have been sorted previously, the sort variables are listed here.

Merge

The Merge command joins selected data from any two worksheets into a new workbook.

Note: Exclusions are ignored when merging data sets.

To merge data sets:

1. Open the two data sets to merge in one or two workbook windows.

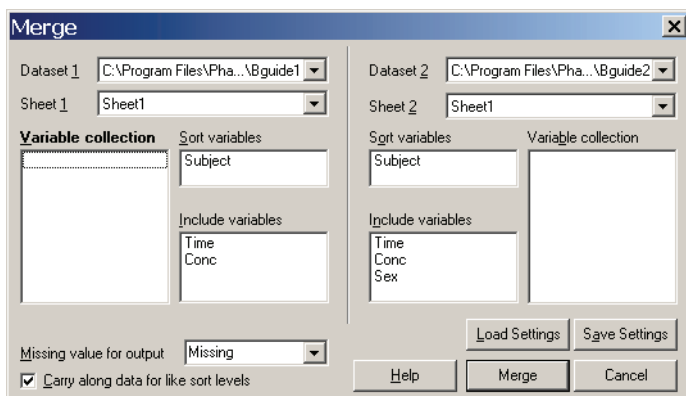
2. Select **Tools>Merge** from the WinNonlin menus.

The Merge dialog appears. It is divided into two areas, one per data set.

3. Select the workbook containing each data set from the pull-down menus for Dataset 1 and Dataset 2.
4. Select the appropriate worksheet for each data set under Sheet1 and Sheet2.

The merged data records will be matched on values of the sort variables, selected below. If the names differ, the name from data set 1 will be used as the column header in the merged data.

5. For data set 1, drag a variable on which to join records from the data set 1 variable collection to its sort variables list. Drag the corresponding variable for data set 2 from its variables list to its sort variables list, on the right side of the dialog. If a secondary sort variable is needed, repeat. Be sure that the corresponding variables appear in the same order for each data set.



6. For each data set, drag variables to be copied to the output workbook from the Variable Collection box to the Include Variables box.

These variables will appear in individual columns of the merged data set. They need not match in data sets 1 and 2.

7. Check **Carry along data for similar sort keys** if you wish to fill-in empty cells for given sort level with the last-available value. This applies in the case that one data set has more observations than the other for a given sort level. An example is provided below.
8. Click **Merge**. The merged data set appears in a new workbook.

The settings in this dialog can be saved and re-loaded for later use by way of the Load and Save buttons. Note that the two datasets must be selected before loading settings, and data set 1 or 2 must contain the variables used in the file.

Carry along data for like sort levels

The following example shows the effect of selecting to carry along data for like sort levels when merging data. Subject and Time are sort variables; the other two are Include Variables.

Table 3-4. Data sets to merge

Data set 1			Data set 2		
Subject	Time	Conc	Subj	Time	Effect
A	0	2.98	A	0	0.35
A	5	1.27	A	5	0.1
A	10	0.76	A	5	0.95

Table 3-5. Merged data set with carry along for like sort levels

Subject	Time	Conc	Effect
A	0	2.98	0.35
A	5	1.27	0.1
A	5	1.27	0.95
A	10	0.76	

Table 3-6. Merged data set without carry along for like sort levels

Subject	Time	Conc	Effect
A	0	2.98	0.35
A	5	1.27	0.1
A	5		0.95
A	10	0.76	

Data exclusion and inclusion

Specific cells, rows and/or columns in a worksheet can be excluded from analyses. For example, if only a single subject must be refit from a dataset holding many subjects, you can exclude all but the desired subject and refit the data. The data selections are made using the Exclude and Include commands on the Data menu.

Note: Excluded cells are treated as missing data in analyses. Exclusions are ignored in merges and transformations; all data are merged or transformed.

Excluding and including a selected range of cells

To exclude a selected range of cells:

1. Select the cells using the mouse. Select entire row(s) or column(s) by highlighting the row or column label(s).
2. Select **Data>Exclude>Selection** from the WinNonlin menus (**Alt+D, E, S**). The selected region is highlighted in red to indicate that it has been excluded.

To re-include a selected range of cells:

1. Select excluded cells to be re-included using the mouse. Select entire row(s) or column(s) by highlighting the row or column label(s).
2. Select **Data>Include>Selection** from the WinNonlin menus **Selection** (**Alt+D, I, S**). The selected region now is no longer marked in red.

Excluding and including data based on criteria

To exclude data:

1. Select **Data>Exclude>Criteria...** from the WinNonlin menus (**Alt+D, E, C**). The Exclude dialog appears.
2. Enter the value(s) to search for in the **Find** field.
3. Select any desired options from those detailed below.

Search Area

Allows searching over the entire data set, a single column, or the selected cell(s). In the example below, the column Time is shown as a potential search area because Time was the active column when the dialog was opened.

Exclude Row Matching Criteria

To exclude cells containing the search value, un-check Exclude Entire Row Matching Criteria. To exclude each row containing the search value in the search area, check this option.

Case Sensitive

Checking this box causes only occurrences with the exact combination of uppercase and lowercase letters to be found.

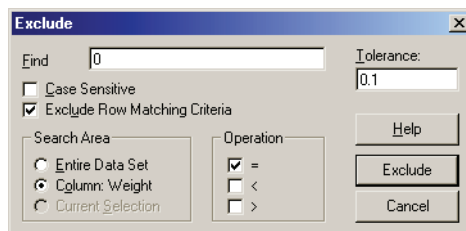
Criteria

Using this option search for a value:

- equal to the search string (=),
- less than the search string (<),
- less than or equal to the search string (<=),
- greater than the search string (>),
- greater than or equal to the search string (>=), or
- not equal to the search string (<>).

Tolerance

A value not different from the search string by more than the tolerance will be treated as a match (applicable only when searching for numeric data).



4. Click **OK**.

Re-including data

To re-include specific data:

1. Select **Data>Include>Criteria** from the WinNonlin menus (**Alt+D, I, C**).
2. Select the inclusion criteria from those described for exclusions under “[Data exclusion and inclusion](#)” on page 60, then click **OK**.

Resetting selections

Resetting the data selections clears all exclusions in the active worksheet. Other worksheets in the same workbook will not be affected.

To remove exclusions from the current worksheet:

1. Select **Data>Reset Selections** (**Alt+D, N**).

Transformations

A transformation creates a new column in the current worksheet, or in the destination worksheet, for multiple transformations (see “[Multiple transformation](#)” on page 64). All data in the source column are transformed regardless of exclusions, which are carried into the new column.

WinNonlin worksheets support the following transformation types:

Natural log	$X \cdot Y$	X^n
Log base 10	X/Y	$X*n$
Square root	$X + Y$	X/n
Absolute value	$X - Y$	$X + n$
Change from baseline	$X * Y$	$X - n$
Percent change from baseline	e^X	
Ratio versus baseline	$1/X$	

where X and Y are columns in the current worksheet and n is a number.

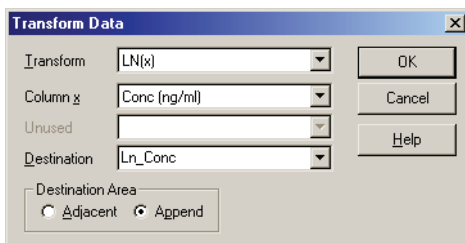
In baseline transformations, WinNonlin will use the value corresponding to Time = 0 as baseline for each profile. Individual profiles are identified using one or more sort variables. If a profile has more than one value at time 0, WinNonlin will use the first as baseline. See the examples below.

To transform a column, creating a new column in the current worksheet:



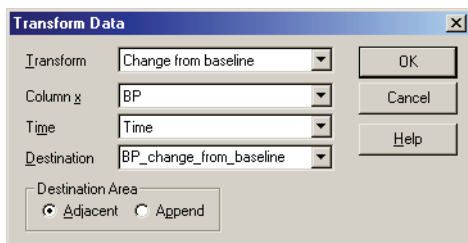
1. Open the data and choose **Tools>Transform** from the WinNonlin menus or click the **Transform** tool bar button to open the Transform Data dialog.
2. Select the desired transformation from the pull-down menu for Transform.
3. Select the variables to be used from the pull-down menus for Column x, Time and Column y, as needed for the transformation type.
4. If required, enter a value for n in the box labeled n.
5. Replace [New] with a column name for the new variable.
6. Select the position for the new variable in the worksheet:
 - **Adjacent** places the new variable to the right of the 'Column x' variable.
 - **Append** places the new variable after the last variable in the data set.

The example below creates a new variable called Ln_Conc, and places it in a new column at the right of the existing worksheet columns.

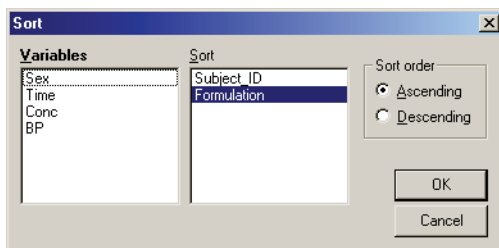


7. For baseline transformations, WinNonlin queries whether to specify sort variables. Sort variables define unique profiles, to compute change from baseline separately for each profile. Click **Yes** to choose sort variables, or **No** to calculate change from a single baseline value for the data set.
8. If sort variables are included: In the Sort dialog, drag variables that identify individual profiles to the sort variables list. WinNonlin will seek a baseline value (at time zero) for each unique combination of sort variable values. It will also sort the worksheet on the sort variables, in the order presented here. In most cases, a single variable representing subject IDs suffices. Other data may require more than one variable, such as Subject and Period.

The example below creates a new column to the right of the BP column.



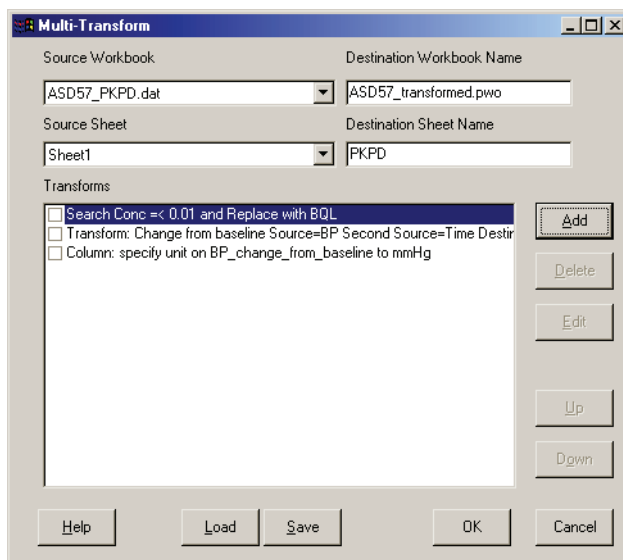
This example is from a crossover study. The transform will seek a baseline value at time zero for each unique combination of values for Subject ID and Formulation.



9. Click **OK** to complete the transformation and create the new variable.

Multiple transformation

The multiple transformation feature creates a copy of a source worksheet with specific data operations applied. The results appear in a new workbook. Supported operations include transformation of multiple columns, as well as search/replace/delete, data inclusion/exclusion, custom equations and rank transformations. You may specify column headers and units for the new worksheet as part of the multi-transform settings.



All operations listed under Transforms in the Multi-Transform dialog will be applied to a copy of the source worksheet data.

To copy and transform data in an existing worksheet:

1. Open the data to transform in WinNonlin and choose **Tools>Multi-Transform...** from the WinNonlin menus.
2. Select the workbook and worksheet containing the data to transform under Source Workbook and Source Sheet in the Multi-Transform dialog.
3. Enter a name for the new workbook and worksheet, which will contain the results, under Destination Workbook Name and Destination Sheet Name.

Note: Use the Save button to capture all settings in XML format for later re-use.

4. To set up transformations, click the **Add** button and follow the instructions for the [Transform action](#) dialog.
5. After adding transformations, use **Add>Column Properties** to set the column names and/or units for new columns.
6. To delete one or more transformation operations, check the box next to the transform(s), then click the **Delete** button.

7. To change an operation, check the box next to that transformation (and no others) and click the **Edit** button.
8. Click **OK** to create the new workbook.

Transform action

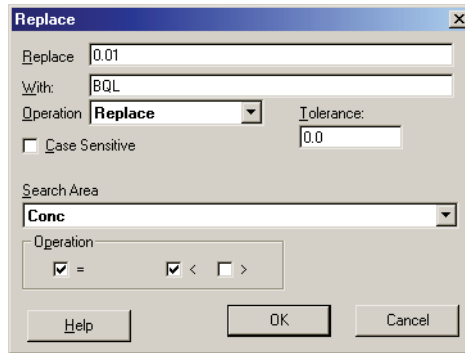
To add an operation to a [Multiple transformation](#), select one of the following actions in the Transform Action dialog and click **OK** to proceed.

Table 3-7. Transform actions

Action	Description	See:
Add Filter	Searches the source data for a specified value and includes, excludes, removes or replaces it in the destination sheet.	“Multi-transformation filter” on page 66
Add Equation	Creates a new column in the destination worksheet, using a formula or function.	“Multi-transformation equation manager” on page 67
Column Properties	Sets the column headings and units for the destination worksheet.	“Column names and units” on page 53
Rank	Creates a new column in the destination sheet as a rank transformation of a source worksheet column.	“Rank transformations” on page 68
Transform	Creates a new column in the destination sheet as a transformation of a source worksheet column.	“Transformations” on page 62

Multi-transformation filter

This dialog applies a search/replace, include/exclude or search/remove operation to the output of a [Multiple transformation](#):



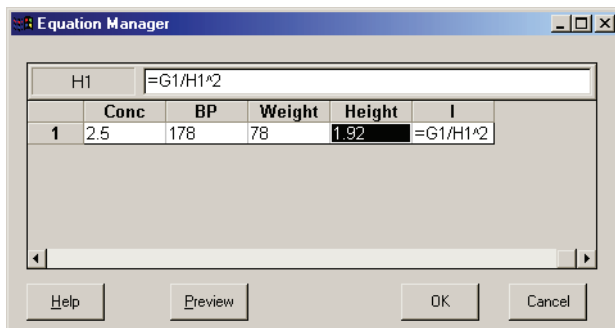
To filter data values to appear in the destination worksheet:

1. Select the desired action under Operation:
 - **Replace:** to find a value (or range) and replace it with another.
 - **Include/Exclude:** to (re-)include or exclude specified values (or rows containing them) in the destination worksheet. See also “[Data exclusion and inclusion](#)” on page 60.
 - **Remove:** to delete rows containing specified values.
2. Enter the value or string to search for in the topmost field. Check **Case Sensitive** to search for strings that match not only letters but also capitalization.
3. For Replace operations, enter the value with which to replace the search string under With.
4. To search for values within a range of a numeric search value, enter a value under Tolerance. Numeric values that are +/- tolerance of the search value will be treated as matching the search value.
5. Select the Search Area: the whole source worksheet or a specific column.
6. Select the appropriate operation(s) to find values equal to and/or less or greater than the search value.
7. Click **OK** to add the operation to the [Multiple transformation](#).

Multi-transformation equation manager

This dialog creates a new column in the destination worksheet of a [Multiple transformation](#) using a formula or function of columns in the source work-

sheet. Set the column name and units separately, using Add>Column Properties from the Multi-Transform dialog.

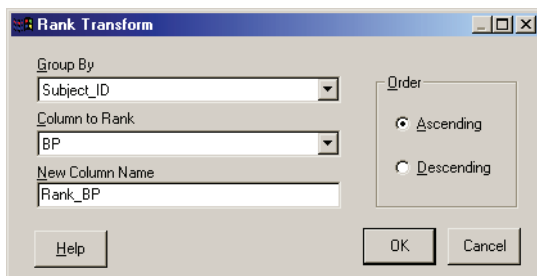


Enter the formula or function in the field at the top of the dialog. See:

- “Using formulas and functions” on page 44 for instructions on entering formulas and functions, and
- “Worksheet Functions” on page 555 for functions and syntax.

Rank transformations

This dialog creates a new column in the destination worksheet of a [Multiple transformation](#) as a rank listing of a column from the source worksheet.



Group By: WinNonlin will generate a separate set of ranking values for each unique value of the Group By variable.

Order: Ascending order ranks the lowest value as 1, and increments by one for each larger value. Descending order ranks the highest value as 1 and increments by one for each smaller value.

Status code transformations

Data analysis, tabulation, plotting, and summarization are problematic when the data contain concentrations that fall below the lower limit of quantification (*LLOQ*) of the assay. Standard laboratory procedures may require that such data be reported as a character string rather than a numerical value. Representing such a concentration as 0 (zero) is not always appropriate, as doing so can introduce a statistical bias or misrepresentation in some analyses. Different substitution rules are required depending on the intended use of the data, and substitution rules vary between companies and departments. Some common substitution rules for *BQL* samples are presented in the table below.

Table 3-8. Common status code substitution rules

Use data for:	Non-numeric code	Unconditional substitution	Conditional substitution			
			Before Tmax	After Tmax	First of consecutive after Tmax	After first of consecutive after Tmax
PK analysis	BQL		0	Missing	LLOQ/2	Missing
Summary statistics	BQL	0				
Listing of individual data	BQL	"BQL				"
Plotting of individual data on log-scale	BQL	LLOQ/2				
Plotting of individual data on linear scale	BQL	0				

The substitution rules are typically defined in standard operating procedures or method sheets for a given company or department. The rules may become more complex when one wishes to make a distinction between concentrations that are below the quantification limit (BQL) and below the detection limit (BDL). In general, a substitution rule is defined by the list of possible non-numeric representations for concentration and the values to be substituted for each. One possible substitution rule for PK analyses is presented below:

Table 3-9. Possible substitutions for PK analysis

Non-numeric code	Unconditional substitution	Before Tmax	After Tmax	First of consecutive after Tmax	After first of consecutive after Tmax
BQL		0	Missing	LLOQ/2	Missing

Table 3-9. Possible substitutions for PK analysis (continued)

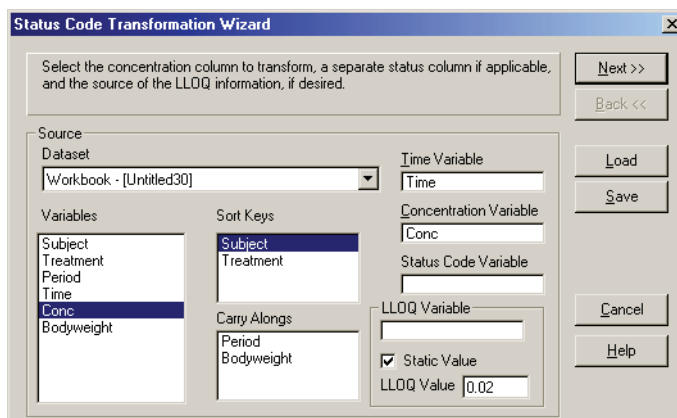
Non-numeric code	Unconditional substitution	Before Tmax	After Tmax	First of consecutive after Tmax	After first of consecutive after Tmax
NS	Missing				
ISV	Missing				
NA	Missing				
...	...				

WinNonlin provides a tool for transforming observation values from non-numeric status codes to number values for use in analyses and plots. The Status Code Transformation Wizard creates a new workbook based on an existing one, by copying over selected columns. It transforms values as specified by user-defined rules. Numeric concentration data and the non-numeric concentration status codes may be the same or different columns of the data set.

The wizard also stores information about the lower limit of quantification (*LLOQ*) for one or more sampling assays. The user may specify an LLOQ and the new column header into which it will be placed, or identify an existing column containing the LLOQ for each observed concentration. The LLOQ, if any, can be used in defining how status codes are transformed.

To select data to be transformed:

1. Open the data set in WinNonlin.
2. Choose **Tools>Status Codes (e.g., BQL)...** in the WinNonlin menus to launch the Status Code Transformation Wizard.



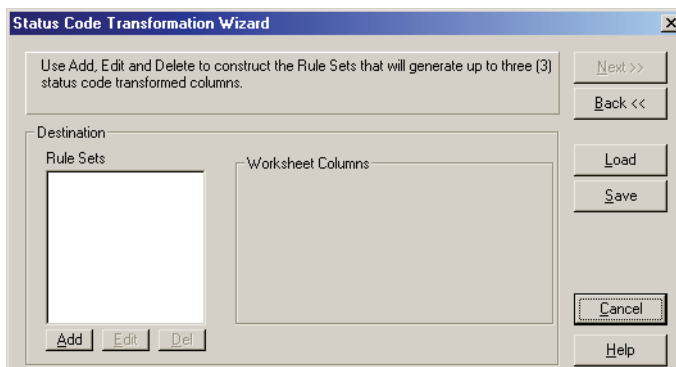
At any point in this wizard, the settings and transformation rules may be loaded or saved from/to a file using the **Load** and **Save** buttons.

3. Make sure the appropriate data set is selected under Data Set.
4. Drag and drop columns from the Variables list to assign the variables to be included and that to be transformed, as follows:
 - *Sort Keys*: each unique combination of sort key values defines one profile from which T-max will be calculated.
 - *Carry Alongs*: variables to be copied into the output data set.
 - *Time Variable*: column containing times for the data to be transformed.
 - *Concentration Variable*: column containing the data to be transformed.
 - *Status Code Variable*: (optional) column containing the non-numeric status codes associated with each datum in the Concentration Variable (above); will determine transformation of those concentration values.
 - *LLOQ Variable*: (optional) column containing the lower limit of quantification (LLOQ) associated with each measurement in the Concentration Variable; alternately, a fixed LLOQ value, set by checking **Static Value** and entering the value under LLOQ Value.
5. Click **Next** to proceed to the **Rule sets**.

Rule sets

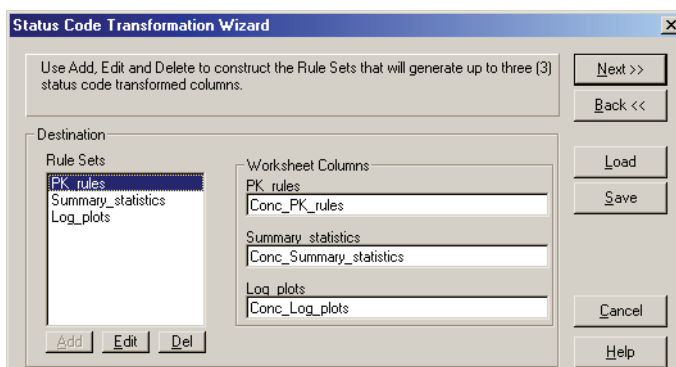
Up to three rule sets may be included in the [Status code transformations](#). Each rule set will generate a new column of transformed observation data. The new columns will use the units (if any) of the original concentration data.

A rule set includes a list of non-numeric codes and the value substitutions to be made when each code is encountered in the data. The substitutions may apply unconditionally across the data, or conditionally depending when the data point occurs relative to T_{max}, and whether two or more consecutive observations of the same code occur.



To set up the status code transformation rules:

1. Click the **Add** button under Rule Sets, and follow the instructions under “Creating a rule set” on page 73 to create a new rule set.
2. Repeat to create up to three rule sets.



Each rule set will generate a new data column in the output.

3. Click **Next** to proceed to the final dialog of the wizard.

The wizard will detect the presence of unidentified non-numeric codes in the specified column. The unidentified codes will be listed in an alert dialog.

4. To save a rule set to an external file, stored locally, click **Save**. To save a rule set to the PKS as a system object, check **PKS Load** and click **Save**.
5. Review the summary of the transformation settings. Use the **Back** button to make changes, and click the **Finish** button to create the new workbook.

Note: If code-transformed data are saved to the PKS as a scenario, the applied rule(s) will be stored with the scenario

Creating a rule set

Status code transformations can include as many as three rule sets. (See “Status code transformations” on page 69.)

	Non-Numeric Code	Unconditional Substitution	Conditional Substitution				Use When < LLOQ
			Before Tmax	After Tmax	First of Consecutive after Tmax	After First of Consecutive after Tmax	
1	BQL		0	Missing	LLOQ/2	Missing	yes
2	NS	Missing					no
3	ISV	Missing					no
4	NA	Missing					no

It is possible to load rule sets from a local file or the PKS. To import a rule set from an external file, stored locally, click **Load**. To load a rule set from the PKS, check **PKs Load** and click **Load**.

To define a rule set:

1. Enter a name under Rule Set Name. This name will be the default name for the column generated by this rule.
2. Enter any descriptive text under Rule Set Description (optional).
3. Enter each status code to be transformed in a separate row, under Non-Numeric Code. For each code, enter the following:
 - a. *Unconditional Substitution*: if the transformation is the same regardless of whether the status code occurs before or after Tmax, or is repeated, then enter the value to which the status code will be transformed here.
 - b. *Conditional Substitutions*: if the transformation depends on the status code's position relative to Tmax and/or whether the code is repeated in consecutive observations, then leave Unconditional Substitution blank and enter all four of the following:
 - *Before Tmax*: value to assign if the status code occurs before Tmax.

- *After Tmax*: value to assign if an isolated (not more than one consecutive) status code occurs after Tmax.
 - *First of consecutive after Tmax*: if the status code appears in two or more contiguous observations, after Tmax, this value will be assigned to the first of those contiguous observations.
 - *After First of Consecutive after Tmax*: if the status code appears in two or more contiguous observations, after Tmax, this value will be assigned to all but the first of those contiguous observations.
- c. *Use when < LLOQ*: click to toggle between **Yes** and **No**. One row or zero rows may be set to Yes. Yes indicates that when numeric values less than the LLOQ are encountered, those values will be transformed as though they contained the status code in that row. The LLOQ value is set in the first dialog of the Status Code Transformation Wizard, above.

Syntax for the rules: The status rules are not case-sensitive. The values entered may include numbers, operators, and the variable LLOQ, if a LLOQ value or data column was assigned (see [“Status code transformations” on page 69](#)).

4. Click **OK** when the rule set is complete to return to the Status Code Transformation wizard. See [“Rule sets” on page 71](#).

Data import and export

Sharing data with other applications

WinNonlin opens and saves a variety of data, text and graphics file types for convenient interchange with other applications. (See [Table 2-1, “Supported file formats,” on page 14.](#)) WinNonlin also exports data to several software packages, and can re-import S-PLUS and Sigma Plot script output, as follows.

- [“Microsoft Word export” on page 76:](#) automatically transfer selected text, charts and worksheets into Word text, graphics and tables.
- [“SAS Transport file export” on page 78:](#) save data in SAS-ready format.
- [“NONMEM export” on page 80:](#) export data and dosing to NONMEM-compatible files.
- [“SigmaPlot export” on page 85:](#) export data, (optional) run a script and (optional) return script output to WinNonlin.
- [“S-PLUS export” on page 86:](#) export data, (optional) run a script and (optional) return script output to WinNonlin.

The Enterprise edition of WinNonlin can, in addition, exchange data directly with Open Database Connectivity (*ODBC*) compliant databases (e.g., Oracle, SAS, etc.) For example, WinNonlin Enterprise can read rows of PK data from an ODBC-compliant database, and save analysis output back to a table in that database. WinNonlin Enterprise includes a Software Development Kit (SDK) for building custom ODBC query tools. See the following for details.

- [“ODBC import” on page 87](#)
- [“ODBC export” on page 102](#)
- [“The Enterprise Software Development Kit” on page 107](#)

WinNonlin's Enterprise edition integrates with the Pharsight Knowledgebase Server to provide secure data storage, management and security. See [“Using the Pharsight Knowledgebase Server”](#) on page 449.

Microsoft Word export

WinNonlin can export any or all open windows to a Microsoft Word document. It can also automatically export modeling or NCA output, a useful feature for automated analyses. Charts and model objects export as Windows bitmaps; thus, they cannot be edited in Word. Worksheets are exported as Word tables. Text windows become text in Word's “normal” paragraph style.

To export the active object to a new or existing Word document:

1. To append data to an existing document, open that document in Word.
2. In WinNonlin, open the worksheet, chart or text window to export. Apply any chart formatting needed; charts cannot be edited after export.
3. Click the **Export to Word** tool bar button.



If Word is not open, WinNonlin will launch it and export the active window to a new document, or append the data to the end of the open document, if any.

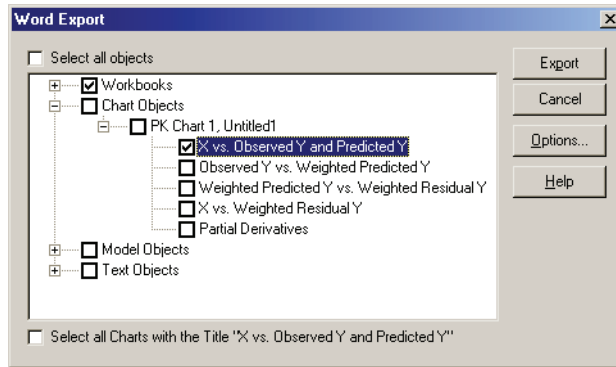
To export modeling output to Word automatically:



1. After completing model setup, click the **Start Modeling and Export to Word** tool bar button. WinNonlin will launch the analysis, create the selected WinNonlin output and export all output to Word.

To export selected objects to a new Word document:

1. Open the objects and choose **File>Word Export** from the WinNonlin menus.
2. To select specific objects for export, uncheck **Select all objects** in the Word Export dialog as shown below.



3. Click the + markers to expand the lists of windows and objects as needed.
4. Check the objects to export. Checking a window or window type will export all objects within that window or window type.
5. Use the **Options** button to set options for page layout, format, headings and object numbering, as detailed under “[Word export document options](#)”.
6. Click the **Export** button.

WinNonlin will launch Word, if needed, and export the selected objects to a new .DOC file.

Word export document options

The following options apply to [Microsoft Word export](#) using File>Word Export in the WNL menus.

Orientation: Portrait (long side of the page is vertical) or Landscape (long side of the page is horizontal).

Add source line to objects: check this option to include text describing the source file and object name to each object exported as follows.

- Charts: adds the chart name and absolute path, file name and most-recent save date for the chart, below the chart picture.
- Workbooks: adds the worksheet name and absolute path, file name and save date for the data, below the data table.
- Text: adds a heading above the text that reads, “Text -” followed by the file path and name (or window name, if unsaved) in square brackets.

Word export chart options

Multiple charts per page: Check this box and select the layout below to put more than one chart on a page, with page breaks between sets of charts.

Add page break: forces a page break before and after each chart.

Add log charts: creates and exports a chart with a logarithmic Y-axis for each linear scatter chart. Each log chart appears below the original linear chart.

Start figure numbers at N: adds figure numbers to charts. Check this option and enter a number for the first chart to include “Figure *N*” above each chart.

Chart format: export charts as Windows metafiles or bitmaps.

Word export workbook options

Preserve formatting for tables: includes formatting for workbooks created in the WinNonlin Table Wizard. This option can slow the export significantly.

Start table numbers at N: adds table numbers to exported worksheets. Each worksheet is one table in the Word export. Check this option and enter a number for the first table to include the text “Table *N*” above each table.

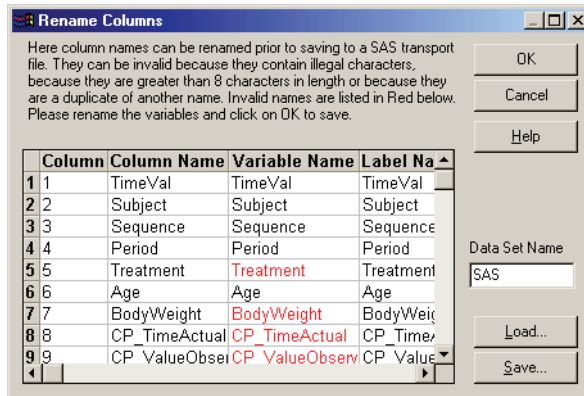
SAS Transport file export

Any worksheet can be saved to a SAS Transport file.

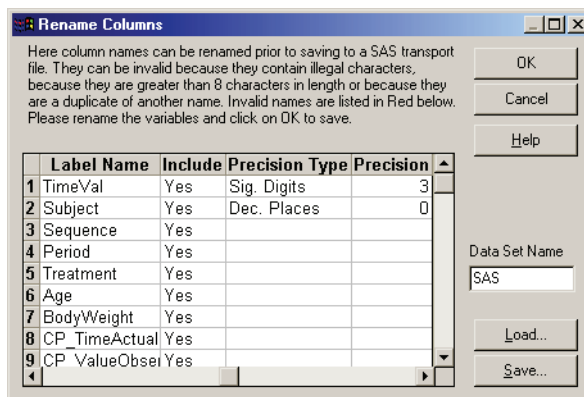
To export data to SAS (via SAS Transport files):

1. With the desired worksheet active in WinNonlin, select **Save As** from the **File** menu. The Save File dialog appears.
2. Select SAS Transport (*.xpt, *.stx) in the **Save as Type** list.
3. Name the file and click **Save**.

Because SAS field names are more restrictive than WinNonlin column names, WinNonlin will request new names for any that would be invalid in SAS.



4. If the Rename Columns dialog appears, enter new names under Variable Name for any names that appear in red text. The new names must be no longer than 8 characters, and are limited to alphanumeric characters and the underscore.
5. Optionally, edit data set name. You may also edit the label names and precision for exported values. Scroll right and click on a cell under Precision Type to toggle between significant figures (Sig. Digits) and decimal places. Enter the desired number of significant figures or decimal places under Precision.



6. Use the **Load** and **Save** buttons to save these settings to an ASCII text mappings file (*.MAP), or reload previously-saved settings.
7. Click **OK** to create the SAS Transport file.

NONMEM export

WinNonlin provides a NONMEM Export function to save observation and (optional) dose data in a NONMEM-compatible data file (*.DAT). NONMEM export can also generate a NONMEM control stream containing an outline of required commands or commands copied from a user-specified template.

Data file structure

The NONMEM data file contains two sections: the headers and data.

The following four items can appear in the header of a NONMEM data file exported from WinNonlin:

1. Names of columns. The first line in the header contains the names of the columns in the data file. This will appear as a comment like that below:

```
# ID TIME WT GEND AGE BP DV MDV DVID AMT RATE EVID
```

2. Encoding. All NONMEM data are numeric. Thus, discrete values stored as strings, e.g., "A," "B," etc. or "male" and "female" must be encoded as numeric values for use in NONMEM. Typically, this encoding is explained in the header. For example, the header might contain:

```
# Variable GEND
```

```
# male = 1
```

```
# female = 2
```

3. DVID. The exported DV column can hold one or more dependent variable(s). The DVID column disambiguates between dependent variables. The header can include the DVID column mappings. For example:

```
# Variables in DV column
```

```
# DEPC = 1
```

```
# DEPC = 2
```

Thus, whenever DVID is 2, the value in DV represents DEPC.

4. Abbreviations. NONMEM places a four-character restriction on names. Comments in the header can indicate how WinNonlin variable names map to those used in NONMEM.

NONMEM data may include the following columns.

Table 4-1. NONMEM columns

Position	Name	Presence	Description
1	ID	Always provided.	Subject identifier.
2	TIME	Usually required.	Nominal or actual time.
3	independent variables	Optional. User may include as many as are available.	The value of the independent variables. There will be one column for each independent variable.
4	DV	Required.	The dependent variable's values. The name DV is a fixed name for NONMEM.
5	MDV	Always provided. (If DV is never missing, then MDV can be skipped, but, for simplicity, one should always provide MDV.)	The status of the dependent variable: 0: The row includes a valid value of DV 1: The row does not include an observation of DV or the value is missing.
6	DVID	Provided only when more than one DV item is present.	If more than one dependent variable (DV item) is exported, this column disambiguates. Thus, if three DV items are chosen, this column will equal 1 for the first, 2 for the second, and 3 for the third.
7	AMT	Required for PK.	The amount value of the dosing variable.
8	RATE	Frequently provided.	For infusions, the dose rate. RATE = 0 for bolus dosing. A positive value indicates an infusion. -1 indicates a zero-order bolus for which NONMEM calculates the duration. -2 means that a zero order bolus for the rate is calculated. The RATE column shows a dot when no value is administered.
9	CMT	Usually provided.	Indicates the compartment into which a dose was administered or from which DV was observed.
10	EVID	This form export should always include it.	The event type: 0: An observation event (for DV) 1: A dosing event 2: An other-type event 3: A reset event 4: A reset-and-dose event

Exporting to NONMEM

To export data to NONMEM:

1. Load a data set and a compiled compartmental PK, PK/PD link or indirect response model.
2. Set up the model variables as detailed under “Modeling” on page 195.
3. Specify the dosing (if any). Dose data for NONMEM export is drawn from the WinNonlin model. To be available for export, dose data must be either entered the Model Properties dialog (Dosing Regimen tab) or available on the Dosing worksheet of PKS study data (WinNonlin Enterprise only).
4. Choose **File>NONMEM Export**. The Export to NONMEM dialog appears.
5. Set the export data and settings using the following tabs.

Tab name	Description
Files	Export file path and name, header information, comment identifier
Dependent variables	Dependent variables and dependent variable ID column
Other data items	Independent variables
Dose information	Dosing variables and compartment to dose into (optional)
Occasions	Identifiers for within-subject occasions to be considered separately, as an additional means to explain variability

6. When all settings are complete, click **OK** to create the NONMEM file.

Files

Problem (optional): This string becomes the heading for the **NONMEM export** file. It is useful to enter descriptive comments here to make it clear what is in the data file. If a control stream is output with the data, then this string will appear in the \$PROB statement.

Data File: Enter the name of the output data file (*.DAT) to be created, including the full (absolute) path, or use the Browse button to set it. If a control stream is output with the data, this string will appear in the \$DATA statement.

Comment character: Enter a character to precede and identify comments in the output file. The default is: #. This character will precede the header rows, and appears in the \$INPUT COMMENT= command in the control stream.

Control stream: Select the appropriate option:

- **Do not generate control stream:** available for users who wish to write their own control stream. Inhibits the creation of a new control stream.
- **Generate skeleton control stream** generates a skeletal control stream, consisting of the \$PROB, \$INPUT, and \$DATA commands only. Specify the file name, including the full path, or use the Browse button to set it.
- **Generate control stream from template** copies a user-supplied control file template and inserts the appropriate \$PROB, \$INPUT, and \$DATA commands at the beginning. Enter the names, including full paths, for the file to create (above) and the template file (below) or use the adjacent Browse button to locate each.

Dependent variables

Specify one or more dependent variable(s) for the [NONMEM export](#) by checking the box under **Select?** for the appropriate columns. All dependent variables must use numeric values.

If a single dependent variable is specified, the header comments in the NONMEM data file will indicate “DV=COL” where “COL” is the NONMEM abbreviation entered in this dialog box. If more than one dependent variable is included, all will appear stacked in a single column identified in the headers as DV. Additional header comments will indicate which column (NONMEM abbreviation) is associated with each DVID value.

CMT stands for compartment. If the dependent variable is associated with a particular compartment of a PK model, enter the compartment number here. CMT must be a non-negative integer, usually less than 10.

DVID: If multiple dependent variables are included, WinNonlin will create a dependent variable ID (DVID) column to disambiguate them. Enter a unique positive integer value for DVID for each dependent variable.

Other data items

Use this tab in the [NONMEM export](#) dialog to identify the subject ID, time and other independent variables, and to set their names in the exported data.

To identify independent variables for export:

1. Check the box under **Select?** for:
 - a. one variable representing subject identifiers (required),
 - b. one variable representing time (required)
 - c. any number of independent variables (optional).
2. Edit the values under NONMEM Abbreviation as necessary to read:
 - a. ID for the subject identifier,
 - b. TIME for the time variable, and
 - c. a unique name up to four letters, numerals and/or underscore characters for each other independent variable. Case will be preserved as entered.

Each four-character abbreviation must be unique and cannot be a NONMEM reserved name.

Dose information

Dose data for NONMEM export is drawn from the WinNonlin model, and must be specified in either the Model Properties dialog (Dosing Regimen tab) or via the Dosing worksheet included with PKS study data (WinNonlin Enterprise only). The Dose Information tab of the **NONMEM export** sets options for exporting this dose information. These options include:

- **Derive RATE from the data.**
- **Treat as bolus.** All doses will be given a RATE of 0.
- **Have NONMEM calculate duration (RATE=1).** For infusions, this allows estimation of the duration parameter.
- **Have NONMEM calculate rate (RATE=2).** For infusions, this allows estimation of the rate parameter.
- **Include no dose information.** In this case, no dose data are exported.
- **Associate dose with a compartment:** To dose to a specific PK compartment, check this box then enter the compartment number under CMT. CMT must be a non-negative integer less than 10. See the NONMEM NM-TRAN documentation to determine which numbers to enter.

Occasions

Including occasional variation in a model can sometimes explain additional variability in the data. The Occasions tab in the **NONMEM export** dialog exports a separate column to differentiate occasions (e.g., separate profiles within subjects). At the beginning of an *occasion*, NONMEM will reset all compartment concentrations to 0.

Use occasions: check this box to generate an occasion identifier, and select a method for creating occasions:

- list the time values at which new occasions start, from earliest to latest (do not include time 0), or
- indicate how long each occasion lasts, and (optionally) when the first occasion begins.

SigmaPlot export

WinNonlin provides options to export worksheet data to SigmaPlot, with or without a script to run. WinNonlin can automatically import script output, if any.

Note: SigmaPlot export requires installation of SigmaPlot 8 or higher on the same machine as WinNonlin.

To export data to SigmaPlot and (optional) run a script:

1. Open the data to export in a WinNonlin workbook.
2. To include and run a script, open or create it in a WinNonlin text window.
3. Choose **File>Sigma Plot Export...** from the WinNonlin menus.
4. Select the data to export under Workbook and Sheet. All columns in the worksheet will be exported to a SigmaPlot notebook.
5. Enter a name for the new notebook under Note Book Name.
6. To include and run a script, choose its text window (or file path) from the list under Optional Script. The script must be open in a WinNonlin text window.
7. To automatically load script output, if any, into WinNonlin:

- a. Check **Wait for completion**.
 - b. Enter the full path to the SigmaPlot output folder under Output or click the “...” button to select the folder.
8. Click **OK** to export the data and execute the script, if any. If you chose to return script output to WinNonlin, do not perform any actions in WinNonlin until the script run completes and its output appears in WinNonlin.

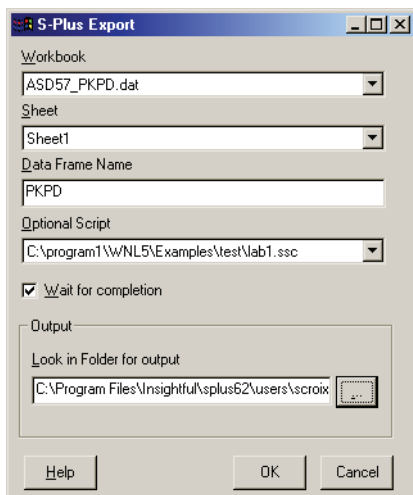
S-PLUS export

WinNonlin provides options to export worksheet data to S-PLUS, with or without a script to run. WinNonlin can automatically import script output.

Note: S-PLUS export requires that S-PLUS version 6.1 is installed on the same machine as WinNonlin.

To export data to S-PLUS and (optional) run a script:

1. Open the data to export in a WinNonlin workbook.
2. To include and run a script, open or create it in a WinNonlin text window.
3. Choose **File>S-PLUS Export...** from the WinNonlin menus.
4. Select the data to export under Workbook and Sheet. All columns in the worksheet will be exported to an S-PLUS data frame.



5. Enter a name for the new data frame under Data Frame Name.
6. To include and run a script, choose its text window (or file path) from the list under Optional Script. The script must be open in a WinNonlin text window.
7. To automatically load script output, if any, into WinNonlin:
 - a. Check **Wait for completion**.
 - b. Enter the full path to the output folder under Output or click the “...” button to select the folder.
8. Click **OK** to export the data and execute the script, if any. If you chose to return script output to WinNonlin, do not perform any actions in WinNonlin until the script run completes and its output appears in WinNonlin.

ODBC import

Use ODBC import to load data from an *ODBC*-compliant database directly to a WinNonlin worksheet. An import involves establishing a connection to the database, then defining a query to draw specific fields and records from it. The connection and query settings can be saved to an import file (*.IMP), or *import*, for later re-use. Instructions are provided under:

- “Creating an import” below
- “Loading an existing import” on page 96

WinNonlin also provides a Custom Query Builder for creation of dynamic link library (DLL) files that can access additional data source types, including Watson version 6.0 DMLIMS. See “The custom query builder” on page 97.

Once ODBC-imported data are loaded in WinNonlin, the ODBC import command provides the option to update the data to reflect the current database content. See “Refreshing an imported data set” on page 97 for instructions.

See also “ODBC export” on page 102.

Creating an import

An *ODBC import* specification (optionally saved as *.IMP) contains all information required to draw data from a remote data source. It includes two parts:

1. *Database connection*: Both ODBC import and export use a *connection string* that defines the database type and location. This string can be cre-

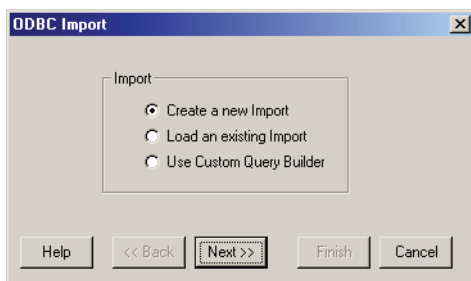
ated by pointing to the database type and file, or loaded from a previously-saved import or export as detailed under [“Connection” on page 90](#).

2. *Query*: Once the database connection is established, an import requires a query to set which data table, field(s) and records to import. The query may be created in the ODBC Import wizard or loaded from previously-saved import as described under [“Queries” on page 92](#). WinNonlin translates the query into Structured Query Language (SQL), which can be edited before executing the import.

One connection string can be re-used with any number of different queries; each can be loaded separately.

To create and save (optional) a new import:

1. Choose **File>ODBC import** from the WinNonlin menus.

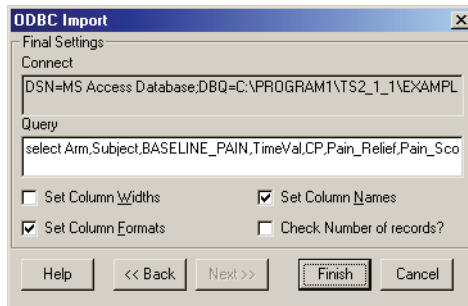


2. Select **Create a new import** in the ODBC Import wizard and click **Next**.
3. Create or load a connection string as detailed under:
 - a. [“Creating a new connection” on page 90](#)
 - b. [“Loading a saved connection” on page 92](#)
4. Create or load a query as detailed under:
 - a. [“Building a query” on page 93](#)
 - b. [“Loading a saved query” on page 96](#)
5. Proceed to the [Final settings](#) dialog.

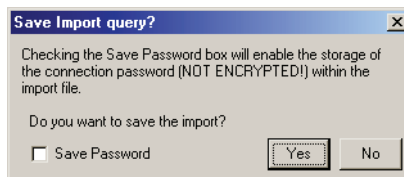
Final settings

The Final Settings dialog summarizes the [ODBC import](#). Its top section displays the connection string. The middle section contains the SQL string for the query. The lower section determines data formatting in WinNonlin.

1. Choose settings in this dialog (below), and edit the SQL if desired.



- **Set Column Widths** adjusts the destination worksheet's columns to match the widths of the fields in the database.
 - **Set Column Names** adjusts the destination worksheet to have the same column names as the source fields.
 - **Set Column Formats** applies the appropriate number formats to columns containing times, currency, and dates.
 - **Check Number of Records?** provides a count of filtered records prior to the import, and the option to return and edit the query if needed.
2. Click **Finish** to import the selected data to a WinNonlin workbook.
 3. If **Check Number of Records?** was checked, a dialog appears with the number of records returned by the filter. Click **OK** to proceed.



Save Password: This option includes the database user name and password in the import file with the connection string. This allows scripting to use ODBC connectivity without end user intervention (see “[FileType](#)” on page 597).

CAUTION: The import file is not encrypted, so a password saved within is not secure.

4. To save the import (*.IMP), click **Yes** and save to the appropriate directory. Else, click **No**. In either case, WinNonlin will then execute the import.

WinNonlin can import up to 65,000 rows into one worksheet. Each cell can hold up to 16KB of text. The ODBC drivers might impose additional limits.

Connection

The first step in an **ODBC import** or **ODBC export** is to establish a connection to a specific database.

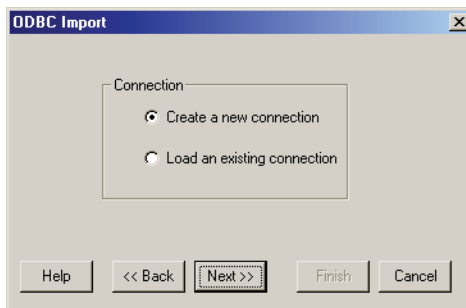
“**Creating a new connection**” on page 90 creates a new *connection string*, defining the database type and location. This connection string may be saved as part of an import or export settings file for re-use.

“**Loading a saved connection**” on page 92 extracts the connection string from a previously-saved import or export file. This establishes a link to the database, to be used with a new or saved query or export.

Creating a new connection

To create a new connection string:

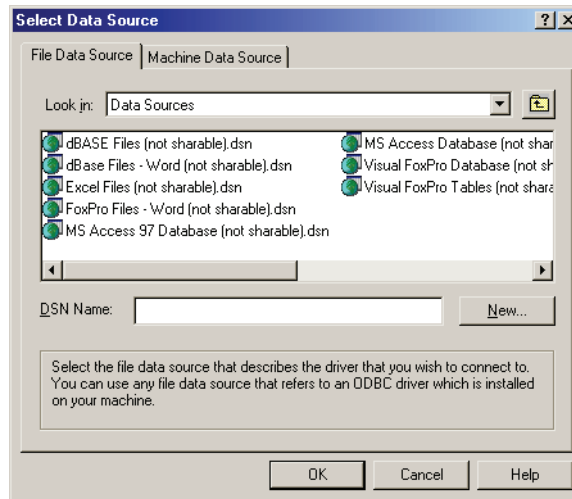
1. Choose **File>ODBC Import** (or **Export**) from the WinNonlin menus.
2. Select **Create a new Import** (or **Export**) in the ODBC Import (or Export) wizard, and click **Next**. The import wizard is shown below.



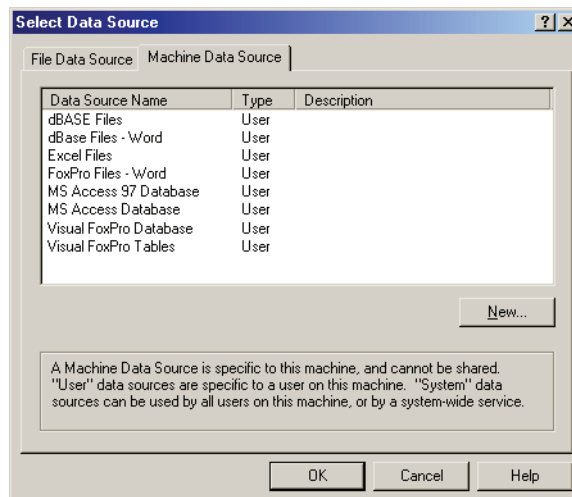
3. Select **Create a new connection** and click **Next**.
4. Select a data source from one of the two tabs in the Select Data Source dialog.

The File Data Source tab (below) lists all file Data Source Names (DSN's) and subdirectories on the system. These are file-based data sources that can be shared among all users who have the same ODBC drivers installed. The data sources need not be dedicated to a single user or local to a specific computer.

New data sources can be set up using the New button. See the Microsoft Windows documentation for details.



The Machine Data Source tab (below) uses information stored in the machine registry, which provides the majority of the connection string information.



5. Select a data source and click **OK**.
6. If a log in dialog appears, enter the appropriate log in ID and password for the data source. These are set up by the Database Administrator or Information Technology personnel. Click **OK** to proceed with the import or export.

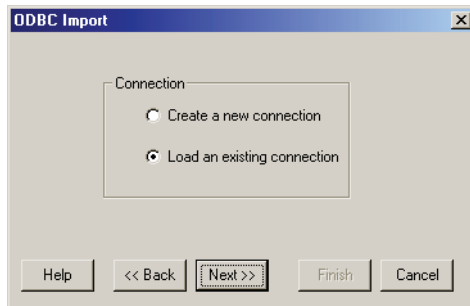
The next step is to set up a query to pull the desired data from the database, or an export definition to map WinNonlin data columns to database fields. See [“Queries” on page 92](#) or [“Export definition” on page 103](#).

Loading a saved connection

Once a connection string is saved in an import (*.IMP) or export (*.EXP) file, it can be loaded for use with different import queries or export definitions.

To load an existing database connection:

1. Choose **File>ODBC Import** or **File>ODBC Export** from the menus.
2. Select **Create a new Import (or Export)** in the first dialog and click **Next**.



3. Select **Load an existing connection** and click **Next**. An Open dialog appears.
4. Select the import file (*.IMP) or export definition file (*.EXP) containing the connection string for the desired database and click **Open**.
5. Proceed with the import or export, as detailed under [“Queries” on page 92](#) for imports or [“Export definition” on page 103](#) for export.

Queries

Once a database connection string is established (see [“Connection” on page 90](#)), ODBC import requires a query specifying the data table, field(s) and records to be imported, as detailed in the following sections.

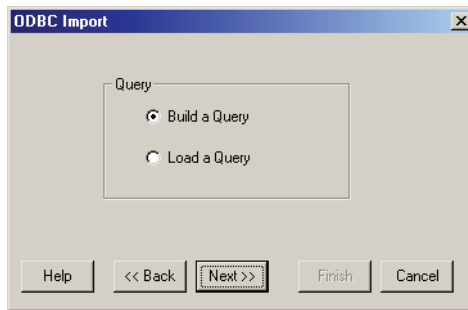
- [“Building a query” on page 93](#)
- [“Loading a saved query” on page 96](#)

Building a query

A *query* notes the database table and fields (optional) to import to WinNonlin from an ODBC-compliant database (see “[ODBC import](#)” on page 87). It can also include filters to extract specific records from the table.

To create a new query:

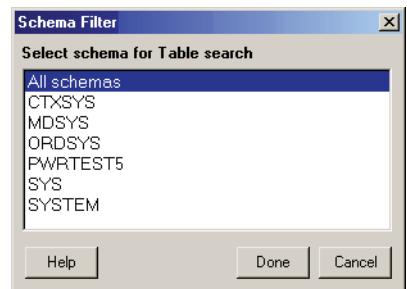
1. Choose **File>ODBC Import** from the WinNonlin menus.
2. Load a new or saved connection string, as detailed under “[Creating a new connection](#)” on page 90 and “[Loading a saved connection](#)” on page 92.



3. Select **Build a Query** and click **Next** to proceed to “[Schema filter](#)” below.

Schema filter

The WinNonlin Query Builder will access the database, and load a list of the available tables. If the appropriate ODBC drivers are available, the Schema Filter dialog appears. It provides a means to filter the available database tables by schema, a metadata organizational tool. These schema were created by your database administrators.

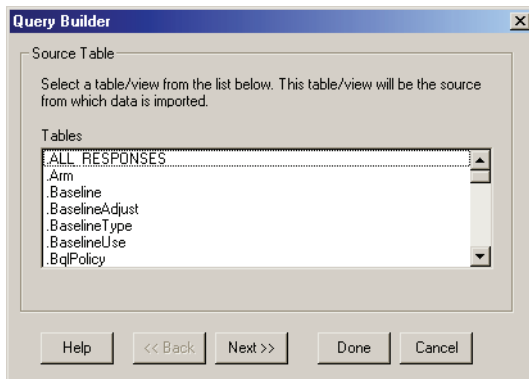


If the Schema Filter dialog appears while building a query (“[Building a query](#)” on page 93), select the schema to use in filtering the list of data tables then click **OK** to proceed to “[Query builder](#)” below.

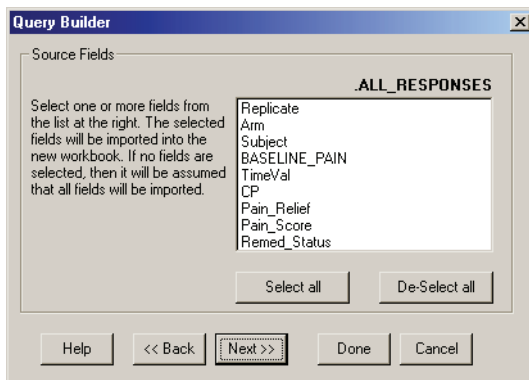
Query builder

To select source tables and fields while building a query:

1. Begin the query setup as described under “Building a query” on page 93.



2. When the Source Tables list appears, select a table from which to import data.
3. Click **Next** to select specific fields to be imported, or click **Done** to load all fields in the table and proceed to step 7 below.



4. Select one or more fields to import. To import all the records from those fields, click the **Done** button and proceed to step 7 below. To filter the records, click **Next**.

Query Builder

Source Filter

The current, unfiltered query would return 252 rows. A filter could reduce the number of rows retrieved. Define a filter that will be used to determine which of the fields will be imported. Selecting the List Distinct Values button will populate the dropdown list box with all the distinct values associated with the selected variable.

Field Name: Operator:

Filters

Criteria
☒ And
☐ Or

5. To create the filter:

- Select a Field Name to filter on from that list box.
- Choose an operator from that field's list box.
- Enter a value in the Field Value field, click the **List Distinct Values** button to select from among all existing values for that field.
- Click the **Add** button to set the filter.
- If more than one criterion will be applied, select the **And** or **Or** radio button and return to step a above. Filters are applied in the order created.

Query Builder

Source Filter

The current, unfiltered query would return 252 rows. A filter could reduce the number of rows retrieved. Define a filter that will be used to determine which of the fields will be imported. Selecting the List Distinct Values button will populate the dropdown list box with all the distinct values associated with the selected variable.

Field Name: Operator:

Filters

Criteria
☒ And
☐ Or

6. Click **Done** when the filters are complete.

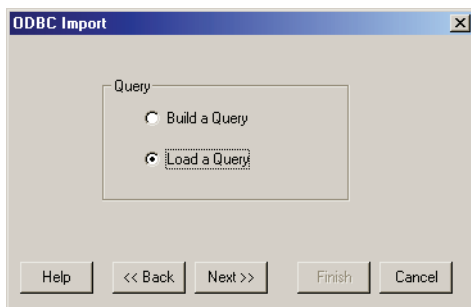
7. The Import: Final Settings dialog appears. Follow the instructions under “Final settings” on page 88 to complete the import.

Loading a saved query

Once a connection string (new or existing) has been established as part of [Creating an import](#), any saved query can be loaded so long as it uses tables, fields and record types in the database being connected to.

To load an existing query:

1. Use **File>ODBC Import** from the WinNonlin menus to connect to a new or previously-saved database, as detailed under “[Creating a new connection](#)” on page 90 and “[Loading a saved connection](#)” on page 92, respectively.



2. In the ODBC Import: Query dialog select **Load a query** and click **Next**.
3. An Open dialog appears. Select the appropriate import (*.IMP) file and click **Open**. The ODBC Import: Final Settings dialog appears with the connection string and query loaded. See “[Final settings](#)” on page 88 to complete the import and load the data into a WinNonlin workbook.

Loading an existing import

An import setting file (*.IMP) contains the connection string and query to quickly re-load data from an ODBC database into WinNonlin. It also provides a convenient template for a new [ODBC import](#).

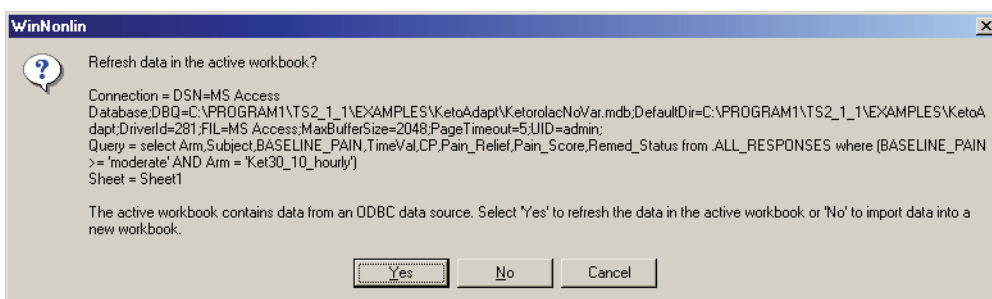
To load an existing import:

1. Choose **File>ODBC Import** from the WinNonlin menus.
2. In the ODBC Import dialog select **Load an existing Import** and click **Next**. An Open dialog appears. The default file type is *.IMP.
3. Select the file with the desired connection string and click **Open**. The ODBC Import: Final Settings dialog appears with all settings for that import loaded.

4. Makes changes as needed in the Import: **Final settings** dialog, and using the **Back** button.
5. When the settings are correct, click the **Finish** button. The data from the database are then loaded into a WinNonlin worksheet.

Refreshing an imported data set

When data from an **ODBC import** are open in WinNonlin, those data can be refreshed to reflect the current the database source by choosing the Import ODBC command from the File menu.



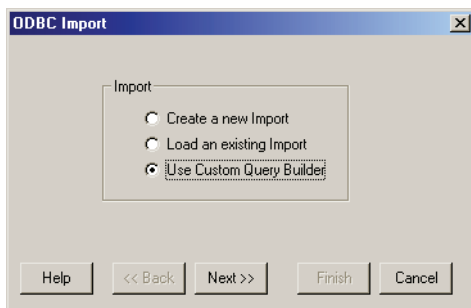
Click **Yes** to refresh the data in the worksheet from the database. Even if the workbook has been saved locally, it can be re-opened and refreshed following this same procedure.

The custom query builder

The Custom Query Builder provides a means to build custom dynamic link library (DLL) files that access additional data source types, including Watson version 6.0 DMLIMS. Further, WinNonlin Enterprise provides templates for creating the DLL files to access custom data sources.

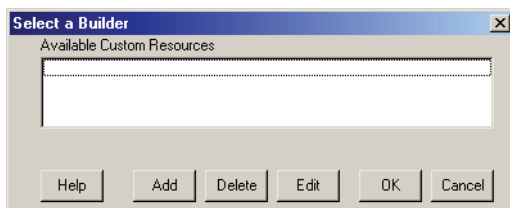
To use the custom query builder with Watson v. 6.0:

1. Select **File>ODBC Import** from the menus.



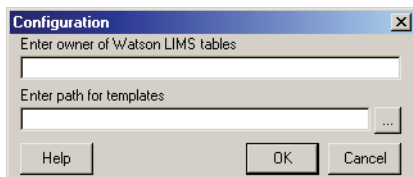
2. Select **Use Custom Query Builder** and click **Next** to proceed to “[Select a builder](#)” below.

Select a builder



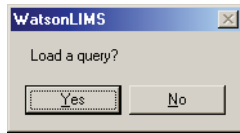
The Select a Builder dialog appears as part of the Custom Query Builder (see “[The custom query builder](#)” on page 97) to set the data resources to access. If the Custom Query Builder has not been used before, this dialog may be empty.

1. To add resources to the Query Builder, click **Add**, then double-click the appropriate DLL file, e.g., WATSON_DMLIMS.DLL.
2. To configure the Watson DMLIMS custom resource to ensure that data are imported correctly, select that item and click **Edit**.

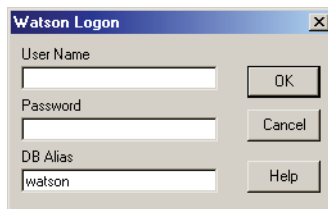


3. In the Configuration dialog, enter the name of the Watson LIMS tables owner.
4. Enter the full directory path for the Watson templates or use the **Browse** button to locate the directory on your local machine or on a network server.

5. Click **OK** to close the Configuration dialog.
6. Highlight **Watson DMLIMS** (or another custom query builder) in the Select a Builder dialog and click the **OK** button.



7. A dialog asks whether to load a query.
 - a. To load a saved query, click **Yes**. The Open File dialog appears and provides the opportunity to select an existing query. Select a file (file type *.WTS for Watson LIMS) and click **Open**. The query is read and the file data is loaded into a worksheet. The import is complete.
 - b. If no query has been saved for this builder, click **No** and proceed with the following steps. A log on dialog appears.



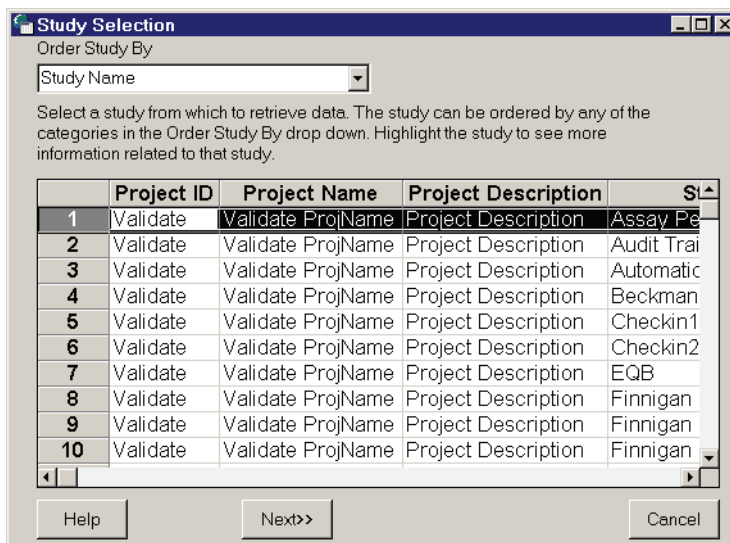
Note: The user name, password, and alias for any custom database system (such as the Watson DMLINS) are set by the database administrator or IT personnel. Request this information from that person or department.

8. Enter user name, password, and database alias, and click **OK** to proceed to the [Study selection](#) dialog.

Study selection

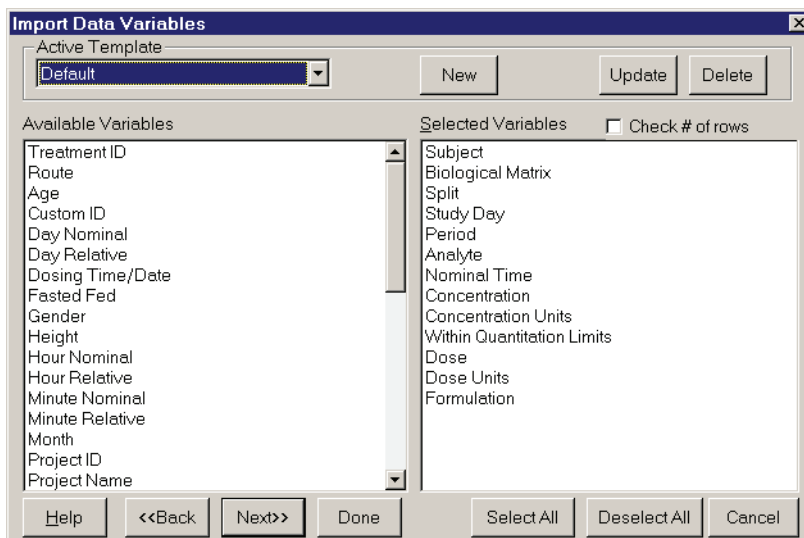
The Study Selection dialog lists the resources available in the data source for a custom query (see [“The custom query builder” on page 97](#)).

1. Select a study by clicking on the row. Use the Order Study By field to sort any category (e.g., study number/ID, study director, etc.) in order to find a particular listing more quickly.



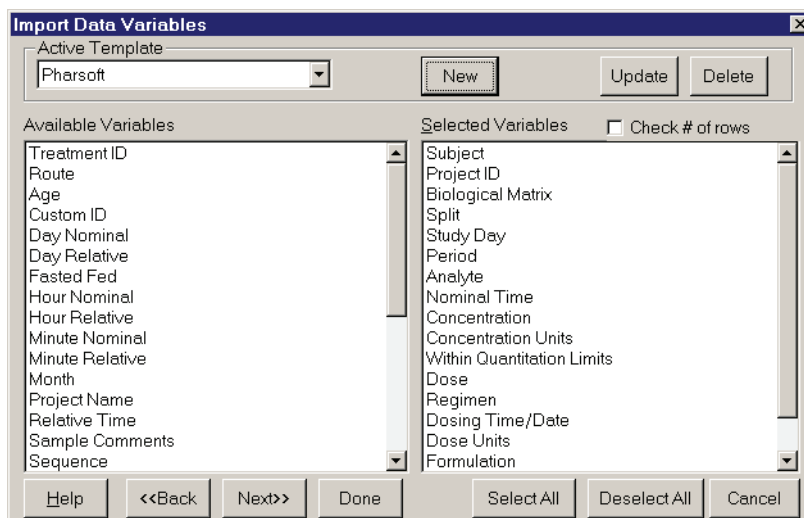
Note: If no projects are listed in this dialog, contact your IT department.

- Click **Next**. The Import Data Variables dialog appears, containing the list of variables available in the selected database.



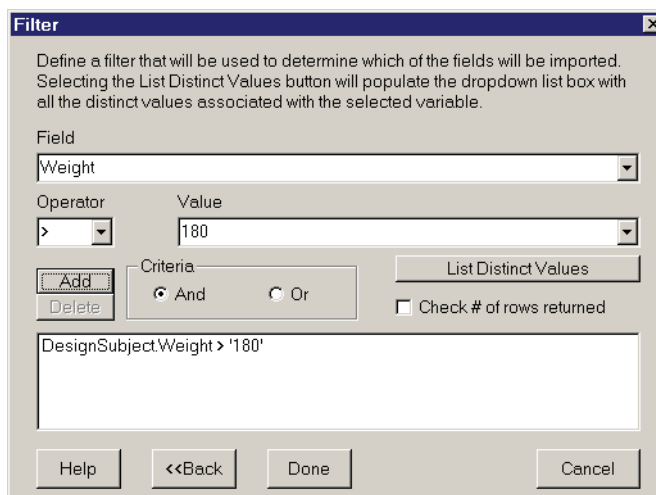
- If all available variables are desired, click the **Select All** button. For a subset, drag each variable to the **Selected Variables** field.

4. A particular combination of variables can be saved as a template. To create a new template drag the variables for the template to the **Selected Variables** field and click the **New** button. Then enter a name for the template.



Note: The fields are imported in the order shown in the **Selected Variables** list.

5. Click **Done** if all the data in the selected fields are to be imported. The data can also be filtered further by clicking **Next**. The Filter dialog appears.



A filter is defined by specifying one field name, operator, and field value.

6. To set a filter: Select the field to filter on from the Field box. Choose the operator below. Either enter a value in the Value field or select from all specified values of that field. Alternately, click the **List Distinct Values** button and select a value. Click the **Add** button to set the filter. Use the And/Or operators to create compound filters (e.g., Age > 12 and Gender = Male).
7. Click **Done** to load the data into a WinNonlin worksheet. If **Check # of rows returned** was checked, WinNonlin will report a count of records returned.

ODBC export

WinNonlin Enterprise edition can export selected worksheet columns to specific fields within a table in any *ODBC*-compliant database. Like *ODBC import*, the export requires a *connection string* defining the database type and location. It also requires an *export definition* to map columns in the source worksheet to fields in the target database table. Both the connection string and export definition can be saved to an export file (*.EXP) for later re-use. See:

- “Creating an export” below
- “Loading a saved export” on page 107

Creating an export

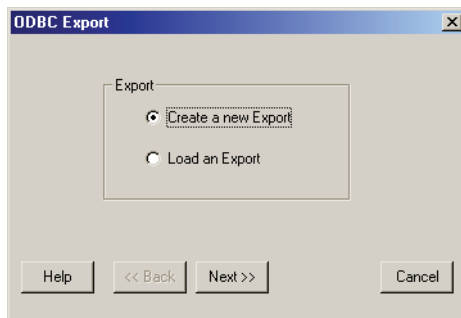
An *ODBC export* specification (optionally saved as *.EXP) contains all information required to save data directly to an external database. It includes:

1. *Database connection*: Both ODBC import and export can use the same *connection string* to define the database type and location. This string is created in the ODBC Export wizard or loaded from a previously-saved import or export as detailed under “*Connection*” on page 90.
2. *Export definition*: Once the database connection is established, an export requires a mapping of columns from the WinNonlin worksheet to fields in the target database table. This export definition may be created in the ODBC Export wizard or loaded from a previously-saved export as shown under “*Export definition*” on page 103.

One connection string can be re-used with any number of different export definitions; each can be loaded separately.

To create and save (optional) a new export:

1. Choose **File>ODBC export** from the WinNonlin menus.



2. Select **Create a new export** in the ODBC Export wizard and click **Next**.
3. Create or load a connection string as detailed under:
 - a. “[Creating a new connection](#)” on page 90
 - b. “[Loading a saved connection](#)” on page 92
4. Create or load an export definition as detailed under:
 - a. “[Building an export definition](#)” on page 104
 - b. “[Loading a saved export definition](#)” on page 106
5. Make any adjustments needed in the [Field selection](#) dialog and click **OK** to complete the export.
6. After the export completes, WinNonlin will ask whether to save the export settings. Click **Yes** to create a new export (*.EXP) file. Else, click **No**.

Export definition

An export definition maps WinNonlin worksheet columns to fields in one table of the target database for an [ODBC export](#).

- “[Building an export definition](#)” on page 104 specifies which data are to be exported, to what tables and fields in the selected database. The settings can be saved to an export definition file (*.EXP).
- “[Loading a saved export definition](#)” on page 106: One connection string can use any number of different export definitions.

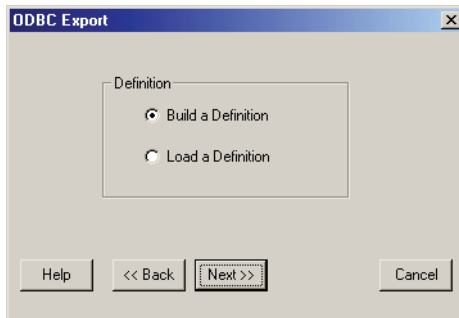
Building an export definition

Export definitions require that a connection string be loaded first, to connect to the target database.

To create a new ODBC export definition:

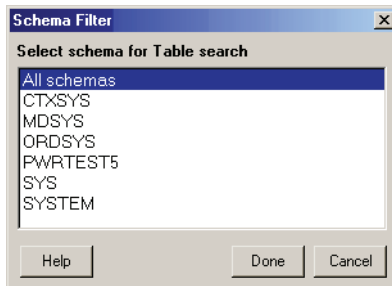
1. Open the set to export in WinNonlin and choose **File>ODBC Export** from the menus. The ODBC Export dialog appears.
2. Select **Create a new Export** and click **Next**.
3. Create or load a connection string as detailed under:
 - a. “Creating a new connection” on page 90
 - b. “Loading a saved connection” on page 92

After a connection string is set, the ODBC Export: Definition dialog appears.



4. Select **Build a Definition** and click **Next**.

As with the [ODBC import](#), the schemas to use in selecting the data variables can be set. The Schema Filter dialog displays available schemas, if any have been set up by your IT personnel.



5. If the Schema Filter dialog appears, select one item and click **OK**. Using schema filters can speed up export operations using saved export files.
6. Proceed to the **Field selection** dialog to complete the export.

Field selection

This dialog determines the core of an **Export definition**; it sets which data columns are exported to which database tables and fields.

Field Name	Data Type	Field Size	Fixed
Replicate	Number	8	<input type="checkbox"/>
Arm	Text	100	<input type="checkbox"/>
Subject	Number	8	<input type="checkbox"/>
BASELINE_PAIN	Text	100	<input type="checkbox"/>
TimeVal	Number	8	<input type="checkbox"/>

1. Select the database table to receive the exported data under Table Name.
2. Drag the variables from the Variables list to the appropriate database fields under Field Definitions.
3. To export one constant value for all records in a given field, check the **Fixed** check box for that variable and enter the value under the field name. This method can be used to differentiate one export from another.

The data type and field size are set by your Database Administrator.

4. Click **OK** to export the data to the database. If this is a new export, WinNonLin will ask whether to save the connection string and export definition to a new export file (*.EXP). This file contains all information needed to reproduce the same export, excluding any constants in fixed fields.

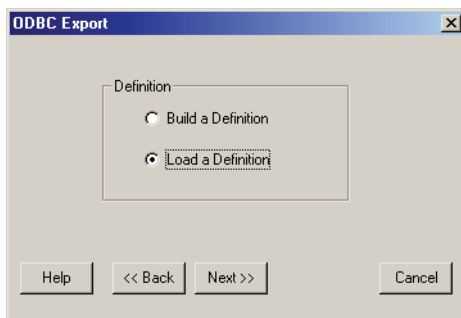
Loading a saved export definition

The field mappings in any [ODBC export](#) definition file (*.EXP) can be re-used with any database connection, assuming the field and table names in the export file exactly match a table in the selected database.

To load an export definition:

1. Open the set to export in WinNonlin and choose **File>ODBC Export** from the menus. The ODBC Export dialog appears.
2. Select **Create a new Export** and click **Next**.
3. Create or load a connection string as detailed under:
 - a. [“Creating a new connection” on page 90](#)
 - b. [“Loading a saved connection” on page 92](#)

Once the connection string has been loaded, the ODBC Export: Definition dialog appears.



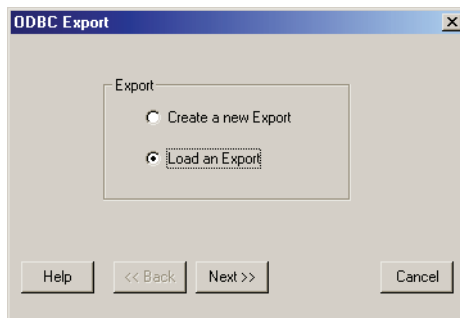
4. Select **Load a Definition** and click **Next**.
5. In the Open dialog, select the export file (*.EXP) and click the **Open** button.
6. Log on to the data source, if necessary.
7. If the Schema Filter dialog appears, use the Schema Filter to narrow the set of available data tables (optional) and click **OK**.
8. Make any adjustments needed in the [Field selection](#) dialog and click **OK** to complete the export.
9. If this is a new export, WinNonlin will ask whether to save the export settings. Click **Yes** to create a new export (*.EXP) file. Else, click **No**.

Loading a saved export

Once an [ODBC export](#) has been created and saved, the export settings, including the connection string and/or the export definition, can be reloaded for reuse with any data set containing the column names used in the export file. This feature extracts both the connection string and the export definition from a saved export file (*.EXP).

To load an existing export:

1. Choose **File>ODBC Export** from the WinNonlin menus. The ODBC Export: Export dialog appears.



2. To load both a connection string (pointer to a specific database) and export definition (mappings), select **Load an Export** and click **Next**. An Open dialog appears.
3. Select the desired export definition file (*.EXP) and click **Open**.
4. Log on to the data source if needed.

The Field Definitions are loaded automatically in the Field Selection dialog.

5. Make any needed edits to the export settings, and click **OK** to export the data.

The Enterprise Software Development Kit

The Enterprise edition of WinNonlin includes a Software Development Kit (SDK) for Developers, System Administrators and Database Administrators who wish to create custom query builders for ODBC connectivity.

Import file format

The Import file is an ASCII file, 6 lines contained in quotes.

```

        BEGIN FILE (Note 6 lines)
1  Connect string
2  Query
3  Set Column Formats (0 or 1:True or False)
4  Set Column Names (0 or 1:True or False)
5  Set Column Widths (0 or 1:True or False)
6  Set Max RC (Always set to 1)
        END FILE

```

Example:

```

BEGIN FILE
"DRIVER=SQL Server;SERVER=thunder;UID=dan;APP=Visual
Basic;WSID=THUNDER;Network=DBMSSOCN;"
"Select Subject,Time1,Conc from bguidela where (Sub-
ject = '2 ')"
"0"
"0"
"1"
"1"
END FILE

```

Usage

Generally, the creation of an import file for loading should leave the connect string blank, and let the application define the connect string, since the connect string is dependent on the drivers installed on the machine. So, an import file distributed by an IT department would, potentially, contain the following:

```

BEGIN FILE
""
"Select Subject,Time1,Conc from bguidela where (Sub-
ject = '2 ')"
"0"
"0"
"1"
"1"
END FILE

```

This would allow WinNonlin Enterprise to define the connect string. When the end-user creates or loads a query, this file is loaded. At query completion,

the import could be saved to store the connect string generated by WinNonlin Enterprise. At this point a complete import file has been created.

Export file format

An Export file is an ASCII file that contains $2+3*n$ lines, where n is the number of column-field definitions. Each column-field definition is represented by three (3) consecutive lines. The definition consists of the column in the worksheet that is exported, the table field that the column is exported to, and another line to determine if this field is fixed or not (i.e., a Flag, “True” or “False”). If a field is fixed, the exported column in the worksheet is left blank.

```
BEGIN FILE
1      Connect String
2      Table Name (Table that data is exported to)
3      Worksheet Column for definition 1
4      Table Field for definition 1
5      Fixed Field Flag for definition 1
6      Worksheet Column for definition 2
7      Table Field for definition 2
8      Fixed Field Flag for definition 2
...
2+3*n-2) Worksheet Column for definition 1
2+3*n-1) Table Field for definition 1
2+3*n) Fixed Field Flag for definition 1
END FILE
```

Custom query builder interface

File: QUERYBUILDERINTERFACES.DLL

The customized query building feature allows site-specific and database-specific customization of query building. In some cases, it may be necessary to code to a specific database API set to optimize performance when determining access rights and database structure.

Current State: This ActiveX in-process server [QUERYBUILDERINTERFACES.DLL] is essentially a stubbed out API. When this DLL is registered on a machine, the developer can reference it and implement its interface. The new code will not compile unless all interfaces in the DLL are implemented.

The interfaces: There are two interfaces: one accessed by WinNonlin (IQUERYBUILDER) and one accessed by the Custom Query Builder (ICONSUMER).

IConsumer is the consumer of the imported data. The Custom Query Builder is responsible for importing data into the WinNonlin object by implementing the IConsumer interface. This object is set by WinNonlin via IQueryBuilder.

IQueryBuilder interface

The interface comprises the properties and methods in [Table 4-2](#) and [Table 4-3](#). This interface is implemented by the Custom Query Builder.

Table 4-2. Properties

Property Name	Type	Function
ConnectString	String	Connect string used by the generic ODBC query builder. The Custom Query Builder could build its own string. It is not required to.
QueryString	String	Final query string to be accessed once the custom query builder is done. This string is then passed to WinNonlin for audit purposes.
QueryBuilder-Description	String	A description of the query builder. It is also stored in the registry when the Custom Query Builder is added to WinNonlin.
QueryBuilder-Version	String	The version of the custom query builder. It is stored in the registry when the Custom Query Builder is added to WinNonlin.
QueryBuilderExt	String	The extension used by the query builder for saving import files. This enables refreshing of the data source.
QueryBuilder-ListName	String	Name to be used in the list of available Custom Query Builders.
AppHelpFile	String	This is used to set or get the help file name used by the Custom Query Builder. This should point to the exact location of the help file. Currently, the help file is located in the same directory as the DLL. When the resource manager registers the DLL, it takes the path of the DLL, the name of the DLL and appends .HLP on to the end of the name to create the help file name to be used by the Custom Query Builder. It is up to the developer of the Custom Query Builder whether to use this file is used as the help.

Table 4-3. Methods

Method Name	Type	Function
GetQuery	Long	This tells the custom query builder to get the query (i.e. build it). The return values are: <0 (Cancel), =0 (No error), >0 (Error)
GetError	String	Returns the error message if one exists.

Table 4-3. Methods (continued)

Method Name	Type	Function
ImportData-VialImportFile	Import-File as string	This function takes an import file name (usually built and saved by the Custom Query Builder), and attempts to Import data based on the contents of the Import File. This is used for scripting purposes. The structure of the import file is up to the developer. The extension of the file should be the same as returned in QueryBuilderExt.
Configure-CustomQueryBuilder	String	This is a hook to allow the Custom Query Builder Manager to configure the DLL, if necessary. For instance, for the WATSONLIMS.DLL, the owner of the Watson tables must be defined if the log on ID is not the owner. This information is stored in the registry by the WATSONLIMS.DLL. For the Integraware LIMS, the base table must be defined. This is also stored in the registry. To access this method, highlight the Custom Query Builder on the Custom Query Builder list and click Edit.
WorkSheet	wrhSht As Iconsumer	WinNonlin calls this subroutine to set the consumer object in the Custom Query Builder. The consumer object is an interface to the worksheet within WinNonlin.
Populate-WorkSheet	Long	This function is called after the GetQuery function and the WorkSheet subroutine. This tells the Custom Query Builder to populate the worksheet object handed to it in the WorkSheet subroutine. The return values are: <0 (Cancel), =0 (No error), >0 (Error)

IConsumer interface

The interface is accessed by the Custom Query Builder.

Table 4-4. IConsumer properties

Property	Type	Function
Cell (row As Long, col As Long)	Variant	Sets/Gets the value for the given row and column.
ColumnHeader(idx As Long)	String	This sets/gets the column header for the column.
Units(idx As Long)	String	Not implemented yet. Held for future use.

VB example

1. Create a new project of type ActiveX DLL.
2. Add a reference to QueryBuilderInterface.DLL
3. Create a class named CQueryBuilder. Make it public. Implement QueryBuilderInterface.IQueryBuilder. Currently, it is required that the class implement-

ing the interface is named CQueryBuilder. The calling application needs to know the name of the class to create that implements the interface.

```
Option Explicit  
Implements QueryBuilderInterface.IQueryBuilder
```

4. Now each required method and property can be defined as illustrated below.

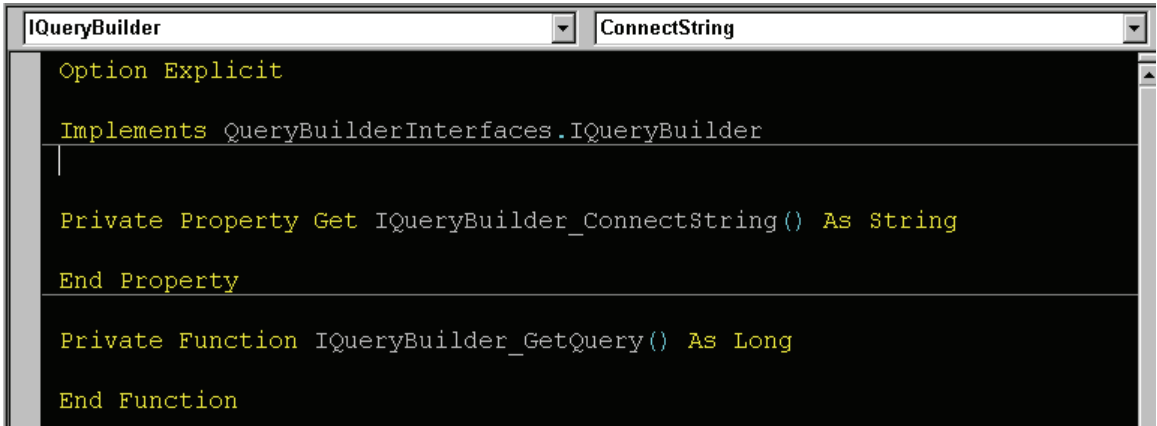
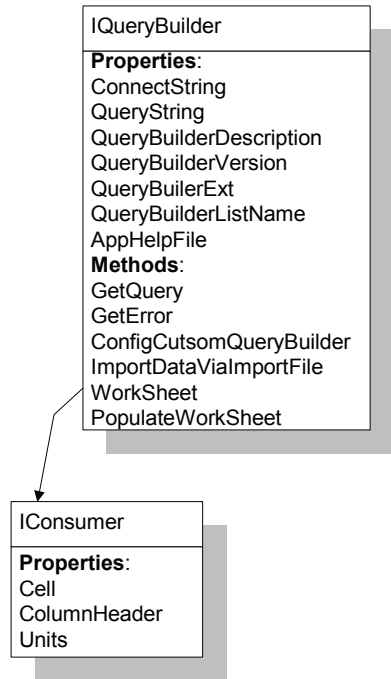
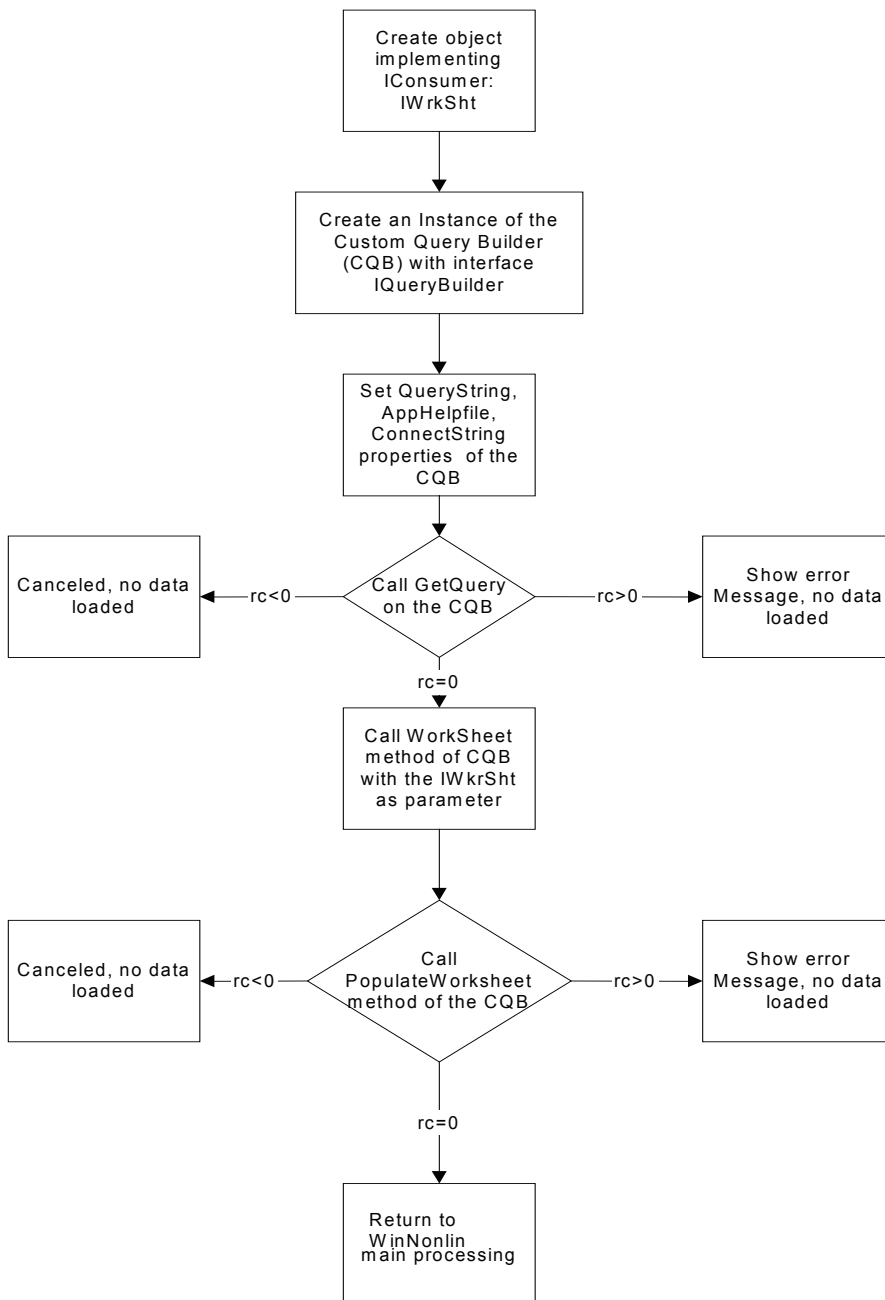


Figure 4-1. Since *IQueryBuilder* is implemented, the object list has *IQueryBuilder*, and the other list has its properties and methods. If the selected property/method is not defined, stubbed code is generated.

5. Next, finish the code required to generate a valid SQL Query. A Simple Object Diagram is shown to the right.



Typical usage of the Custom Query
Builder (CQB)

Charts

Using the Chart Wizard and Chart Designer to create and format plots, and other chart options

WinNonlin provides high quality plots of input data and modeling or NCA output. There are three means of creating and editing plots.

1. Chart Wizard: Use this wizard to plot data in any worksheet as a XY plot, bar chart, or histogram. For additional plot types, create a plot and edit it in the Chart Designer (below). See [“Chart Wizard” on page 115](#).
2. Modeling output: Use the Model Options to include chart output from compartmental modeling or noncompartmental analysis. See [“Output options” on page 206](#).
3. Chart Designer. Modify type and formatting of existing charts in the Chart Designer. The Chart Designer supports additional plot types not available in the Chart Wizard. See [“Chart Designer” on page 123](#).

See the following for additional chart options.

- [“Frequently used chart formatting options” on page 124](#)
- [“Chart templates” on page 129](#)
- [“Copy and paste” on page 130](#)

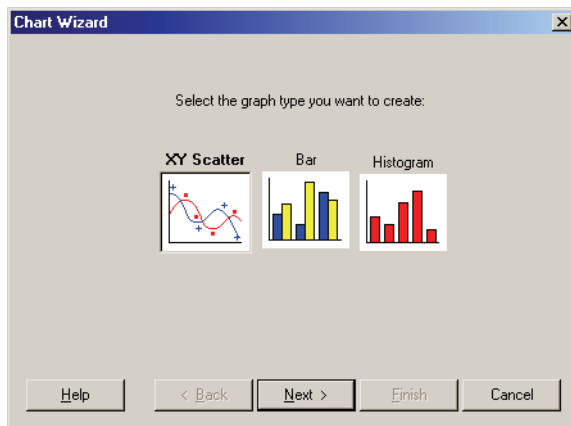
WinNonlin can save charts to many different file formats. See [“File types” on page 14](#) for details. Information on printing charts appears under [“Printing” on page 21](#).

Chart Wizard

Use the Chart Wizard to plot worksheet data. You can then change the plot type and formatting using the Chart Designer as detailed under [“Chart Designer” on page 123](#).

To create a plot:

1. Open the worksheet(s) containing data to plot. The workbook must not be hidden. The worksheet must be the active worksheet in the workbook window.
2. Click the **Chart Wizard** tool bar button or choose **Tools>Chart Wizard** from the WinNonlin menus. The Chart Wizard appears.



3. Choose a chart type.

Note: Additional chart types are available after a plot is created. Make a scatter plot, bar chart, or histogram with the desired variables, then use the Chart Designer to change the plot type. See “[Chart Designer](#)” on page 123.

4. Click the **Next** button to proceed to the **X-Y variables**.

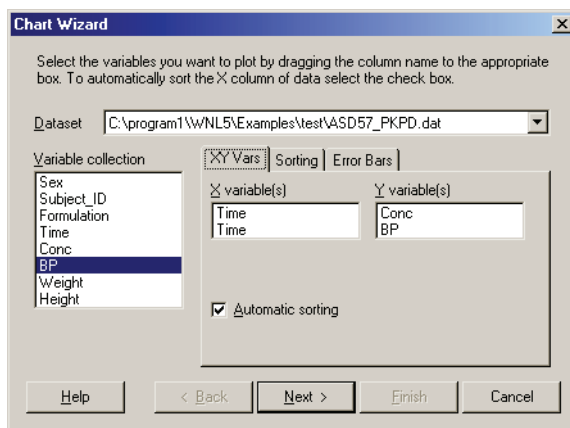
X-Y variables

The X-Y variables tab of the [Chart Wizard](#) sets the independent and dependent variables to be plotted. Multiple pairs of XY variables can be included, for overlay plots. These settings differ for [Scatter plots and bar charts](#) versus [Histograms](#).

Scatter plots and bar charts

To select variables for a scatter plot or bar chart:

1. Make sure that the workbook to be plotted is displayed in the Data Set field (shown below). The columns in the active worksheet appear under Variable Collection.
2. To specify each X or Y variable, highlight a variable under Variable Collection and drag it to the X or Y Variable box.



3. To create an overlay chart, showing more than one pair of X and Y variables:
 - a. If all variables are in one workbook, drag paired sets of X and Y variables to the appropriate variable boxes, so that pairs are listed side-by-side.
 - b. If the variables come from more than one workbook, select a second workbook under Data Set to select X and Y variables from that data set.
4. (Scatter plots only) check **Automatic Sorting** to sort the data in ascending X-value order before drawing plot lines. This option would be inappropriate for a hysteresis curve. [Figure 5-1](#) below provides an illustration of a simple data set plotted with and without automatic sorting.

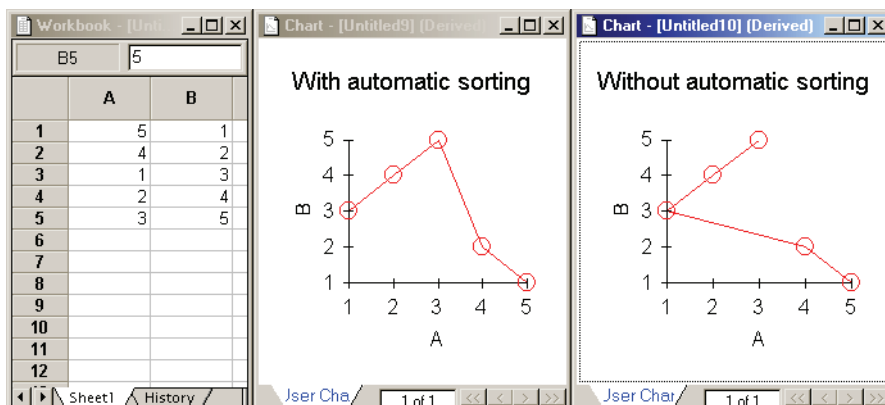


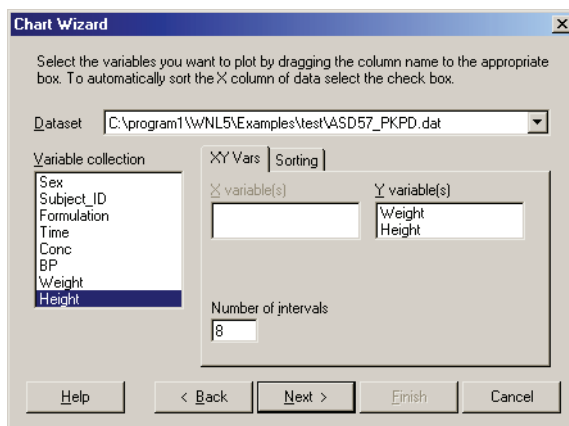
Figure 5-1. Plotting data with (middle) and without (right) automatic sorting

5. To create multiple plots or multiple series within a plot, click the Sorting tab and follow the directions under “Sort and group variables” on page 119.
6. (Scatter plots only) To set up error bars, open that tab of the dialog and see “Error bars” on page 120.
7. Click **Next** to proceed to the **Chart titles**.

Histograms

To select variables for a histogram:

1. Make sure that the workbook to be plotted is displayed in the Data Set field. The columns in the active worksheet are listed in the Variable Collection list.
2. To specify the variable(s) for which frequencies will be displayed, highlight a variable under Variable Collection and drag it to the Y Variable box. Repeat to add variables on the same set of axes.
3. Enter the number of bars for the X axis (optional) under Number of Intervals.



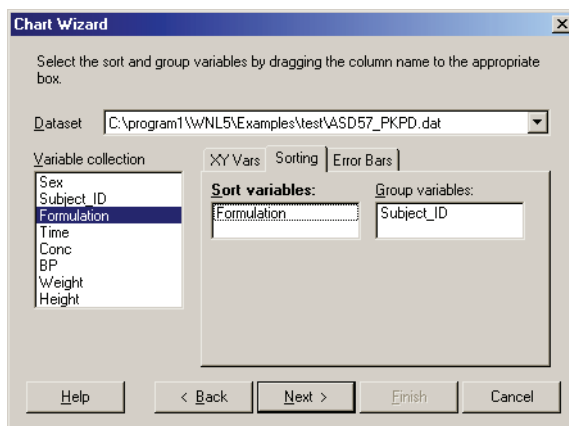
WinNonlin will determine the range of the Y variable, divide this range into the number of intervals, then count the number of observations that fall into each interval, i.e., the frequency for each interval.

4. To create multiple plots, click the Sorting tab and follow the directions under “Sort and group variables” on page 119.
5. Click **Next** to proceed to the Chart titles.

Sort and group variables

To sort data into multiple plots, or group data series within a plot:

1. Open the **Sorting** tab of the Chart Wizard.



2. Drag and drop variables from the Variable Collection list to the Sort or Group variables box:

- *Sort variable(s)*: a separate plot (set of axes) will be created for each unique combination of sort variable values.
- *Group variable*: each plot will contain a separate line (scatter plot) or set of bars (bar charts) for each unique combination of group variable values.

The example illustrated above will generate one plot for each drug formulation, with an individual line (scatter) or set of bars (bar chart) for each subject.

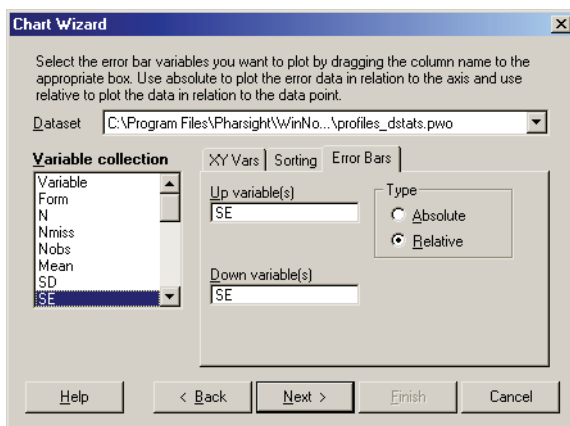
3. (Scatter plots only) To set up error bars, open that tab of the dialog and see “Error bars” on page 120.

4. Click **Next** to proceed to the **Chart titles**.

Error bars

To add error bars to a scatter plot:

1. On the Error Bars tab of the **Chart Wizard** and set the following:
 - a. *Up variable*: error values in the upward direction (Y-axis increase).
 - b. *Down variable*: error values in the downward direction (Y-axis decrease).
 - c. *Relative error bars*: error values are expressed as absolute difference from the mean; that is, each error value is added to (up variable) or subtracted from (down variable) the corresponding datum.
 - d. *Absolute error bars*: Error data are expressed as Y-axis co-ordinates for the top (up variable) or bottom (down variable) of each error bar.



CAUTION: In overlay charts (more than one set of X-Y variables per chart), the selection of absolute versus relative error bars applies to all data series in the chart.

The example above uses relative error bars for a plot of average concentrations plus or minus the standard error at each time point. In this case, the standard error supplies both the Up and Down variables.

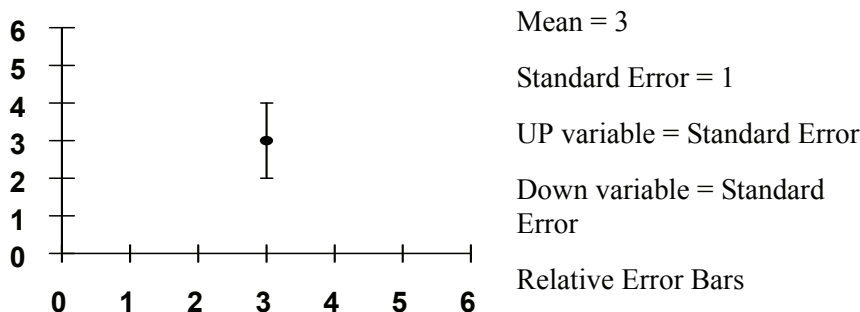


Figure 5-2. Plotting mean and standard error using relative error bars

Absolute error bars are useful for plotting more than one value at each X-axis point, e.g. minimum, mean and maximum, as illustrated below.

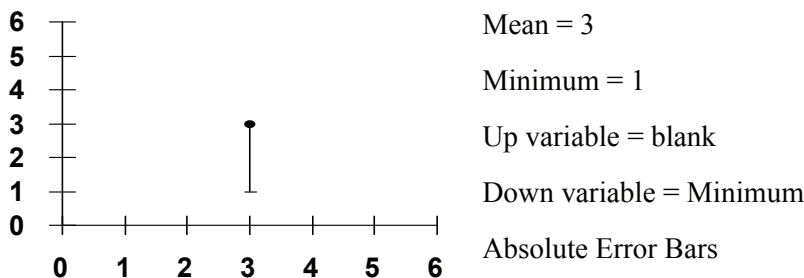
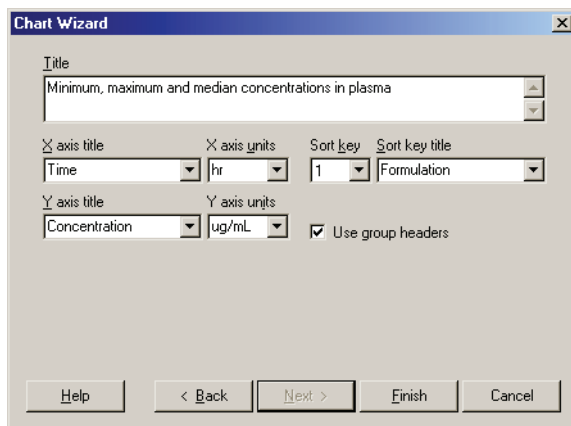


Figure 5-3. Plotting mean and minimum values using absolute error bars

2. Click **Next** to proceed to the [Chart titles](#).

Chart titles



1. To include a title at the top of each chart, enter it in the Title field in the final dialog of the [Chart Wizard](#).
2. Select or enter the other options:
 - *Axis titles* by default use the variable names from the data set. To change an axis title, select and replace the text in the appropriate axis title field.
 - *Units*: Enter or edit the units for the axis labels. See “[Units](#)” on [page 31](#) to select whether column units are displayed as axis units by default.
 - *Sort key title*: If sort variables are used the *sort key title* appears below the plot title to identify which value of the sort variable(s) appears in the plot, e.g., “Formulation = Tablet.” Edit the display name of each sort variable (in this case, Formulation) by selecting its number in the *sort key* field.
 - *Use group headers*: Check this option to include the group variable name(s) in the chart legend, e.g., “Subject = DW.”
 - *Legend*: When a chart includes more than one pair of X and Y variables, the Legend field provides a means to edit the text that will identify data from each data set in the chart legend.
3. Click **Finish** to create the chart.

Axis options

Once a plot is created using the [Chart Wizard](#) or as part of modeling output, adjust the X and Y axes as follows. See also “[Chart Designer](#)” on [page 123](#).

To set axis options

1. Choose **Chart>Axis Options** from the WinNonlin menus to open the Axis Options dialog.
2. Select the appropriate options from the following.
 - *Uniform Axis*: If a sort key was used to identify multiple profiles, this optional applies the same axis range and divisions to all profiles.
 - *Logarithmic*: Check this box to plot on a natural-log scale.
 - *Intervals*: For a histogram, enter the number of intervals under Intervals.
3. Click **OK**.

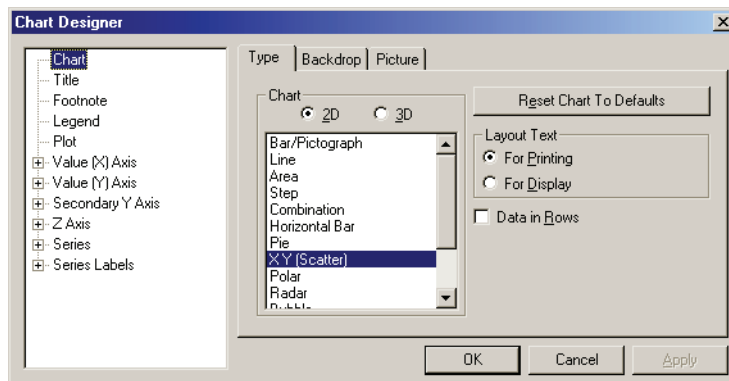
Chart Designer

Use the Chart Designer to format and enhance charts created using the Chart Wizard (see “[Chart Wizard](#)” on page 115) or as part of modeling or NCA output. The Chart Designer provides extensive plot formatting options and a selection of additional chart types, including three-dimensional plots.

See the Chart Designer online help system for detailed usage instructions.

To use the Chart Designer:

1. With a chart in the active window, choose **Chart>Chart Designer** from the WinNonlin menus, or double click on the chart or a specific plot element. Alternately, right-click on the chart and choose **Chart Designer** from the shortcut menu.



The Chart Designer provides a tree view in the left pane. Click an + symbol to expand an element to view sub-elements. The tabs to the right change as different elements are selected in the tree.

When an entire group of elements is selected from the tree view—all chart series, for example—the controls to the right apply to all of the selected items.

2. Select an item or item(s) to format from the tree view, and choose settings to the right. See the Chart Designer Help or [“Frequently used chart formatting options” on page 124](#) for additional information.
3. Click **Apply** to execute the changes or **OK** to execute the changes and close the Chart Designer.

Frequently used chart formatting options

This section highlights some commonly-used chart formatting options:

- [“Changing the plot type” on page 124](#)
- [“Changing line style and markers” on page 125](#)
- [“Changing axis scaling” on page 125](#)
- [“Creating a semi-log plot” on page 126](#)
- [“Setting grid lines” on page 127](#)
- [“Removing or adding walls” on page 128](#)

See also the Chart examples in the *Examples Guide*.

Changing the plot type

To select the type of plot:

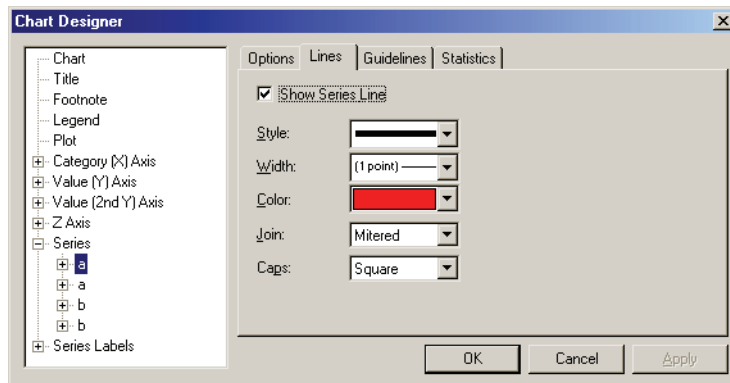
1. In the Chart Designer, click **Chart** at the top of the tree view.
2. Select the Type tab on the right side of the Chart Designer.
3. Select the **2D** radio button for two-dimensional plots; **3D** for three-dimensional.
4. Select the chart type from those available under Chart. See the Chart Designer Help for details and data requirements for each chart type.

Changing line style and markers

Chart colors, line styles, and markers (i.e., symbols) can be changed in the Chart Designer. The formatting may be applied to all data series or an individual series in the chart.

To format series lines:

1. To select a particular line for formatting, do one of the following.
 - a. Double-click on that line in the chart. The Chart Designer opens to the appropriate tab, or
 - b. With the chart in the active window choose **Chart>Chart Designer** from the WinNonlin menus. The Chart Designer opens. Then click **Series** in the tree view for all series, or use the + to open the list of series and select one.
2. Click the Lines tab.



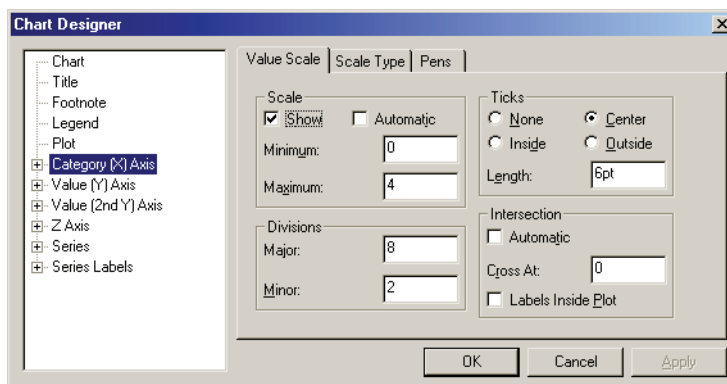
3. To hide the line, uncheck the **Show Series Line** check box.
4. Otherwise, select the desired line style, width or color.
5. To set how line segments join at data points use the Join menu.
6. To specify how the ends of lines are displayed select from the menu for Caps.
7. Click **Apply** or **OK**.

Changing axis scaling

Charts use automatic scaling by default. As a result, axis scales may change when the chart window is resized. The axis scaling can be set to a fixed scale.

To change the scale of an axis:

1. Double-click on the axis or choose **Chart>Chart Designer** from the WinNonlin menus. The Chart Designer opens.
2. Select the appropriate axis in the tree view pane.
3. To change the scale sizes use the **Value Scale** tab.
4. Make sure that the **Automatic** check box is unchecked to set a fixed scale.
5. Enter the minimum and maximum value to appear on the axis.
6. Enter the total number of major divisions.
7. Enter the number of minor divisions per major division.



This example creates an X axis that ranges from 0 to 4, with a major division at every multiple of 0.5, and a minor division at every quarter.

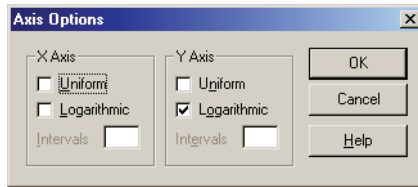
8. Click **Apply** or **OK**.

Creating a semi-log plot

A semi-log plot can be created quickly by using the Axis Options dialog or Chart Designer. Use the Chart Designer to set the log base.

To create a semi-log plot setting the Axis Options:

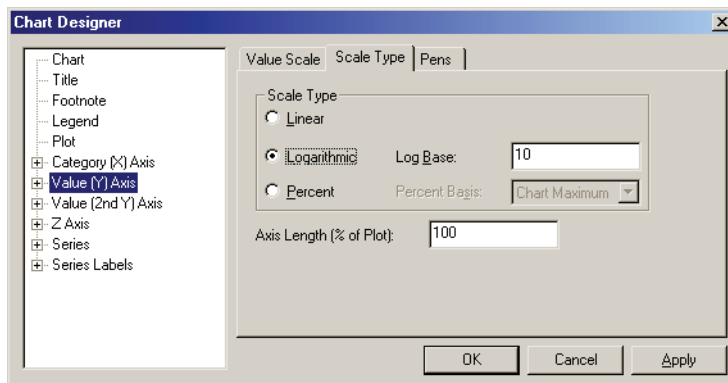
1. Select **Axis Options** from the **Chart** menu. The Axis Options dialog appears.
2. Select the **Logarithmic** check box for the Y-axis.



3. Click **Apply** or **OK**.

To set the log base using the Chart Designer:

1. Open the chart and double-click on the Y axis or right-click on the chart and choose **Chart Designer** from the pop-up menu.
2. Select the Y axis in the tree view as shown below.
3. Open the Scale Type tab to the right.
4. Select the **Logarithmic** check box and enter the desired log base.



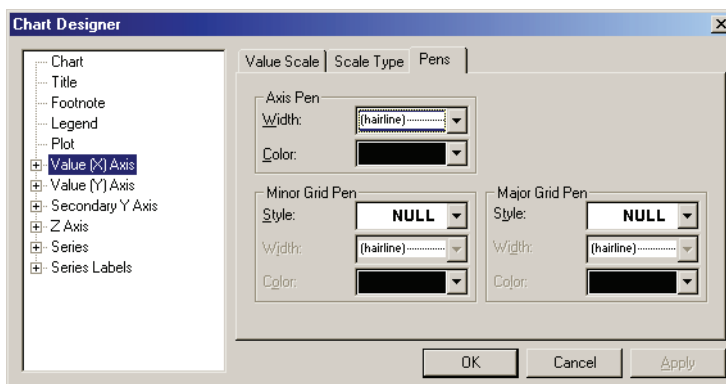
5. Click **Apply** or **OK**.

Setting grid lines

To remove the grid lines from a chart:

1. Open the chart and double-click on the Y axis or right-click on the chart and choose **Chart Designer** from the pop-up menu.
2. Select the X-axis in the tree view.

3. On the Pens tab change the Major Grid Pen and Minor Grid Pen Style to the desired line style. Use NULL to show no grid lines.
4. Click **Apply**.



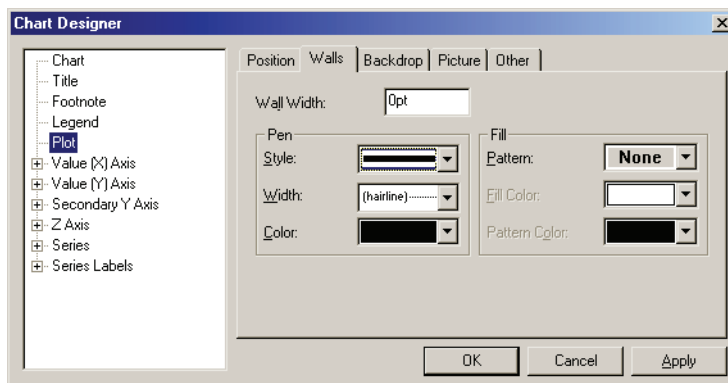
5. Repeat for the Y-axis.
6. Click **Apply** or **OK**.

Removing or adding walls

Chart walls form a border at the maximum X and Y values, enclosing the chart in a rectangle.

To set chart walls:

1. Double-click on the chart or right-click on the chart and choose **Chart Designer** from the pop-up menu.
2. Select **Plot** from the tree view menu.
3. Select the **Walls** tab.



4. In the Pen group box, select the line style or change it to NULL for none.
5. Click **Apply** or **OK**.

Chart templates

WinNonlin provides the option to create *chart templates* containing customized settings, including chart type and formatting set in the Chart Designer.

To save a chart template:

1. Use the **Chart Wizard** or **Chart Designer** to create and format a plot with all the options to be saved in the template.
2. With the chart window selected, choose **Chart>Save Template** from the WinNonlin menus.
3. Set the file location and name. Save the file with the extension **.GTP**.
4. Click **Save**.

To load a graph template:

1. Create a chart with the desired content. The chart must contain the same number of X, Y, sort, group and error variables as the template.
2. With the chart selected, choose **Chart>Load Template** from the WinNonlin menus.
3. The Load Template dialog appears. Select the location and file that contains the graph template (*.GTP).

4. Click **Open**.

The graph is now updated with the formatting established in the template.

Note: If a template that contains formatting for one data series is applied to a chart that contains multiple series, the template is applied only to the first series.

Copy and paste

WinNonlin provides the capability to copy a graph paste it into another Windows application in Windows Metafile (*.WMF) format.

Note: The Export to Word tool bar button exports the active object directly to Microsoft Word, and can be used to export an image of the current chart.

To copy and paste a chart:

1. Select the chart window containing the chart to copy.
2. Select **Copy** from the **Edit** menu. A copy of the chart image is placed on the clipboard in Windows Metafile (*.WMF) format.
3. Switch to the other application and (depending upon which application is being used), use the **Paste** or **Paste Special** function to paste the image.

Note: In order to support cut and paste, the Windows Character Map is limited to the 255 characters noted in the Windows 98 ANSI version. Windows XP has multiple Character Mapping options (use Programs>Accessories>System Tools>Character Map in the Windows Start menu to view them). Windows: Western must be selected as the default option in order to cut and paste the characters supported in WinNonlin. Look to the lower, right corner of the Character Map to see if “Keystroke: Alt+XXXX” is within the 255 range for any special characters (characters not included on the computer keyboard).

Tables

Creating formatted tables in WinNonlin

The Table Wizard generates a report- and analysis-ready table of worksheet data and (optional) summary statistics in a new WinNonlin workbook. Nine table templates provide a choice of layouts and summary statistics, with flexible formatting options. See the following for instructions and options.

- “Table Wizard” on page 131
- “Summary statistics” on page 144
- “Significant digits and column alignments” on page 145
- “Table formatting” on page 146
- “Table definition files” on page 148

Table Wizard

Follow these steps to create a new workbook containing a formatted table, with or without calculating summary statistics.

1. Open one or two data sets to provide table data. Make sure that the desired data are selected as the active worksheet in the workbook(s).
2. Select **Tools>Table Wizard** from the WinNonlin menus or click the **Table Wizard** tool bar button to open the Table Wizard.
3. Select a table template from those listed in [Table 6-1](#). See “WinNonlin table templates” on page 134 for details and examples of each template.
4. A sample of the selected template appears at the top of the Table Wizard. Click on the sample to expand it; click on it again return to the Table Wizard.



Table 6-1. Table templates

Template	Description
Template 1	Summary statistics for variables in one data set.
Template 2	Raw data from one data set.
Template 3	Raw data and summary statistics for one data set.
Template 4	Summary statistics for one data set by cross variable values.
Template 5	Raw data from one data set by cross variable values.
Template 6	Raw data and summary statistics from one data set by cross variable values.
Template 7	As template 6, with columns ordered by cross variable within variable.
Template 8	As template 6, with summary statistics in columns instead of rows.
Template 9	As template 6, joining data from two workbooks.

Note: Template 9 is available only when at least two workbooks are open.

5. Select the dataset to be used from the drop-down list near the lower left corner of the Wizard. For template 9, select the workbook containing raw data in the left list, and the workbook containing computed values in the right list.
6. Assign variables: Drag the variables to be included from the list at the bottom left to the appropriate boxes to the right. For template 9, set the join variables as described under “[Joining data sets](#)” on page 143.
7. Choose **File>Options** from the Table Wizard menus to set options for page breaks between group variable values (optional). By default, group variables appear within the table, demarcating the separate data groupings, as shown for the variable Form below.

Form	Subject	Time	Conc
Capsule	1	0.00	0.00
	1	5.00	340.32
	1	10.00	1914.00
	1	15.00	2069.17
	1	20.00	1470.92

Alternately, each value of the group variable(s) can generate a page break, with the group variable within the table or above it as shown below. See also “[Tables](#)” on page 31 for details on each table option.

Form= Capsule

Subject	Time	Conc
1	0.00	0.00
1	5.00	340.32
1	10.00	1914.00
1	15.00	2069.17
1	20.00	1470.92

A button will appear to the left of each group variable in the Table Wizard, indicating the page break settings. If the button contains an X, a page break will be inserted when the value of that group variable changes. To toggle the setting for a specific variable, click the button next to that variable.

8. Click **Next**.
9. If the template includes statistics, go to [“Summary statistics” on page 144](#).
10. Set significant digits as described under [“Significant digits and column alignments” on page 145](#).
11. Set formatting as detailed under [“Table formatting” on page 146](#).
12. Click **Finish** to create a preview of the table.
13. Click the **Prev** button to modify table settings if needed. Click **Create** to create the table in a new WinNonlin workbook.

Missing and excluded data

If a value needed for any statistical calculation is missing, the table cell designated for the result will be left blank. For example, when $N = 1$ the SD cannot be calculated. As a result, the cell that would contain SD is blank in the resulting table. (The representation of missing values is set in the Tools Options dialog, on the Workbook tab. See [“Workbooks” on page 28](#).)

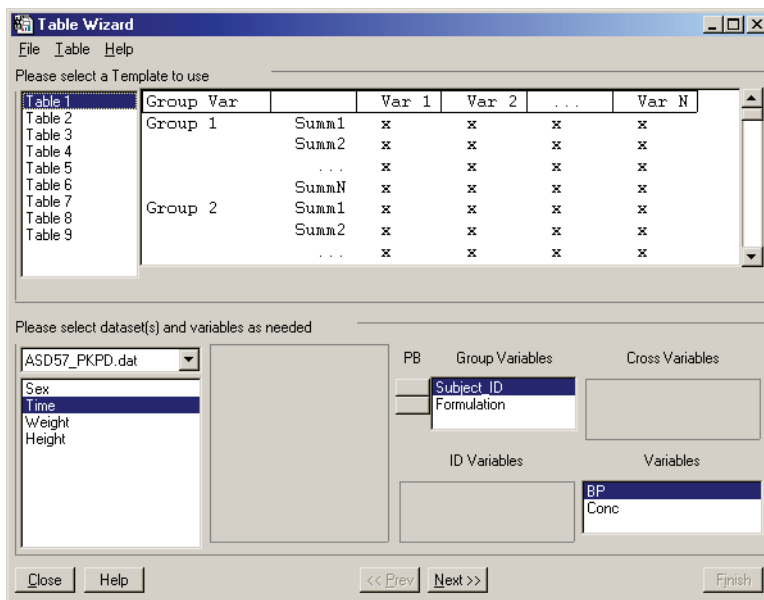
Excluded values in the data set are treated as missing in the calculation of summary statistics. If an entire row is excluded, that row will not appear in the table.

WinNonlin table templates

The [Table Wizard](#) provides the following nine templates with different layouts and summary statistics usage. Template 9 joins data from two workbooks.

Template 1

This template computes and displays summary statistics for each column included under Variables. The statistics are computed separately for each unique combination of values for the group variable(s), if any.

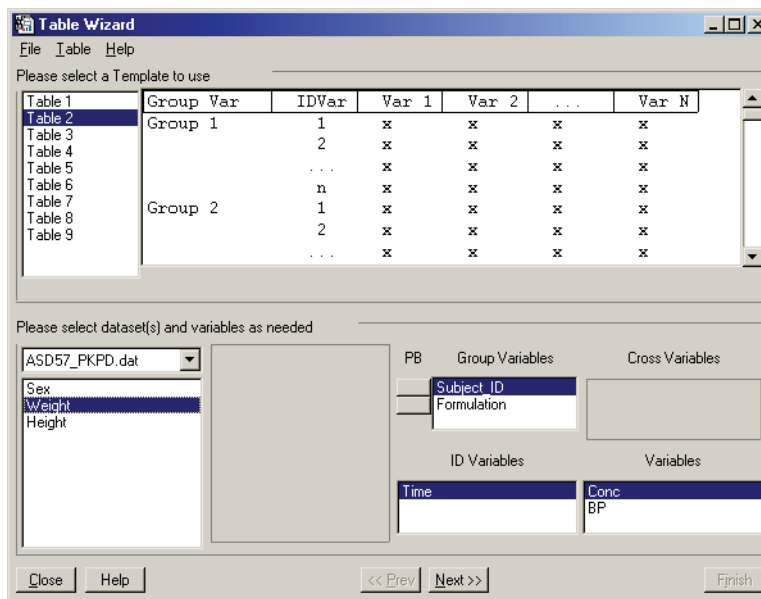


This example computes the minimum, median and maximum concentration and blood pressure for each subject, for each drug formulation.

Table - [Untitled15] (Derived)							
F10							
	A	B	C	D	E	F	G
1	Subject ID	Formulation		Conc	BP		
2	DW	Capsule	Min	1.39	170.00		
3			Median	4.10	173.50		
4			Max	6.20	181.00		
5		Tablet	Min	1.22	169.79		
6			Median	3.93	173.17		
7			Max	6.06	181.00		
8	GS	Capsule	Min	1.29	182.00		
9			Median	3.70	189.50		

Template 2

This template lists raw data for each column selected under Variables. It does not generate summary statistics. Data are sorted alphanumerically by group variable values, then by ID variable values, in the order listed.



This example lists concentration and blood pressure measurements for each subject, for each drug formulation, at each time in a crossover study.

Table - [Untitled19] (Derived)

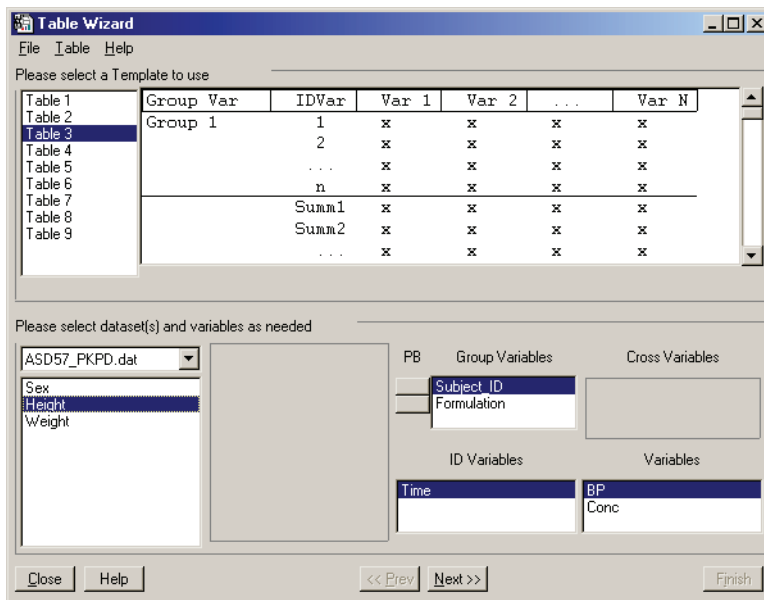
E20 172

	A	B	C	D	E	F	G
1	Subject_ID	Formulation	Time	Conc	BP		
2	DW	Capsule	0.00	2.50	170		
3			0.25	4.92	171		
4			0.50	6.20	172		
5			0.75	6.07	171		
6			1.00	6.05	172		
7			1.50	5.21	171		
8			2.00	4.41	172		
9			2.50	4.30	173		
10			3.00	4.19	174		
11			4.00	4.01	175		
12			5.00	3.30	176		
13			6.00	3.10	177		
14			8.00	3.08	178		
15			12.00	2.84	179		
16			14.00	2.48	180		
17			24.00	1.39	181		
18		Tablet	0.00	2.38	170		

Sheet1 History

Template 3

Template 3 adds summary statistics to template 2. It lists the raw data for each variable at each value of the ID variable(s), and summarizes it for each unique combination of group variable values, if any. The example below lists concentrations and blood pressure measurements at each time, and summarizes those values for each subject and formulation.



	A	B	C	D	E	F	G
1	Subject_ID	Formulation	Time	Conc	BP		
2	DW	Capsule	0.00	2.50	170		
3			0.25	4.92	171		
4			0.50	6.20	172		
5			0.75	6.07	171		
6			1.00	6.05	172		
7			1.50	5.21	171		
8			2.00	4.41	172		
9			2.50	4.30	173		
10			3.00	4.19	174		
11			4.00	4.01	175		
12			5.00	3.30	176		
13			6.00	3.10	177		
14			8.00	3.08	178		
15			12.00	2.84	179		
16			14.00	2.48	180		
17			24.00	1.39	181		
18			Min	1.39	170		
19			Max	6.20	181		
20							
21		Tablet	0.00	2.38	170		

Template 4

Like template 1, this template generates summary statistics and displays them without the raw data. It computes separate statistics for each unique combination of group and cross variable values.

Table Wizard

File Table Help

Please select a Template to use

Table 1	Table 2	Table 3	Table 4	Table 5	Table 6	Table 7	Table 8	Table 9

Please select dataset(s) and variables as needed

ASD57_FKPD.dat

Sex
Time
Weight
Height

PB Group Variables Cross Variables

Subject_ID Formulation

ID Variables Variables

Conc
BP

Close Help << Prev Next >> Finish

This example reports minimum, median and maximum concentration and blood pressure for each subject and each drug formulation. Subjects appear in separate table rows with each formulation in a separate set of columns.

	A	B	C	D	E	F	G
1			Formulation				
2			Capsule		Tablet		
3	Subject_ID		Conc	BP	Conc	BP	
4	DW	Min	1.39	170	1.22	170	
5		Median	4.10	174	3.93	173	
6		Max	6.20	181	6.06	181	
7	GS	Min	1.29	182	1.12	182	
8		Median	3.70	190	3.53	189	

Template 5

This template reports raw data for each column in the Variables list, at each ID variable value. A separate listing is included for each unique combination of values for the group and cross variables, with cross variable values in adjacent columns.

Table Wizard

Please select a Template to use

Table 1	Table 2	Table 3	Table 4	Table 5	Table 6	Table 7	Table 8	Table 9

Please select dataset(s) and variables as needed

ASD57_FKPD.dat

Sex
Weight
Height

PB Group Variables Cross Variables

Subject_ID Formulation

ID Variables Variables

Time Conc
BP

Close Help << Prev Next >> Finish

This example reports raw concentration and blood pressure data for each subject at each time, with the two formulations in adjacent sets of columns.

	A	B	C	D	E	F	G
1			Formulation				
2			Capsule		Tablet		
3	Subject ID	Time	Conc	BP	Conc	BP	
4	DW	0.00	2.50	170	2.38	170	
5		0.25	4.92	171	4.75	171	
6		0.50	6.20	172	6.06	172	
7		0.75	6.07	171	5.90	171	
8		1.00	6.05	172	5.90	172	
9		1.50	5.21	171	5.09	171	
10		2.00	4.41	172	4.24	172	
11		2.50	4.30	173	4.13	173	
12		3.00	4.19	174	4.02	174	
13		4.00	4.01	175	3.84	175	
14		5.00	3.30	176	3.13	176	
15		6.00	3.10	177	2.93	177	
16		8.00	3.08	178	2.91	178	
17		12.00	2.84	179	2.67	179	
18		14.00	2.48	180	2.31	180	
19		24.00	1.39	181	1.22	181	
20	GS	0.00	1.60	182	1.43	182	

Template 6

This template adds computation of summary statistics to the listing of raw data as presented in template 5. Statistics are computed separately for each unique combination of group and cross variable values.

Table Wizard

File Table Help

Please select a Template to use

Table 1	Table 2	Table 3	Table 4	Table 5	Table 6	Table 7	Table 8	Table 9

Please select dataset(s) and variables as needed

ASD57_FKPD.dat

Sex
Weight
Height

Group Variables: Subject ID, Formulation

ID Variables: Time, Conc, BP

Close Help << Prev Next >> Finish

Table - [Untitled27] (Derived)						
F26		184				
	A	B	C	D	E	F
1	Formulation					
2	Capsule			Tablet		
3	Subject_ID	Time	Conc	BP	Conc	BP
4	DW	0.00	2.50	170	2.38	170
5		0.25	4.92	171	4.75	171
6		0.50	6.20	172	6.06	172
7		0.75	6.07	171	5.90	171
8		1.00	6.05	172	5.90	172
9		1.50	5.21	171	5.09	171
10		2.00	4.41	172	4.24	172
11		2.50	4.30	173	4.13	173
12		3.00	4.19	174	4.02	174
13		4.00	4.01	175	3.84	175
14		5.00	3.30	176	3.13	176
15		6.00	3.10	177	2.93	177
16		8.00	3.08	178	2.91	178
17		12.00	2.84	179	2.67	179
18		14.00	2.48	180	2.31	180
19		24.00	1.39	181	1.22	181
20		Min	1.39	170	1.22	170
21		Median	4.10	174	3.93	173
22		Max	6.20	181	6.06	181
23						
24	GS	0.00	1.60	182	1.43	182

Template 7

This template presents the same information as template 6 with the columns re-ordered. In template 6, the regular variable columns are grouped together within values of the cross variable. In template 7, all cross variable values are grouped together, with one set for each regular variable.

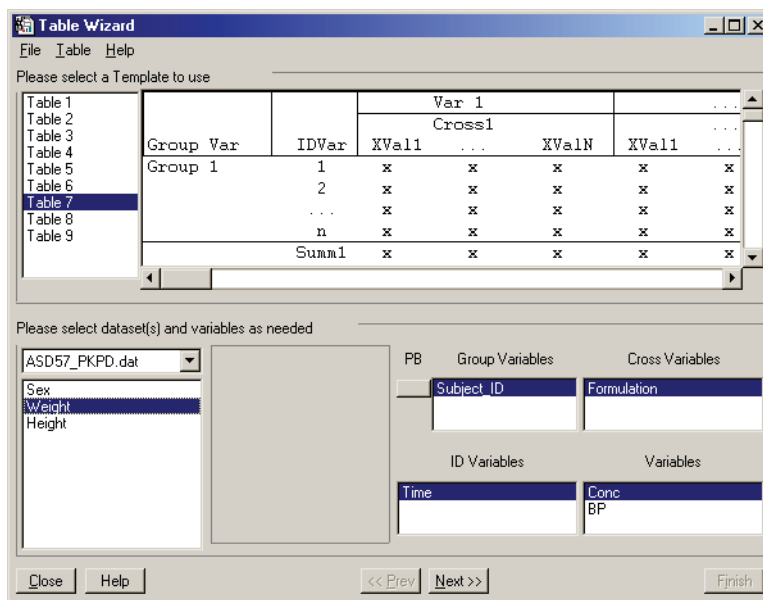


Table - [Untitled29] (Derived)							
F26		184					
	A	B	C	D	E	F	
1			Conc		BP		
2			Formulation		Formulation		
3	Subject ID	Time	Capsule	Tablet	Capsule	Tablet	
4	DW	0.00	2.50	2.38	170	170	
5		0.25	4.92	4.75	171	171	
6		0.50	6.20	6.06	172	172	
7		0.75	6.07	5.90	171	171	
8		1.00	6.05	5.90	172	172	
9		1.50	5.21	5.09	171	171	
10		2.00	4.41	4.24	172	172	
11		2.50	4.30	4.13	173	173	
12		3.00	4.19	4.02	174	174	
13		4.00	4.01	3.84	175	175	
14		5.00	3.30	3.13	176	176	
15		6.00	3.10	2.93	177	177	
16		8.00	3.08	2.91	178	178	
17		12.00	2.84	2.67	179	179	
18		14.00	2.48	2.31	180	180	
19		24.00	1.39	1.22	181	181	
20		Min	1.39	1.22	170	170	
21		Median	4.10	2.00	174	173	

Template 8

Most templates present raw data in columns. Template 8 presents raw data in rows with summary statistics in the right-most columns. Statistics are calculated for each ID variable value, within group and cross variable values.

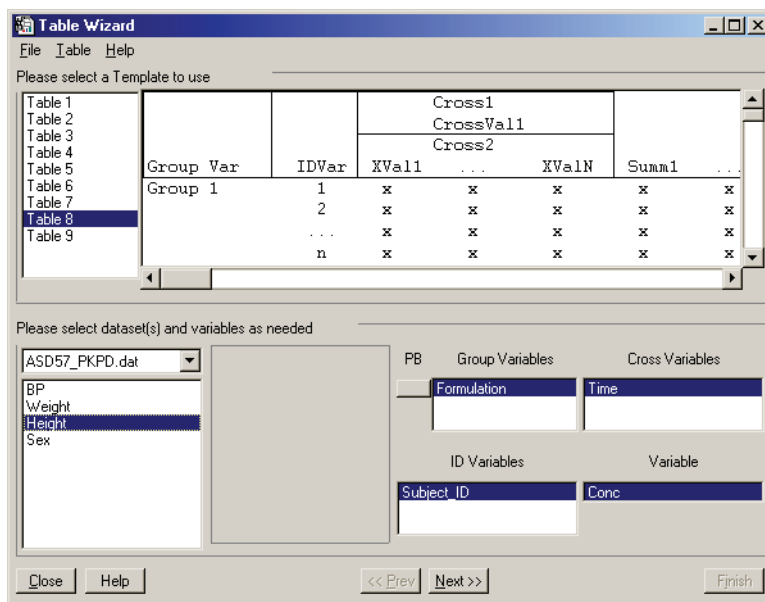


Table - [Untitled31] (Derived)											
C3		Time									
	A	B	M	N	O	P	Q	R	S	T	
1	Variable = Conc										
2											
3	Time										
4	Formulation	Subject_ID	5.00	6.00	8.00	12.00	14.00	24.00	Min	Max	
5	Capsule	DW	3.30	3.10	3.08	2.84	2.48	1.39	1.39	6.20	
6		GS	3.50	3.21	3.08	2.74	2.38	1.29	1.29	6.47	
7		GW	3.30	3.10	3.08	2.84	2.48	1.39	1.39	6.20	
8		MC	4.30	3.60	3.30	3.10	2.70	2.40	2.40	5.10	
9		RH	3.70	3.50	3.38	2.84	2.38	1.29	1.29	6.60	
10	Tablet	DW	3.13	2.93	2.91	2.67	2.31	1.22	1.22	6.06	
11		GS	3.33	3.04	2.91	2.57	2.21	1.12	1.12	6.30	

Template 9

Template 9 joins data from the active worksheet in each of two workbooks. For example, it can combine raw PK data from an input workbook with PK parameter estimates from a modeling output workbook. The template is designed to present raw data (aggregate variable) at different sample times (DS1 cross variable) and PK parameters (DS2 variables) in columns, with a row for each subject (ID variable). It generates summary statistics for each unique combination of group and ID variable values. The two data sets are “joined” on these group and ID variable values.

The workbook containing raw data must be selected from the left drop-down list under “Please select datasets...” in the wizard. The workbook holding PK parameters or other statistics must be selected from the right drop-down list.

Table Wizard

File Table Help

Please select a Template to use

Table	Group	Var	IDVar	0.0	0.5	...	24.0	AUC
Table 1	Group 1	1	x	x	x	x	x	x
Table 2		2	x	x	x	x	x	x
Table 3		...	x	x	x	x	x	x
Table 4		n	x	x	x	x	x	x
Table 5		Summ1	x	x	x	x	x	x

Please select dataset(s) and variables as needed

ASD57_FKPD.dat PK Workbook - [Untitled:1] PB Group Variables DS1 - Cross Variables

Sex BP Weight Height K01_CV% K10_CV% K01_UnivarCI_Lower K01_UnivarCI_Upper K01_PlanarCI_Lower K01_PlanarCI_Upper K10_UnivarCI_Lower K10_UnivarCI_Upper K10_PlanarCI_Lower K10_PlanarCI_Upper

Edit Group Var Joins Formulation - Formulation Edit ID Var Joins Subject ID - Subject ID

DS1 - Aggregate Var: Conc

Close Help << Prev Next >> Finish

Table - [Untitled34] (Derived)											
C2	A	B	L	M	N	O	P	Q	R	S	T
1	Aggregate = Conc										
2											
3	Time										
4	Formulation	Subject_ID	4.00	5.00	6.00	8.00	12.00	14.00	24.00	V_F	V_F_StdError
5	Capsule	DW	4.01	3.30	3.10	3.08	2.84	2.48	1.39	8.71	0.66
6		GS	3.61	3.50	3.21	3.28	2.74	2.38	1.29	9.22	0.60
7		MC	5.10	4.30	3.60	3.30	3.10	2.70	2.40	10.07	1.00
8		RH	4.01	3.70	3.50	3.38	2.80	2.32	1.24	8.32	0.62
9	Tablet	Mean	4.150	3.618	3.304	3.186	2.875	2.486	1.550	9.007	0.708
10		SE	0.250	0.186	0.104	0.065	0.059	0.058	0.214	0.303	0.074
11											
12		DW	3.84	3.13	2.93	2.91	2.67	2.31	1.22	8.89	0.68
13	Tablet	GS	3.44	3.33	3.04	2.92	2.57	2.21	1.14	9.47	0.62
14		MC	4.93	4.13	3.43	3.13	2.93	2.53	2.23	10.39	1.04
15		RH	3.84	3.53	3.33	3.21	2.69	2.24	1.12	8.51	0.63
16		Mean	3.980	3.448	3.134	3.016	2.705	2.316	1.380	9.235	0.727
17		SE	0.250	0.186	0.104	0.065	0.059	0.058	0.214	0.326	0.078

Joining data sets

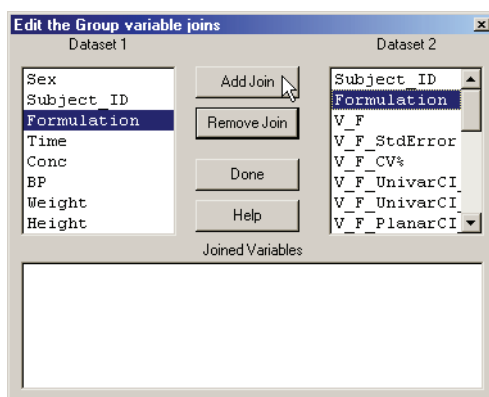
Template 9 merges two workbooks into one table. The Table Wizard joins the datasets by matching values for group and ID variables. The group and ID variables are selected in matched pairs, one from each workbook.

To define a join:

1. In the Table Wizard, select template 9 and select the two data sets that will provide table data. (See “[Template 9](#)” on page 142.)
2. *Group variables*: Click **Define Join** for group variables at the lower right, to match group variables from the two data sets. One set of statistics will be computed for each unique combination of group variable values.

The Select Join Variables dialog appears.

3. Select one variable from each data set to make a matched pair. They need not have identical names, but must represent the same entity with an overlapping set of values. The values of these variables will be used to merge records from the two data sets.



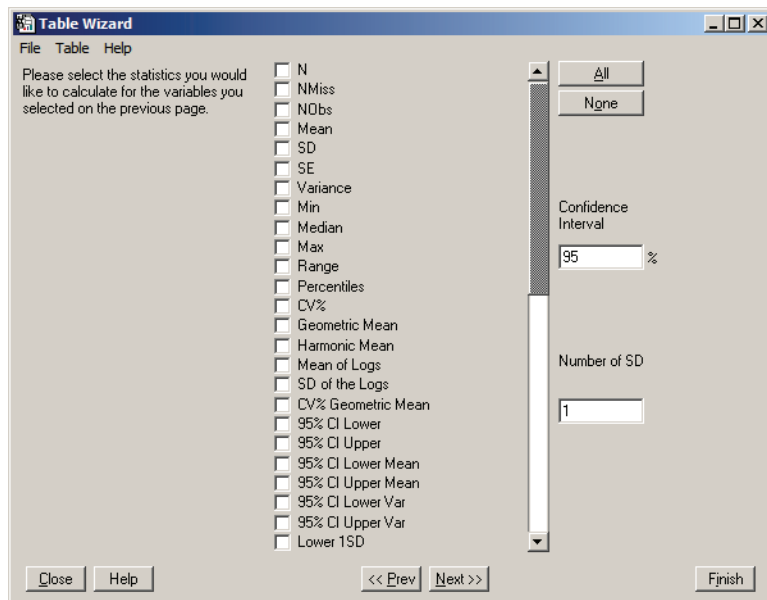
4. Click **Add Join**.
5. Repeat to set all needed pairs, then click **Done** to return to the [Table Wizard](#).
6. *ID variables*: Click **Define Join** for ID variables at the lower right and repeat the steps above to match variables that uniquely identify individual data profiles.

Summary statistics

The [Table Wizard](#) can compute summary statistics, which are a part of most table templates. (See “[WinNonlin table templates](#)” on page 134.)

To select the summary statistics to be included in a table:

1. Check the check box preceding each summary statistic to be included in the table. To select all of the available summary statistics, click **All**. See “[Output and computational formulae](#)” on page 153 for details on each statistic.
2. To include a confidence interval at a confidence level other than 95%, enter the desired percentage in the Confidence Interval box.
3. In the **Number of SD** field, enter the desired number of standard deviations for the following statistics: Lower_xSD, Upper_xSD, Geo_Lower_xSD, and Geo_Upper_xSD, where x = the number of specified standard deviations.



Note: The statistics in the table will appear in the order listed here.

4. When finished, click **Next** to set the [Significant digits and column alignments](#).

Significant digits and column alignments

The [Table Wizard](#) allows the user to select the number of decimal places or the number of significant digits to be used. This selection is made separately for each column and summary statistic in the table.

Variable name	Type	Decimal Places / Significant Digits	Alignment
Time	Numeric	Dec. Places = 2	Center
Formulation	Alpha	N/A	Center
Subject_ID	Alpha	N/A	Center
V_F	Numeric	Dec. Places = 2	Right
Conc	Numeric	Sig. Digits = 2	Right
V_F (Mean)	Numeric	Dec. Places = 3	Right
V_F (SE)	Numeric	Dec. Places = 3	Right
Conc (Mean)	Numeric	Sig. Digits = 3	Right
Conc (SE)	Numeric	Sig. Digits = 3	Right

Significant digits: For each table column and summary statistic listed, click the button under Significant digits/Decimal places to select which you will enter, then enter the number of significant digits or decimal places to the right.

Note: For variables with integer values, e.g. Subject or Period, choose Decimal places and enter 0 as the number of places.

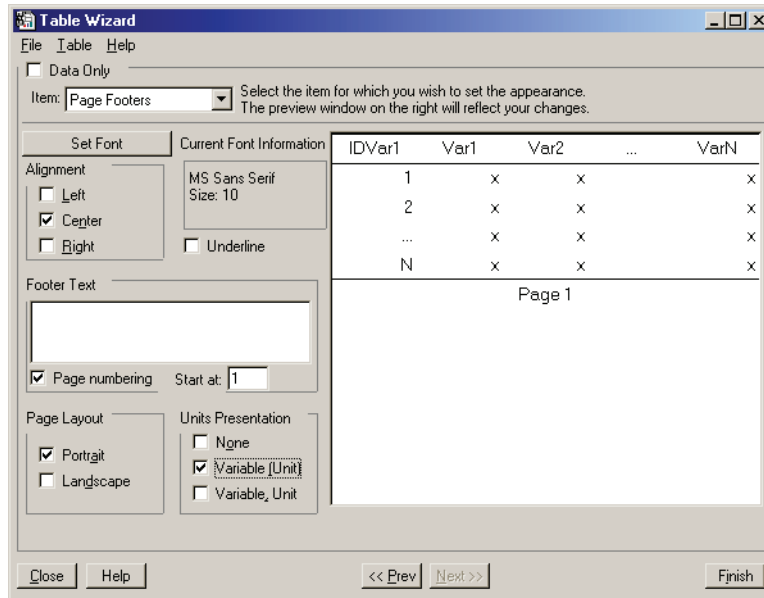
Column alignments: For each column and summary statistic, click the button under Alignment to select the alignment within table cells.

After the settings are complete, click **Next** to proceed with [Table formatting](#).

Table formatting

The final dialog of the [Table Wizard](#) provides the means to format table cells, headings and page layout, including page numbers and footnotes.

Check **Data Only** at the top left corner of this dialog to create a table (work-sheet) without page breaks, column headers, borders or formatting.



1. Select an item to format from the pull-down list called Item.
2. To include text at the bottom of each page, select **Page Footers** under Item and enter the text under Footer Text.
3. To include a table title on each page, select **Page Headers** under Item and type a title under Header Text. Insert line breaks by pressing the **ENTER** key.
4. Click the **Set Font** button to change the font, font size, or font style for that item. The Font dialog appears. Make adjustments and click **OK**.
5. To underline text for an item (e.g., Heading), check the **Underline** box.
6. Check the appropriate box under Alignment to align the table title, header or footer relative to the page. (Click the **Prev** button to set table cell alignments.)
7. To include page numbers at the bottom of each page, select **Page Footers** under Item and check **Page numbering** below the footer text and enter the first page number under **Start at**.
8. Select the appropriate radio button in the Page Layout group box: **Portrait** to orient the long side of the page vertically; **Landscape** to orient it horizontally.
9. If the input workbook included units, options for units display appear at the bottom right. Check **None** to exclude units from the table; **Variable (Unit)** to present units in parentheses after the variable name or **Variable, Unit** to present units after a comma, following the variable name.

Table definition files

A *table definition file* (*.TDF) stores a user-defined table format. The table definition file can then be reused to quickly create tables with that format. Table definition files are also useful for generating formatted tables via scripting. To use a table definition file, a data set must include the exact column names used in creating the definition file.

The table definition file includes:

- Table template number
- Names and mappings of table variables (data columns)
- Summary statistics to be included (if applicable)
- Titles (up to five lines)
- Footers (up to five lines)
- Formatting:
 - Alignment and decimal places or significant figures for the data
 - Table and page formats from the final dialog of the Table Wizard

Creating a table definition file

Table definition files must be created in WinNonlin, by creating a table with the desired formatting, then saving the settings from within the Table Wizard.

To create and save a table definition file (.tdf):*

1. Follow the instructions under “[Table Wizard](#)” on [page 131](#) up to the table preview (just before clicking the **Create** button), to set up a table with the desired template, variables, statistics (optional) and formatting.
2. Choose **File>Save Definition As (Alt+F, S)** from the Table Wizard menus. The Save Current Settings dialog appears.
3. Select a drive and directory and enter a descriptive file name, using the file extension .TDF.
4. Click **OK**.

Loading a table definition file

To load a table definition file (.tdf) in the Table Wizard:*

1. Open or create the worksheet containing data for the table. Make sure it uses the exact column names used in the Table Definition File.
2. Choose **Tools>Table Wizard** from the menus or click the **Table Wizard** button on the tool bar. The Table Wizard appears.
3. Select the appropriate workbook.
4. Choose **File>Open Definition** from the Table Wizard menus.
5. Select the appropriate table definition file and click **Open**.

Specifications made via the table definition file will be imported into the Table Wizard. The table can be edited if needed, as detailed under “[Table Wizard](#)” on page 131.

Descriptive Statistics

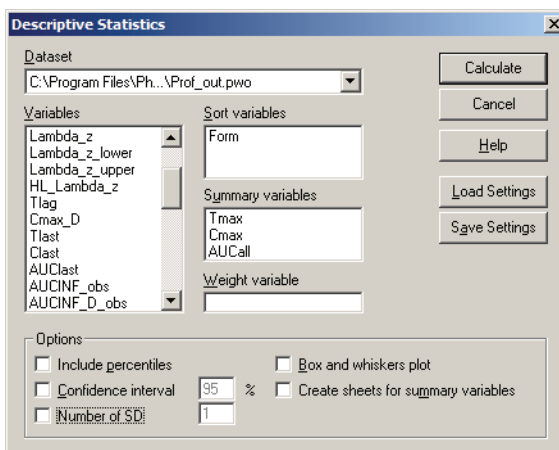
Summary statistics and weighted summary statistics

WinNonlin can compute summary statistics for variables in any workbook. This feature is frequently used to create data to plot means and standard errors, for preclinical summaries, to summarize modeling results, or to test for normal distribution of data. Separate statistics for subgroups are obtained through the use of sort variable(s).

Computing summary statistics

To set up descriptive statistics:

1. Open the input data worksheet. The data set must be open and not hidden.
2. Choose **Tools>Descriptive Statistics** from the WinNonlin menus, or click the **Descriptive Statistics** button on the tool bar.



The Descriptive Statistics dialog allows settings to be saved to an ASCII file for later re-use, through the Save Setup and Load Setup buttons.

3. Drag the variable(s) to be summarized to the **Summary Variables** box.

Note: Select variables either by dragging the variable name to the appropriate box, or by highlighting the variable name then, while pressing the **Shift** key, typing the underlined letter in the name of the destination box.

4. Select sort variable(s) if desired. A separate analysis is done for each unique combination of sort variable values.

5. Select the options for the statistics:

- *Weight variable:* (Optional) See “[Weighted summary statistics](#)” on [page 157](#).
- *Include percentiles:* Selecting the check box means that the first, fifth, tenth, twenty-fifth, fiftieth, seventy-fifth, ninetieth, ninety-fifth, and the ninety-ninth percentiles will be included in the output.
- *Confidence interval:* Enter the desired confidence interval in the % field.
- *Number of SD:* select the **Number of SD** check box to include four extra summary statistics in the output: Lower_xSD, Upper_xSD, Geo_Lower_xSD, and Geo_Upper_xSD, where x = the number of specified standard deviations. Enter the desired number of standard deviations in the **Number of SD** field.
- *Box and whiskers plot:* The Box and Whiskers plot appears in a separate text window. The box hinges are the 25th and 75th percentiles. The median is plotted as an asterisk (*). The whiskers end at the data points farthest from the median that are still within 1.5 times the H-spread (the distance between the hinges) of the hinges. If there are no outliers beyond this range, the whiskers end at the maximum and minimum points. Data points between 1.5 and 3 H-spreads from the hinges are marked by an \circ , and the those outside 3 H-spreads are marked by \times . If there are fewer than five data points, the data are graphed individually rather than as a box.
- *Create Sheets for Summary Variables:* When there are multiple summary variables the output can be sent to one or multiple worksheets in the output workbook. If output is combined onto one worksheet, no units will appear with the output. See “[Units](#)” on [page 153](#).

6. Click **Calculate**.**Units**

When summary statistics are calculated on a variable with units, certain of the calculations will have units. Assuming that the variable summarized is x and has x -units specified, the units for the summary statistics are:

Statistic	Units
N, Nmiss, Nobs	No units
CV%, CV% Geometric Mean	No units
Skewness	No units
Kurtosis	No units
Mean log, SD log	No units
Variance	$x\text{-unit}^2$
CI Lower Var, CI Upper Var	$x\text{-unit}^2$
Everything else	$x\text{-unit}$

Output and computational formulae

Summary statistics include the following computations.

Table 7-1. Descriptive statistics output

Statistic	Description
N	Number of observations with non-missing data
Nmiss	Number of observations with missing data
Nobs	Number of observations
Mean	Arithmetic Average
SD	Standard Deviation: $\sqrt{\text{Variance}}$
SE	Standard Error: $\frac{SD}{\sqrt{N}}$

Table 7-1. Descriptive statistics output (continued)

Statistic	Description
Variance	Unbiased sample variance: $\frac{\sum_{i=1}^N (X_i - \bar{X})^2}{N - 1}$
Min	Minimum value
Median	Median value
Max	Maximum value
Range	Range of values (maximum value minus minimum value)
CV%	Coefficient of variation: (SD/Mean)*100
Geometric Mean	Nth root of the product of the N observations
Harmonic Mean	Reciprocal of the arithmetic mean of the reciprocals of the observations

Table 7-1. Descriptive statistics output (continued)

Statistic	Description
Pseudo SD	<p>Jackknife estimate of the standard deviation of the harmonic mean. For n points, x_1, \dots, x_n the pseudo standard deviation is:</p> $\text{pseudo SD} = \sqrt{(n-1) \sum_{i=1}^n (\bar{H}_i - \bar{H})^2}$ <p>where</p> $\bar{H} = \frac{1}{n} \sum_{i=1}^n \bar{H}_i$ <p>and</p> $\bar{H}_i = \frac{(n-1)}{\left(\frac{1}{x_1} + \frac{1}{x_2} + \dots + \frac{1}{x_{i-1}} + \frac{1}{x_{i+1}} + \dots + \frac{1}{x_n}\right)}$ $= \frac{(n-1)}{\left(\sum_{j=1}^n \frac{1}{x_j}\right) - \frac{1}{x_i}}$
Mean log	Arithmetic average of the natural logs of the observations
SD Log	Standard deviation of the natural logs of the observations
Geometric Mean	Nth root of the product of the N observations
CV% Geometric Mean	<p>Coefficient of variation of the geometric mean:</p> $\sqrt{\exp((\text{SD_log})^2) - 1} \times 100$
CI X% Lower	Lower limit based on the chosen confidence interval.
CI X% Upper	Upper limit based on the chosen confidence interval.
CI X% Lower Mean	<p>Lower limit of an X% confidence interval for the mean:</p> $\text{Mean} - t_{\alpha/2} \frac{SD}{\sqrt{N}}$

Table 7-1. Descriptive statistics output (continued)

Statistic	Description
CI X% Upper Mean	Upper limit of an X% confidence interval for the mean: $Mean + t_{\alpha/2} \frac{SD}{\sqrt{N}}$ <p>where $(1 - \alpha) \times 100$ is the percentage given for the confidence interval, and $t_{\alpha/2}$ is from the t-distribution with N-1 degrees of freedom. Thus, a 95% confidence level indicates that alpha = 0.05. Note that for N>30, the t-distribution is close to the normal distribution.</p>
CI X% Lower Var	Lower limit of an X% confidence interval for the variance: $\frac{(N - 1) \times Variance}{\chi_U^2}$
CI X% Upper Var	Upper limit of an X% confidence interval for the variance: $\frac{(N - 1) \times Variance}{\chi_L^2}$ <p>where χ_L^2 and χ_U^2 are from the χ^2-distribution with N-1 degrees of freedom. χ_L^2 cuts off a lower tail, and χ_U^2 cuts off an upper tail, of area $\alpha/2$ where $(1 - \alpha) \times 100$ is the percentage for the confidence interval.</p>
Lower XSD	X = the number of SD chosen.
Upper XSD	X = the number of SD chosen.
CI Geo X% Lower	Lower limit for the geometric mean based on the chosen confidence interval.
CI Geo X% Upper	Upper limit for the geometric mean based on the chosen confidence interval.
CI X% Lower Geo Mean	Lower limit of an X% confidence interval for the geometric mean.
CI X% Upper Geo Mean	Upper limit of an X% confidence interval for the geometric mean.
Geo Lower XSD	X = the number of SD chosen.
Geo Upper XSD	X = the number of SD chosen.

Table 7-1. Descriptive statistics output (continued)

Statistic	Description
P(ercentiles)	<p>The Pth percentile divides the distribution at a point such that P percent of the distribution are below this point. For a sample size of n, the quantile corresponding to the proportion p (0<p<1) is defined as:</p> $Q(p) = (1 - f) * x(j) + f * x(j+1)$ <p>where</p> $j = \text{int}(p*(n+1)), \text{ (integer part)}$ $f = p*(n+1) - j, \text{ (fractional part)}$ <p>x(j) is the j-th order statistic.</p> <p>The above is used if 1 <= j < n. Otherwise, the empirical quantile is the smallest order statistic for j=0 or the largest order statistic for j=n.</p>
Skewness	<p>Coefficient of skewness:</p> $\frac{\sum w_i (x_i - \bar{x}_w)^3 / N}{[\sum w_i (x_i - \bar{x}_w)^2 / N]^{3/2}}$
Kurtosis	<p>Coefficient of excess (kurtosis):</p> $\frac{\sum w_i (x_i - \bar{x}_w)^4 / N}{[\sum w_i (x_i - \bar{x}_w)^2 / N]^2} - 3$
KS_pval	Kolmogorov-Smirnov normality test p value.

Weighted summary statistics

Summary statistics can be weighted, using a column in the data set to provide weights. If a weight is non-numeric, missing, or excluded, it is set to zero.

To create weighted summary statistics:

1. Make all specifications as for un-weighted summary statistics. See “Computing summary statistics” on page 151.
2. Drag the variable containing weights to the **Weight Variable** box (or highlight the variable to be used as the weight and press **Shift+W**).

Output and computational formulae for weighted calculations

The output for weighted summary statistics contains a column indicating the summary variable(s), one for each sort variable, and the statistics below.

Table 7-2. Weighted summary statistics output

Statistic	Description
N	Number of non-missing observations (including those with weights = 0)
Nmiss	Number of observations with missing data
Nobs	Number of observations (including observations with weights of 0)
Mean	Weighted Arithmetic average: $\frac{\sum w_i \times x_i}{\sum w_i}$
SD	Weighted Standard Deviation: $\sqrt{\text{Weighted Variance}}$
SE	Weighted Standard Error: $\frac{\text{Weighted SD}}{\sqrt{N}}$
Variance	Weighted Variance: $\frac{\sum w_i (x_i - \bar{x}_w)^2}{N - 1}$ where \bar{x}_w is the weighted mean.
CV%	Weighted Coefficient of variation: (Weighted SD/Weighted Mean)*100
CI X% Lower	Lower limit of an X% confidence interval for the data: Mean - $t_{\alpha/2}$ * SD.
CI X% Upper	Upper limit of an X% confidence interval for the data: Mean + $t_{\alpha/2}$ * SD.
CI X% Lower Mean	Lower limit of an X% confidence interval, using weighted mean and SD
CI X% Upper Mean	Upper limit of an X% confidence interval, using weighted mean and SD
CI X% Lower Var	Lower limit of an X% confidence interval, using weighted variance
CI X% Upper Var	Upper limit of an X% confidence interval, using weighted variance
Lower XSD	Range determined by adding or subtracting X standard deviations from the mean: Mean +/- X*SD.

Table 7-2. Weighted summary statistics output (continued)

Statistic	Description
Upper XSD	Range determined by adding or subtracting X standard deviations from the mean: Mean +/- X*SD.
CI GEO X% Lower	Lower limit of an X% confidence interval for the logs of the data, back-transformed to original scale: $\exp(\text{Mean_Log} - t_{\alpha/2} * \text{SD_Log})$.
CI GEO X% Upper	Upper limit of an X% confidence interval for the logs of the data, back-transformed to original scale: $\exp(\text{Mean_Log} + t_{\alpha/2} * \text{SD_Log})$.
CI 95% Lower GEO Mean	Lower limit of an X% confidence interval for the Geometric Mean: $\exp(\text{Mean_Log} - t_{\alpha/2} * \text{SD_Log} / \sqrt{n})$
CI 95% Upper GEO Mean	Upper limit of an X% confidence interval for the Geometric Mean: $\exp(\text{Mean_Log} + t_{\alpha/2} * \text{SD_Log} / \sqrt{n})$
GEO Lower 1SD and GEO Upper 1SD	<p>Range determined by adding or subtracting "X" log standard deviations from the log-mean, back transformed to original scale: $\exp(\text{Mean_Log} \pm X * \text{SD_Log})$</p> <p>When X = 1, this range is equivalent to: $(\text{Geometric_Mean} / \exp(\text{SD_Log}))$ and $\text{Geometric_Mean} * \exp(\text{SD_Log})$</p>
Skewness	<p>Weighted coefficient of skewness:</p> $\frac{\sum w_i (x_i - \bar{x}_w)^3 / n}{[\sum w_i (x_i - \bar{x}_w)^2 / n]^{3/2}}$
Kurtosis	<p>Weighted coefficient of excess (kurtosis):</p> $\frac{\sum w_i (x_i - \bar{x}_w)^4 / n}{[\sum w_i (x_i - \bar{x}_w)^2 / n]^2} - 3$

Noncompartmental Analysis

Areas and slopes for plasma, urine or PD data

WinNonlin's noncompartmental analysis (NCA) engine computes derived measures from raw data, using methods appropriate for either richly- or sparsely-sampled data. WinNonlin provides the following noncompartmental analysis “models” for different types of input data.

- Six PK models: extravascular, intravascular (IV) infusion or IV bolus dosing, for concentration data from blood/plasma (single dose and steady state) or urine (single dose only). These models support rich data sets as well as sparsely-sampled studies such as toxicokinetic studies.
- One PD model for analysis of slope, height, areas, and moments in richly-sampled time-effect data.

The following sections detail the steps to set up a WinNonlin NCA.

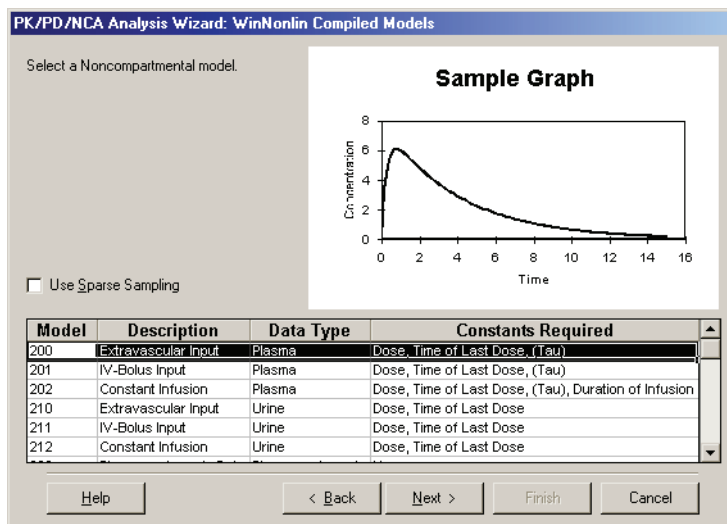
1. [“NCA model selection” on page 162](#)
2. [“Model variables” on page 163](#)
3. [“Lambda Z or slope estimation settings” on page 164](#)
4. [“Partial areas” on page 169](#)
5. [“Therapeutic response” on page 170](#)
6. [“Dosing regimen” on page 173](#)
7. [“Model options” on page 175](#)

NCA computations and output parameters are covered under [“Computational rules for WinNonlin’s noncompartmental analysis engine” on page 183](#)

NCA model selection

To load a noncompartmental analysis model:

1. Open the worksheet containing data to be modeled, and select it as the active worksheet (click on that worksheet to bring it to the front).
2. Start the **PK/PD/NCA Analysis Wizard** by clicking on that tool bar button or selecting that command from the **Tools** menu.
3. Select the **Noncompartmental** radio button. Note that the Model 200 is highlighted and selected by default.
4. Click **Next**.



5. Select the appropriate NCA model. See “Noncompartmental analysis” on page 509 for details on the models and output parameters.
6. To analyze data with few observations per subject, check **Use Sparse Sampling**. See “Sparse sampling computations” on page 190 for discussion, and “Sparse sampling (pre-clinical) data” on page 518 for additional parameters computed. Sparse data sets can include up to 250 subjects.
7. Click **Next**.
8. Confirm that the correct model is selected, and click **Finish** to load the model.

The model window will open with a sample graph. The next step in NCA is setting **Model variables**.

Model variables

To specify data columns for each variable in a noncompartmental analysis:

1. After [NCA model selection](#), choose **Model>Data Variables** from the Win-Nonlin menus to open the Data Variables dialog.
2. For plasma models, drag and drop columns to assign variables to be modeled (see [“Urine models” on page 163](#) for urine models):
 - X: the independent variable, generally time.
 - Y: the dependent variable, i.e., observed values of interest.
3. For sparse data sets, drag and drop to assign the variable containing subject identifiers into the Subj_ID field.
4. Drag and drop to specify the following (optional):
 - Sort Variable(s): categorical variable(s) identifying individual data profiles, e.g., *subject ID* and *treatment* in a crossover study. A separate analysis is done for each unique combination of sort variable values.
 - Carry Along Variables: data to be copied into the output workbook.

Note: The output data set will automatically include the original values of the sort variable(s) and variables being modeled (X and Y for plasma models or mid-point and rate for urine models), in addition to the estimated parameters.

5. Click **OK**.

[Lambda Z or slope estimation settings](#) are the next step in NCA setup.

Urine models

For urine models, WinNonlin computes the midpoint and excretion rate (amount eliminated per unit of time) for each collection interval. Urine models include the following variables. Drag and drop columns to assign variables to be modeled as detailed under [“Model variables” on page 163](#).

- Concentration: the dependent variable, drug concentration in urine.
- Volume: the volume of urine collected per time interval.
- Lower times: the starting times for individual collection intervals.

- Upper times: the ending times for individual collection intervals.

Lambda Z or slope estimation settings

WinNonlin will attempt to estimate the rate constant, λ_z , associated with the terminal elimination phase for concentration data, or slopes of up to two selected data ranges for drug effect data. These measures will be estimated using linear or linear/log trapezoidal rule, as selected in the Model Options (see “[Model options](#)” on page 175). If λ_z can be estimated, parameters for concentration data will be extrapolated to infinity. NCA does not extrapolate beyond the observed data for drug effect.

Lambda Z or slope range selection

If the user does not specify the data points to include in the calculation of λ_z or slopes, yet does not choose disable curve stripping as described under “[Setting Lambda Z or PD slope ranges](#)” on page 166, then WinNonlin will determine the data points to include in λ_z or slope calculations as follows.

For concentration data

During the analysis, WinNonlin repeats regressions using the last three points with non-zero concentrations, then the last four points, last five, etc. Points prior to Cmax or prior to the end of infusion are not used unless the user specifically requests that time range. Points with a value of zero for the dependent variable are excluded. For each regression, an adjusted R^2 is computed:

$$\text{Adjusted } R^2 = 1 - \frac{(1 - R^2) \times (n - 1)}{(n - 2)}$$

where n is the number of data points in the regression and R^2 is the square of the correlation coefficient.

WinNonlin estimates λ_z using the regression with the largest adjusted R^2 and:

- If the adjusted R^2 does not improve, but is within 0.0001 of the largest adjusted R^2 value, the regression with the larger number of points is used.
- λ_z must be positive, and calculated from at least three data points.

For sparse sampling data

The Lambda Z ranges for sparse data show the mean concentration at each time (plasma or serum data) or mean rate for each interval (urine data).

For drug effect data

WinNonlin will compute the best-fitting slopes at the beginning of the data and at the end of the data using the same rules that are used for lambdaZ—roughly, best adjusted R-square with at least three points, with linear or log regression, according to the user's choice. If the user specifies the range only for Slope1, then in addition to computing Slope1, WinNonlin will compute the best-fitting slope at the end of the data for Slope2. If the user specifies the range only for Slope2, then WinNonlin will also compute the best-fitting slope at the beginning of the data for Slope1.

The data points included in each slope are indicated on the Summary table of the output workbook and text for model 220. Data points for Slope1 are marked with “1” in workbook output and footnoted using “#” in text output; data points for Slope2 are labeled “2” and footnoted using an asterisk, “*”.

Calculation of Lambda Z

Once the time points being used in the regression have been determined, WinNonlin can estimate λ_z by performing a regression of the natural logarithm of the concentration values in this range on sampling time. λ_z is defined as:

$$\lambda_z = -1 \text{ (estimated slope)}$$

Note: Using this methodology, WinNonlin will almost always compute an estimate for λ_z . It is the user's responsibility to evaluate the appropriateness of the estimated value.

Calculation of slopes for effect data

WinNonlin estimates slopes by performing a regression of the response values or their natural logarithm. The actual slopes are reported; they are not negated as for λ_z .

Limitations of Lambda Z and slope estimation

It is not possible for WinNonlin to estimate λ_z or slope in the following cases.

- There are only two non-missing observations and the user requested automatic range selection.
- The user-specified range contains fewer than two non-missing observations.
- Automatic range selection is activated, but there are fewer than 3 positive concentration values at or after the Cmax of the profile.
- For infusion data, automatic range selection is activated, and there are fewer than 3 points at or after the infusion stop time.
- The time difference between the first and last data points in the estimation range used is approximately $< 1e-10$.

In these instances, parameters that do not depend on λ_z (i.e. Cmax, Tmax, AUClast, etc.) will still be reported and the software will issue a warning in the text output, indicating that λ_z was not estimable.

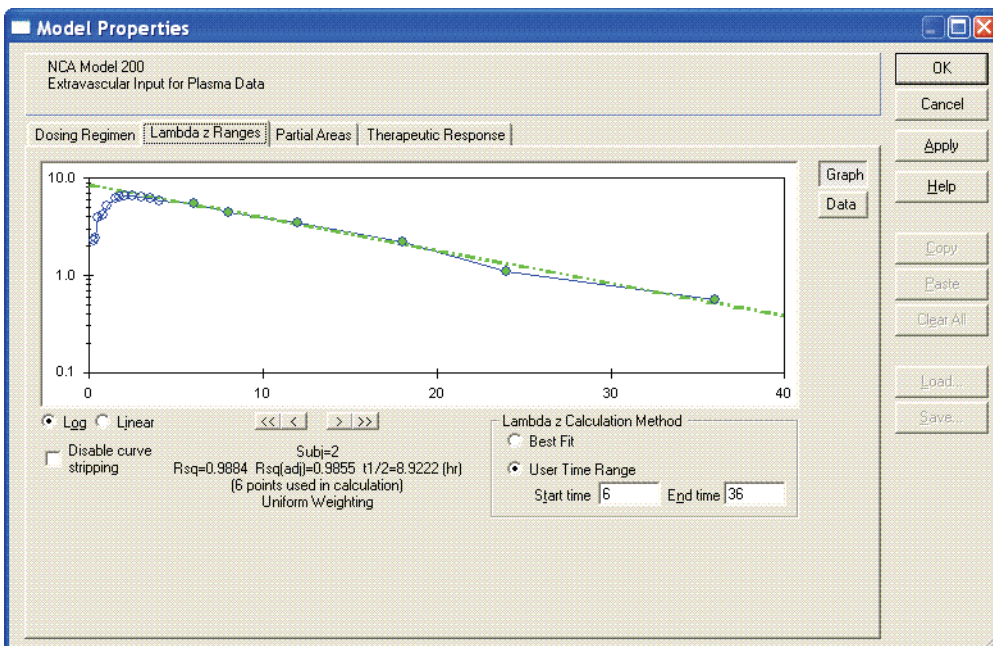
Setting Lambda Z or PD slope ranges

To set the λ_z range for each PK profile, or slopes to compute for each PD profile:



1. Open the Lambda Z Ranges tab or, for model 220, the Slopes tab of the Model Properties dialog (**Model>Lambda Z Ranges** or **Model>Slopes**).

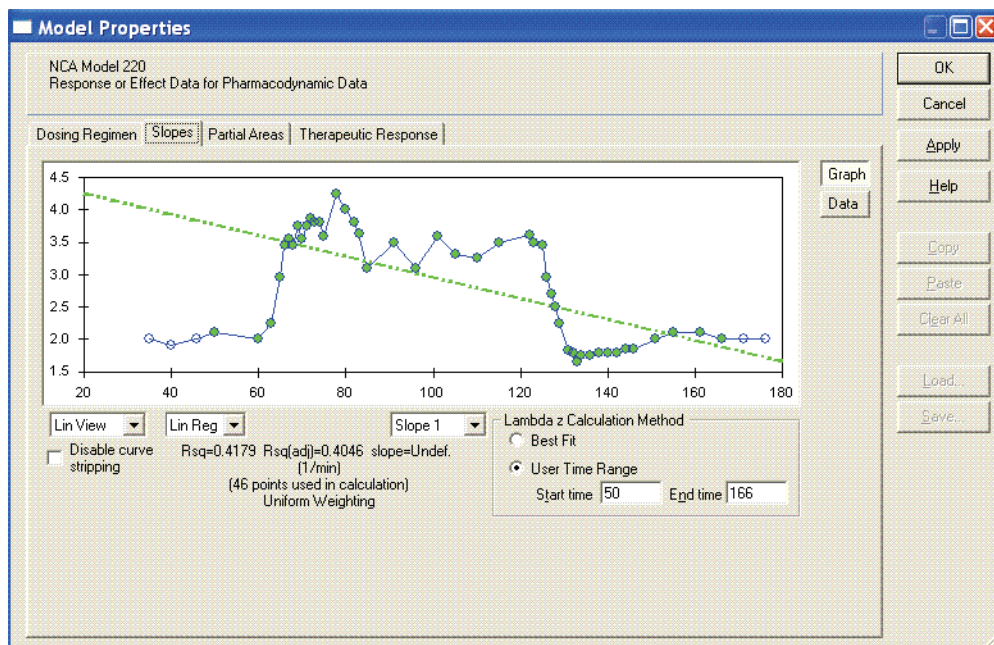
The Graph and Data buttons at the top right corner of the tab switch between a graphical or tabular interface to set time ranges for each data profile. The instructions below are for the graph view. (Individual profiles are defined by the sort variable(s) in “[Model variables](#)” on page 163).



- To turn off λ_z or slope estimation for all profiles, check **Disable Curve Stripping** at the bottom left and skip to step 8. With this option checked:
 - λ_z or slopes will not be estimated.
 - Parameters that are extrapolated to infinity will not be calculated.

Note: To disable curve stripping for one or more individual profiles, enter a start time that is greater than the last time point in a given profile, and an end time greater than the start time.

- Select the axis scale for the graphical representation below the lower left corner of the plots: **Linear** or **Log Linear**.
- For slopes, select whether each slope should be computed using linear regression (**Lin Reg**) or regression on the log data (**Log Reg**).



5. Data ranges (optional): For each profile, define the terminal elimination phase or slopes regions as follows.
 - a. *Best Fit*: Select this option to have WinNonlin determine the Lambda Z or Slope ranges as described under “[Lambda Z or slope range selection](#)” on page 164.
 - b. *User Time Range*: Select the first data point to be included: left-click on a data point or enter a time value under Start time. Select the last data point to be included: hold down the **Shift** key while left-clicking on a data point or enter a time value under End time. For slopes, use these steps to choose data ranges for up to two slopes, Slope1 and Slope2.
 - c. Use the right/left arrows just below the plot view to move between profiles.

Note: The units for Start and End time are those of observation times in the data set.

6. Exclusions (optional): to exclude a data point from the calculations, hold down the **Ctrl** key and left-click on that point. Repeat to reverse an exclusion. These exclusions apply only to Lambda Z or slope calculations; the excluded data points will still be included in computation of AUC's, moments, etc.

7. Use the arrow buttons directly below the plot to move between profiles, and set values for each profile.
8. When done, click **Apply** to save the settings and move to another tab of the Model Properties dialog. To save the settings and close the dialog, click **OK**.

Loading Lambda Z or slope ranges from an ASCII file

The **Load** and **Save** buttons make it possible to store these time ranges to an external ASCII data file, and reload them for re-use.

Partial areas

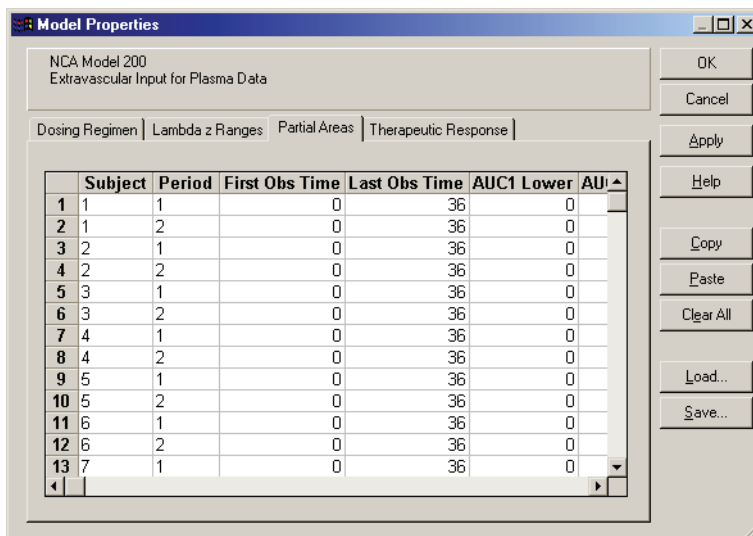
The Partial Areas tab of the Model Properties dialog includes settings for computation of partial areas under the curve. Partial area computations are optional; to exclude them, simply leave the start and end times on this tab empty.

Up to 126 partial areas can be computed per profile, by setting start and end times for each. The start and end times need not appear in the data set, and can be after the last observed time, if λ_z is estimable. See “Partial areas” on page 186 for additional information on partial area computations.

To specify the partial areas under the curve to be calculated:



1. Open the Partial Areas tab of the Model Properties dialog (**Model>Partial Areas** or click the **Partial Areas** tool bar button).
2. Enter the start time in the cell labeled AUC1 Lower, and the end time in the cell labeled AUC1 Upper. The time units are those of the input data set.
3. For another partial area, repeat this step for the next area (AUC2, etc.).



4. Repeat steps 2 and 3 for other profiles or use the fill-down method (drag bottom right corner of selected cells) to copy from one profile to those below.

Use the Save and Load buttons to save all settings in the Partial Areas tab to an ASCII data file, and re-load it for later use.

5. Click **OK** when all settings are complete.

Therapeutic response

The Therapeutic Response tab of the Model Options dialog provides settings for computing times and areas during which observations fall above or below target values.

For non-compartmental analysis of concentration data, WinNonlin will compute additional parameters noted in [Table 8-1](#) below. Parameters included depend on whether upper, lower, or both limits are supplied. See “[Therapeutic response windows](#)” on [page 188](#) for additional computation details.

Table 8-1. Therapeutic window output for PK data

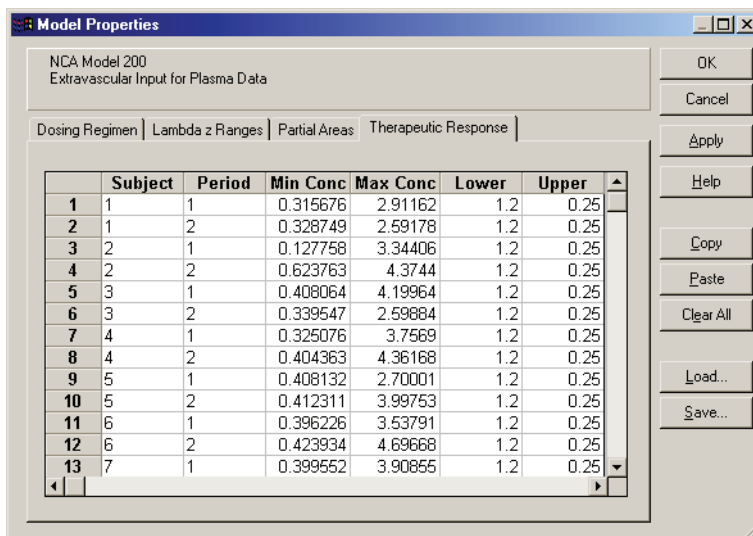
Input values	Parameters computed	Time range for computation	Description
Lower and upper	TimeLow TimeDuring TimeHigh AUCLow AUCDuring AUCHigh	From dose time to last observation	Total time below lower limit Total time between lower limit and upper limit Total time above upper limit AUC for time below lower limit AUC for time between lower limit and upper limit AUC for time above upper limit
Lower and upper	TimeInfDuring TimeInfHigh AUCInfLow AUCInfDuring AUCInfHigh	From dose time to infinity using λ_z (if λ_z exists)	Total time between lower limit and upper limit Total time above upper limit AUC for time below lower limit AUC for time between lower limit and upper limit AUC for time above upper limit
Lower only	TimeLow TimeHigh AUCLow AUCHigh	From dose time to last observation	Total time below lower limit Total time above lower limit AUC for time below lower limit AUC for time above lower limit
Lower only	TimeInfHigh AUCInfLow AUCInfHigh	From dose time to infinity using λ_z (if λ_z exists)	Total time (extrapolated to infinity) above lower limit AUC for time (extrapolated to infinity) below lower limit AUC for time (extrapolated to infinity) above lower limit

For drug effect data (model 220), WinNonlin will make the computations detailed under “[Drug effect data \(model 220\)](#)” on page 519.

To compute time and AUC spent within a therapeutic concentration range:



1. Open the Therapeutic Response tab of the Model Properties dialog (**Model>Therapeutic Response** or click the **Therapeutic Response** tool bar button).



NCA Model 200
Extravascular Input for Plasma Data

Dosing Regimen | Lambda z Ranges | Partial Areas | **Therapeutic Response**

	Subject	Period	Min Conc	Max Conc	Lower	Upper
1	1	1	0.315676	2.91162	1.2	0.25
2	1	2	0.328749	2.59178	1.2	0.25
3	2	1	0.127758	3.34406	1.2	0.25
4	2	2	0.623763	4.3744	1.2	0.25
5	3	1	0.408064	4.19964	1.2	0.25
6	3	2	0.339547	2.59884	1.2	0.25
7	4	1	0.325076	3.7569	1.2	0.25
8	4	2	0.404363	4.36168	1.2	0.25
9	5	1	0.408132	2.70001	1.2	0.25
10	5	2	0.412311	3.99753	1.2	0.25
11	6	1	0.396226	3.53791	1.2	0.25
12	6	2	0.423934	4.69668	1.2	0.25
13	7	1	0.399552	3.90855	1.2	0.25

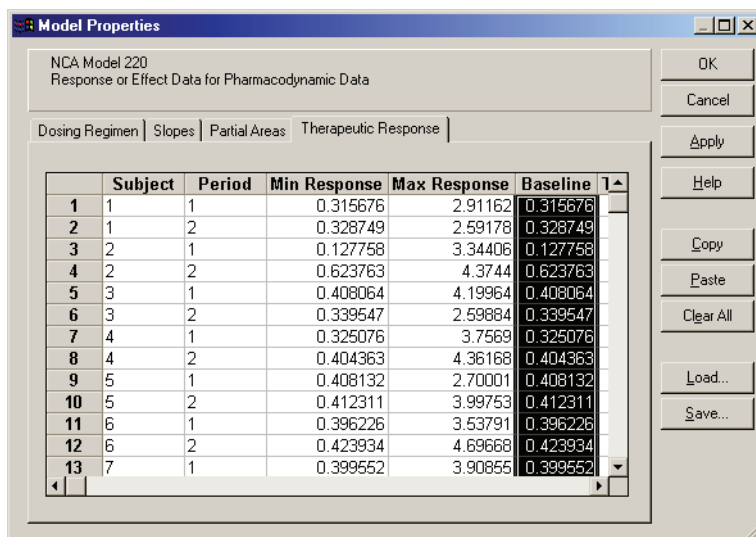
Buttons: OK, Cancel, Apply, Help, Copy, Paste, Clear All, Load..., Save...

2. Enter the lower and upper (optional) concentrations delimiting the target range and click **OK**.

To set the baseline and (optional) threshold values for effect data (model 220):



1. Open the Therapeutic Response tab of the Model Properties dialog (**Model>Therapeutic Response** or click the **Therapeutic Response** tool bar button).
2. For each profile, enter the following:
 - a. *Baseline*: baseline effect value for each profile. Required unless dosing data are provided in a Dosing worksheet (as when data are loaded from PKS). In that case, if no baseline value is entered, WinNonlin will use the effect value at dose time, if any, as baseline. If there is no data point at dose time, a value must be entered here.
 - b. *Threshold* (optional): effect value for use in calculating additional times and areas. (See “Additional parameters for model 220 with threshold” on page 521.)



	Subject	Period	Min Response	Max Response	Baseline
1	1	1	0.315676	2.91162	0.315676
2	1	2	0.328749	2.59178	0.328749
3	2	1	0.127758	3.34406	0.127758
4	2	2	0.623763	4.3744	0.623763
5	3	1	0.408064	4.19964	0.408064
6	3	2	0.339547	2.59884	0.339547
7	4	1	0.325076	3.7569	0.325076
8	4	2	0.404363	4.36168	0.404363
9	5	1	0.408132	2.70001	0.408132
10	5	2	0.412311	3.99753	0.412311
11	6	1	0.396226	3.53791	0.396226
12	6	2	0.423934	4.69668	0.423934
13	7	1	0.399552	3.90855	0.399552

3. Click **OK** when finished.

Dosing regimen

Models 200 to 202 (Plasma) assume that data were taken after a single dose, or after a final dose at steady state. For steady state, WinNonlin also assumes equal dosing intervals. Models 210 to 212 assume single-dose urine data. If dose time is not entered, WinNonlin will assume a time of zero.

Enter dose information in the Dosing Regimen dialog, as described below, or read it from a worksheet as detailed in “[Dosing data variables](#)” on page 204.

To enter dosing via the dosing regimen dialog:



1. Choose **Model>Dosing Regimen** from menus or click the **Dosing Regimen** tool bar button to open the Dosing Regimen tab of the Model Properties.
2. *Current Units*: Enter the units or select one item under Mass Prefix and Mass Unit then click the **Add** button.
3. *Dose Normalization*: If dose amount is normalized by subject bodyweight or body mass index, select the appropriate factor in this drop-list. For example, if doses are in milligrams per kilogram of bodyweight, enter Current Units of mg and Dose Normalization of kg. This will affect output parameter units. For example, if Volume units were L, dose normalization of kg would change those units to L/kg. Dose normalization affects units for all volume and

clearance parameters in NCA and PK models, as well as AUC/D in NCA plasma models, and Percent_Recovered in NCA urine models.

4. For drug effect data (model 220): Using the same time scale and units as the input data set, enter the time of the most recent dose for each profile under Value for each profile.

Model Properties

NCA Model 200
Extravascular Input for Plasma Data

Dosing Regimen | Lambda z Ranges | Partial Areas

	Subject	Treatment	Constant	Value
1	1	Theo12 BID	Dose	200
2	1	Theo12 BID	Time of Last Dose	0
3	1	Theo12 BID	Tau	12
4	1	Theo24 QD	Dose	400
5	1	Theo24 QD	Time of Last Dose	0
6	1	Theo24 QD	Tau	24
7	2	Theo12 BID	Dose	200
8	2	Theo12 BID	Time of Last Dose	0

Dosing Units

Current Units: mg Restore Dose Normalization: none

Mass Prefix: Mass Unit: Clear Add

☒ Steady state

OK Cancel Apply Help Copy Paste Clear All Load... Save...

Excluded profiles appear in red (see “Data exclusion and inclusion” on page 60)

5. Use the **Save** or **Load** button to save or re-load dosing information using an external ASCII file. See “Special file formats” on page 434 for file format.
6. *Dose*: For concentration data (models 200-212), enter the dose amount under Value for each row in which Constant = Dose.
7. *Steady state*: For steady state dosing (models 200-202 only), check **Steady State** and enter the dose interval for each row in which Constant = Tau.
8. *Time of last dose*: If the dose was given at a time other than 0, enter the dose time under Value for each row in which Constant = Time of Last Dose.
9. Click **Apply** to commit changes or **OK** to commit and close the dialog.

Time deviations in steady-state data

When using steady-state data, WinNonlin computes AUC from dose time to dose time + tau, based on the tau value set above. However, in most studies,

there will be sampling time deviations. That is, if dose time = 0 and $\tau = 24$, the last sample might be at 23.975 or 24.083 hours. In this instance, the program will estimate the AUC based on the estimated concentration at 24 hours, and not the concentration at the actual observation time. For steady state data, C_{\max} , T_{\max} , C_{\min} and T_{\min} are found using observations taken at or after the dose time, but no later than dose time + τ .

Model options

Options for NCA output are divided into the following sections.

Table 8-2. NCA model options

Category	Options
Output options	<ul style="list-style-type: none">• Workbook, chart, and/or ASCII text formats for modeling results
Weight	<ul style="list-style-type: none">• Uniform or weighted least squares
Title	<ul style="list-style-type: none">• Text title for workbook, ASCII and chart output
NCA settings	<ul style="list-style-type: none">• Method for computing area under the curve, and format of final parameters worksheet
Units	<ul style="list-style-type: none">• Default units for output parameters
Parameter names	<ul style="list-style-type: none">• Names, inclusion and precision for output parameters

To set NCA model options:



1. Select **Model>Model Options** from the WinNonlin menus or click the **Model Options** tool bar button. The Model Options dialog appears.
2. Click on a tab to select and adjust a group of options from those in [Table 8-2](#).
3. Click **Apply** to set options or **OK** to set options and close the dialog.

Output options

After noncompartmental analysis is run, one or more new windows is generated, with content as selected in the following NCA [Model options](#).

Note: Set default output options via **Tools>Options** in the WinNonlin menus.

Exclude profiles with insufficient data: If this option is checked, *profiles* with all or many missing parameter estimates are excluded from the results. If it is not checked, profiles with insufficient data will output missing parameter values. See “[Data deficiencies that will result in missing values for PK parameters](#)” on page 184 for a listing of cases that produce missing estimates and are excluded using this option.

Workbook: Check this option to generate an output workbook. The NCA workbook contains the worksheets summarized in [Table 8-3](#). These worksheets present the same information as the text output, but in tabular form.

Table 8-3. Summary of NCA worksheets

Sheet name	Contents
Final Parameters	Estimates of the final parameters for each level of the sort variable
Dosing	The dosing regimen for the analysis
Summary Table	The sort variables, X, points included in the regression for λ_z or slopes, Y, predicted Y for the regression, residual for the regression, AUC, AUMC and weight used for the regression
Partial Areas	See “ Partial areas ” on page 169.
User Settings	User defined settings
History	History of actions done in and to the workbook. See “ History worksheet ” on page 37.

Chart: If workbook output is selected, check this box to generate an NCA Chart window with a plot of observed versus predicted data for each profile.

Text: This option provides an ASCII text version of the analysis results, containing the same information as the workbook output. Modeling errors are also reported in this file.

Page Breaks: Check this box to include page breaks in the ASCII text output

Output intermediate calculations: If this option is checked, the text output for the next analysis only will include values for iterations during estimation of λ_z or slopes, and for each of the sub-areas from partial area computations, as shown below.

NCA Text - [Untitled16] (Read-only) (Derived)

Intermediate Output: Trying different time intervals to get best fit for Lambda2

Lower_time	Upper_time	PtsUsed	Lambda2	Rsq	Adj_Rsq
120.00	240.00	3	0.0155	0.9815	0.9631
90.00	240.00	4	0.0160	0.9892	0.9838
60.00	240.00	5	0.0167	0.9913	0.9884
45.00	240.00	6	0.0170	0.9927	0.9908
30.00	240.00	7	0.0176	0.9897	0.9877
20.00	240.00	8	0.0188	0.9732	0.9687
15.00	240.00	9	0.0199	0.9596	0.9538

Best value for Lambda_z: 0.0170, and intercept: 6.8976

Read Only Line 1 / 1437

NCA Text - [Untitled16] (Read-only) (Derived)

Intermediate Output: Partial Areas

Computing partial area from 0.000000 to 120.000000:

piece from (0.000000, 0.000000) to (5.000000, 340.316700) = 850.791750

piece from (5.000000, 340.316700) to (10.000000, 1914.000000) = 5635.791750

piece from (10.000000, 1914.000000) to (15.000000, 2069.167000) = 9957.917500

piece from (15.000000, 2069.167000) to (20.000000, 1470.917000) = 8850.210000

piece from (20.000000, 1470.917000) to (30.000000, 788.750000) = 11298.335000

piece from (30.000000, 788.750000) to (45.000000, 496.416700) = 9638.750250

piece from (45.000000, 496.416700) to (60.000000, 372.833300) = 6519.375000

piece from (60.000000, 372.833300) to (90.000000, 204.250000) = 8656.249500

piece from (90.000000, 204.250000) to (120.000000, 124.050000) = 4924.500000

Partial area from 0.000000 to 120.000000 = 66331.920750

Read Only Line 1 / 1437

Weight

For NCA, selections on the Weight tab of the [Model options](#) dialog apply to the regression that estimates λ_z or slopes. Sparse sampling requires uniform weighting.

Model Options

NCA Model 200
Extravascular Input for Plasma Data

Output Options **Weight** Title NCA Settings Units Parameter Names

☐ Weights On File in Column ☐ Observed to Power n

☒ Pre-Selected Scheme ☐ Predicted to Power n

Uniform Weighting

☐ No Scaling of Weights

OK Cancel Apply Help

Note that it is not the weights themselves that are important; rather it is the relative proportions of the weights. See “[Weighting](#)” on page 243 for discussion of weighting schemes.

CAUTION: When a log-linear fit is done, it already uses weighting implicitly, approximately equal to $1/\hat{Y}^2$.

To use uniform weighting:

1. Select the **Pre-selected Scheme** radio button.
2. Select **Uniform Weighting** from the pull-down list.

To use weighted least squares:

1. Select **Observed to the Power n** button and enter the value of n , or use the **Pre-selected scheme** button to select the most common weighted least square options: $1/Y$ and $1/Y^2$.

Note: When weighting by $1/Y$ and using the Linear/Log trapezoidal rule, one could argue that the weights should be $1/\text{Log}Y$, rather than $1/Y$. However, if this were the case, concentrations between 0 and 1 would have negative weights, and therefore could not be included.

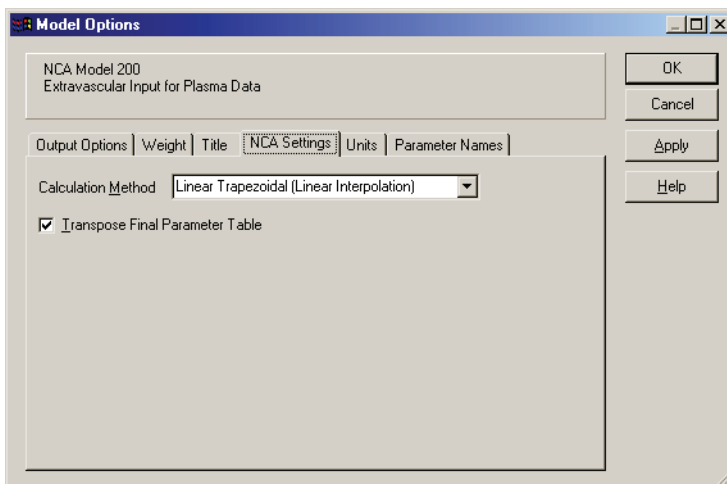
Title

If included, the title is displayed at the top of each printed page in ASCII text output, on the User Settings worksheet in the workbook output, and at the top of each chart in chart output. It may include up to 5 lines of text.

To include a title:

1. Make sure that **Title Text** is checked on the Title tab of the [Model options](#) dialog. If it is not, the title will not display even if text is entered below.
2. Enter the title text below. To include a line break use **Shift + Enter**. The title will be centered on the page.

NCA settings



Calculation methods for AUC

Four methods are available for the calculation of area under the curve, in the NCA Settings tab of the [Model options](#) dialog. The method chosen applies to all AUC, AUMC, and partial area computations. All methods reduce to the log trapezoidal rule and/or linear trapezoidal rule, but differ with regard to when these rules are applied. The default method is set on the Models tab of the WinNonlin options dialog (see “[WinNonlin options](#)” on page 28).

- Linear Trapezoidal rule (Linear Interpolation) [default and recommended for effect data (model 220)]. This method (method 2) is recommended for Model 220 (PD data). It uses the linear trapezoidal rule, given under “[AUC calculation and interpolation formulas](#)” on page 185, applied to each pair of consecutive points in the data set that have non-missing values, and sums up these areas. If a partial area is selected that has an endpoint that is not in the data set, then the linear interpolation rule, given below, is used to inject a concentration value for that endpoint.
- Linear Trapezoidal Method with Linear/Log Interpolation (method 4). As above except when a partial area is selected that has an endpoint that is not in the data set. In that case, WinNonlin inserts a final concentration value using: 1) the linear interpolation rule, or 2) logarithmic interpolation if the endpoint is after C_{max} , or after C_0 for bolus models (if $C_0 > C_{max}$). If C_{max} is not unique, then the first maximum is used.

- Linear/Log Trapezoidal Method (method 1). Uses the log trapezoidal rule (given under [“AUC calculation and interpolation formulas” on page 185](#)) after Cmax, or after C0 for bolus (if C0 > Cmax), and linear trapezoidal otherwise. Log interpolation is used to create points for partial areas after Cmax, or C0 for bolus models (if C0 > Cmax), and linear interpolation is used otherwise. If Cmax is not unique, then the first maximum is used.
- Linear Up/Log Down Method (method 3). The linear trapezoidal rule is used any time that the concentration data is increasing, and the logarithmic trapezoidal rule is used any time that the concentration data is decreasing. Points for partial areas are injected using the linear interpolation rule if the surrounding points show that concentration is increasing, and the logarithmic interpolation rule if the concentration is decreasing.

See [“AUC calculation and interpolation formulas” on page 185](#) for equations.

Note: The Linear/Log Trapezoidal, the Linear Up/Log Down, and the Linear Trapezoidal (with Linear/Log Interpolation) methods all apply the same exceptions in area calculation and interpolation: If a Y value (concentration, rate or effect) is less than or equal to zero, the program defaults to the linear trapezoidal or interpolation rule for that point. Similarly, if adjacent Y values are equal to each other, the program defaults to the linear trapezoidal or interpolation rule.

Transpose Final Parameter Table

This option, if checked, will create a Final Parameters worksheet with one column for each parameter. A second worksheet called Non-transposed Final Parameters will present one row for each parameter. If the Transpose Final Parameters option is not checked, the Final Parameters worksheet will present the untransposed format, and the second worksheet, entitled Transposed Final Parameters, will present the transposed version. See [“Transpose final parameter table” on page 208](#) for example output from compartmental modeling. This option is provided primarily for WinNonlin scripts that depend on the transposed format in the Final Parameters worksheet.

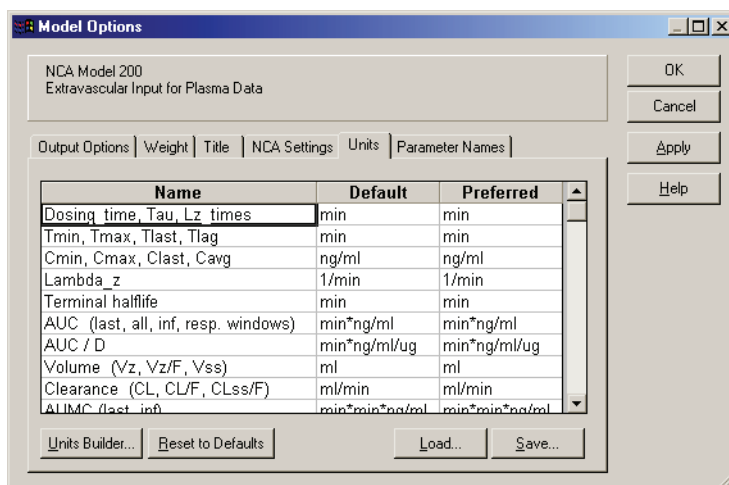
Units

The Units tab of the [Model options](#) dialog (below) lists default units for output parameters and supports entry of preferred units. WinNonlin converts values as necessary to display the output in the preferred units.

To set a preferred unit:

1. Enter the unit under **Preferred** for the appropriate row. The units cannot contain space characters. Case is preserved.
2. Click **Apply** or **OK**.

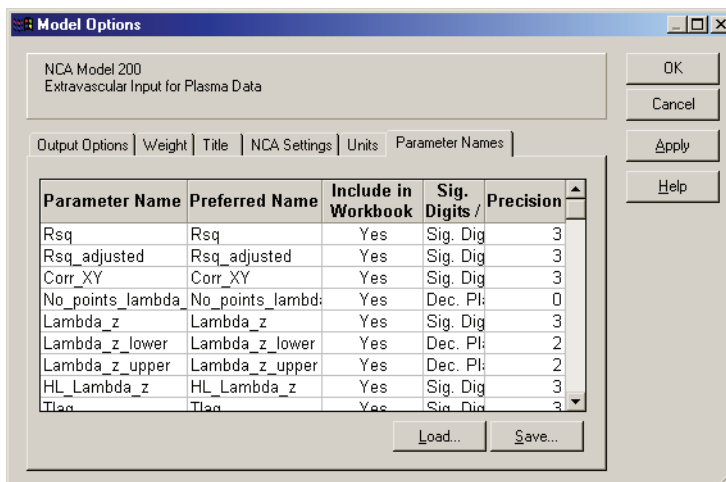
Note: Preferred units can be set only after units are set for the X-variable, Y-variable, and, for plasma models, dosing.



Use the Save and Load buttons to save preferred units as an ASCII text file, e.g., PLASMAUNITS.TXT or URINEUNITS.TXT, then load them back for later modeling.

Parameter names

For noncompartmental analyses (NCA), WinNonlin provides the option to specify the variable names and precision for the final model parameters.



Preferred name: Edit output parameter names as desired, unless parameter names are set by company defaults (see “[Default preferred parameter names](#)” below). The names cannot contain space characters. Case will be preserved.

Include in workbook: WinNonlin provides the option to include only a subset of final parameters in NCA workbook output. Click on a cell under Include in Workbook to toggle between Yes (include that parameter) and No (exclude).

Sig digits: Click on a cell in this row to select whether to enter significant digits, number of decimal places, or nothing as the precision for that parameter.

Precision: enter the number of significant digits or decimal places as (selected under Sig. Digits) for the parameter in that row.

All settings on the Parameter Names tab can be saved to a text file for later reuse by clicking the **Save** button (and loaded using the **Load** button). As the parameters vary by model and with steady state dosing, save parameter information for specific models to different files, for example: MODEL202.NAM, MODEL202SS.NAM and MODELURINE.NAM.

Default preferred parameter names

Preferred parameter names can be set by a map file containing individual preferences or company standards. When WinNonlin opens the Parameter Names tab for an NCA model, it will search its installation directory for an ASCII text file called DEFAULTNCAPARAMETERNAMES.MAP and load preferred parameter names from that file, if it exists. The file lists WinNonlin NCA parameter names paired with preferred names. Each line contains:

ParameterName=PreferredName, Include, Precision

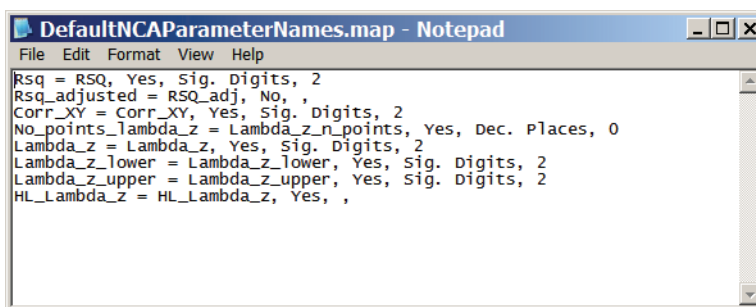
where:

ParameterName is WinNonlin's default name for an NCA parameter.

PreferredName is limited to alphanumeric characters and the underscore.

Include equals "Yes" or "No" and indicates whether to include that parameter wherever NCA parameters are output.

Precision equals "Sig. Digits" or "Dec. Places" followed by a comma, then an integer indicating the number of significant digits or decimal places to use.



```
File Edit Format View Help
Rsq = RSQ, Yes, Sig. Digits, 2
Rsq_adjusted = RSQ_adj, No, ,
Corr_XY = Corr_XY, Yes, Sig. Digits, 2
No_points_lambda_z = Lambda_z_n_points, Yes, Dec. Places, 0
Lambda_z = Lambda_z, Yes, Sig. Digits, 2
Lambda_z_lower = Lambda_z_lower, Yes, Sig. Digits, 2
Lambda_z_upper = Lambda_z_upper, Yes, Sig. Digits, 2
HL_Lambda_z = HL_Lambda_z, Yes, ,
```

Computational rules for WinNonlin's noncompartmental analysis engine

WinNonlin's NCA computations are covered under the following headings.

- "Data checking and pre-treatment" on page 183
- "AUC calculation and interpolation formulas" on page 185
- "Partial areas" on page 186 and "Therapeutic response windows" on page 188
- "Sparse sampling computations" on page 190
- "NCA output parameters and their computation formulas" on page 510

Data checking and pre-treatment

Prior to calculation of pharmacokinetic parameters, WinNonlin institutes a number of data-checking and pre-treatment procedures as follows.

Sorting

Prior to analysis, data within each profile is sorted in ascending time order. (A profile is identified by a unique combination of sort variable values.)

Insertion of initial time points

If a PK profile does not contain an observation at dose time (C0), WinNonlin inserts a value using the following rules:

- Extravascular and infusion data: For single dose data, a concentration of zero; for steady-state, the minimum observed during the dose interval.
- IV bolus data: The program performs a log-linear regression of first two data points to back-extrapolate C0. If the regression yields a slope ≥ 0 , or at least one of the first two y-values is 0, or if one or both of the points is viewed as an outlier and excluded from the lambda Z regression, then the first observed positive y-value will be used as an estimate for C0. If a weighting option was selected, it is used in the regression.
- Drug effect data: An effect value equal to the user-supplied baseline is inserted at dose time.

Data exclusions

NCA will automatically exclude unusable data points, i.e.:

- *Missing values*: For plasma models, if either the X- or Y-value is missing, the associated record is excluded from the analysis. For urine models, records with missing values for any of the following are excluded: collection interval start or stop time, drug concentration, or collection volume.
- *Data points preceding the dose time*: If an observation time is earlier than the dose time, that observation is excluded from the analysis. For urine models, the lower time point is checked. Dose time must be non-negative.
- *Data points for urine models*: In addition to the above rules, if the lower time is greater than or equal to the upper time, or if the concentration or volume is negative, the data must be corrected before NCA can run.

Data deficiencies that will result in missing values for PK parameters

WinNonlin reports missing values for PK parameters in the following cases.

Profile issue(s)	Parameters reported as missing
Profile with no non-missing observations after dose time	all final parameters
Bolus dosing and <=1 non-missing observation after dose time	all final parameters
Blood/plasma data with no non-zero observation values	All final parameters except Cmax, AUClast, AUCall, AUMClast
Blood/plasma data with all zero values except one non-zero value at dosing time	all final parameters
Urine data with all urine volumes equal to zero	all final parameters
Urine data with all concentrations equal to zero	All final parameters except Max_Rate, Amount_Recovered, Percent_Recovered

Multiple observations at the same time point

For NCA, WinNonlin permits only one observation per profile at each time point. If it detects more than one, it halts the NCA analysis and issues an error message. No output is generated.

AUC calculation and interpolation formulas

Linear trapezoidal rule

$$AUC|_{t_1}^{t_2} = \delta t \times \frac{C_1 + C_2}{2}$$

$$AUMC|_{t_1}^{t_2} = \delta t \times \frac{t_1 \times C_1 + t_2 \times C_2}{2}$$

Logarithmic trapezoidal rule

$$AUC|_{t_1}^{t_2} = \delta t \times \frac{C_2 - C_1}{\ln\left(\frac{C_2}{C_1}\right)}$$

$$AUMC|_{t_1}^{t_2} = \delta t \times \frac{t_2 \times C_2 - t_1 \times C_1}{\ln\left(\frac{C_2}{C_1}\right)} - \delta t^2 \times \frac{C_2 - C_1}{\ln\left(\frac{C_2}{C_1}\right)^2}$$

where δt is $(t_2 - t_1)$.

Note: If the logarithmic trapezoidal rule fails in an interval because $C_1 \leq 0$, $C_2 \leq 0$, or $C_1 = C_2$, then the linear trapezoidal rule will apply for that interval.

Linear interpolation rule (to find C^* at time t^*)

$$C^* = C_1 + \left| \frac{t^* - t_1}{t_2 - t_1} \right| (C_2 - C_1)$$

Logarithmic interpolation rule

$$C^* = \exp\left(\ln(C_1) + \left| \frac{t^* - t_1}{t_2 - t_1} \right| \times (\ln(C_2) - \ln(C_1))\right)$$

Note: If the logarithmic interpolation rule fails in an interval because $C_1 \leq 0$, $C_2 \leq 0$, or $C_1 = C_2$, then the linear interpolation rule will apply for that interval.

Partial areas

Rules for interpolation and extrapolation

Partial areas within observed data ranges are calculated as described under “[AUC calculation and interpolation formulas](#)” on page 185 and “[Calculation](#)

methods for AUC” on page 179. The following rules apply if some portion of the partial area falls after the last valid observation:

- If a start or end time falls before the first observation, the corresponding Y value is interpolated between the first data point and C0. C0=0 except in IV bolus models (model 201), where C0 is the dosing time intercept estimated by WinNonlin, and except for models 200 and 202 at steady state, where C0 is the minimum value between dose time and tau.
- If a start or end time falls within the range of the data but does not coincide with an observed data point, then a linear or logarithmic interpolation is done to estimate the corresponding Y, according to the AUC Calculation method selected in the Model Options dialog. (See “NCA settings” on page 179.)
- If a start or end time occurs after the last numeric observation (i.e., not “missing” or “BQL”) and Lambda Z is estimable, Lambda Z is used to estimate the corresponding Y:

$$\begin{aligned} Y &= \exp(\alpha - \text{Lambda-Z} * t) \\ &= \exp(\alpha - \text{Lambda-Z} * t_{\text{last}}) * \exp(-\text{Lambda-Z} * (t - t_{\text{last}})) \\ &= (\text{predicted concentration at } t_{\text{last}}) * \exp(-\text{Lambda-Z} * (t - t_{\text{last}})) \end{aligned}$$

- If a start or end time falls after the last numeric observation and Lambda Z is not estimable, the partial area will not be calculated.
- If both the start and end time for a partial area fall after the last numeric observation, then the log trapezoidal rule will be used, because a logarithmic decline must be assumed in order to obtain the predicted values.
- If the start time for a partial area is before the last numeric observation and the end time is after the last numeric observation, then the log trapezoidal rule will be used for the area from the last observation time to the end time of the partial area.

Partial area boundaries

The end time for the partial area must be greater than the start time. Both the start and end time for the partial area must be at or after the dosing time.

Therapeutic response windows

For therapeutic windows, one or two boundaries for concentration values can be given, and the program computes the time spent in each window determined by the boundaries and computes the area under the curve contained in each window. To compute these values, for each pair of consecutive data points, including the inserted point at dosing time if there is one, it is determined if a boundary crossing occurred in that interval. Call the pair of time values from the data set (t_i, t_{i+1}) and called the boundaries y_{lower} and y_{upper} .

If no boundary crossing occurred, the difference $t_{i+1} - t_i$ is added to the appropriate parameter TimeLow, TimeDur, or TimeHgh, depending on which window the concentration values are in. The AUC for (t_i, t_{i+1}) is computed following the user's specified AUC calculation method as described above in section 3. Call this AUC*. The parts of this AUC* are added to the appropriate parameters AUCLow, AUCDur, or AUCHgh. For example, if the concentration values for this interval occur above the upper boundary, the rectangle that is under the lower boundary is added to AUCLow, the rectangle that is between the two boundaries is added to AUCDur and the piece that is left is added to AUCHgh. This is equivalent to these formulae:

$$\text{AUCLow} = \text{AUCLow} + y_{\text{lower}} * (t_{i+1} - t_i)$$

$$\text{AUCDur} = \text{AUCDur} + (y_{\text{upper}} - y_{\text{lower}}) * (t_{i+1} - t_i)$$

$$\text{AUCHgh} = \text{AUCHgh} + \text{AUC}^* - y_{\text{upper}} * (t_{i+1} - t_i)$$

If there was one boundary crossing, an interpolation is done to get the time value where the crossing occurred; call this t^* . The interpolation is done by either the linear rule or the log rule following the user's AUC calculation method as described above in section 3, i.e., the same rule is used as when inserting a point for partial areas. To interpolate to get time t^* in the interval (t_i, t_{i+1}) at which the concentration value is y_w :

Linear interpolation rule for time:

$$t^* = t_i + [(y_w - y_i) / (y_{i+1} - y_i)] * (t_{i+1} - t_i)$$

Log interpolation rule for time:

$$t^* = t_i + [(\ln(y_w) - \ln(y_i)) / (\ln(y_{i+1}) - \ln(y_i))] * (t_{i+1} - t_i)$$

The AUC for (t_i, t^*) is computed following the user's specified AUC calculation method. The difference $t^* - t_i$ is added to the appropriate time parameter

and the parts of the AUC are added to the appropriate AUC parameters. Then the process is repeated for the interval (t^*, t_{i+1}) .

If there were two boundary crossings, an interpolation is done to get t^* as above for the first boundary crossing, and the time and parts of AUC are added in for the interval (t_i, t^*) as above. Then another interpolation is done from t_i to t_{i+1} to get the time of the second crossing t^{**} between (t^*, t_{i+1}) , and the time and parts of AUC are added in for the interval (t^*, t^{**}) . Then the times and parts of AUC are added in for the interval (t^{**}, t_{i+1}) .

The extrapolated times and areas after the last data point are now considered for the therapeutic window parameters that are extrapolated to infinity. For any of the AUC calculation methods, the AUC rule after t_{last} is the log rule, unless an endpoint for the interval is negative or zero in which case the linear rule must be used. The extrapolation rule for finding the time t^* at which the extrapolated curve would cross the boundary concentration y_w is:

$$t^* = (1/\lambda Z) * (\alpha - \ln(y_w)).$$

If the last concentration value in the data set is zero, the zero value was used in the above computation for the last interval, but now y_{last} is replaced with the following if λZ is estimable:

$$y_{last} = \exp(\alpha - \lambda Z * t_{last}). \text{ (only if the last concentration was zero)}$$

If y_{last} is in the lower window: InfDur and InfHgh values are the same as Dur and Hgh values. If λZ is not estimable, AUCInfLow is missing. Otherwise, $AUCInfLow = AUCLow + y_{last} / \lambda Z$.

If y_{last} is in the middle window for two boundaries: InfHgh values are the same as Hgh values. If λZ is not estimable, InfLow and InfDur parameters are missing. *Otherwise:* Use the extrapolation rule to get t^* at the lower boundary concentration. Add time from t_{last} to t^* into TimeInfDur. Compute AUC* from t_{last} to t^* , add the rectangle that is under the lower boundary into AUCInfLow, and add the piece that is left into AUCInfDur. Add the extrapolated area in the lower window into AUCInfLow. This is equivalent to:

$$t^* = (1/\lambda Z) * (\alpha - \ln(y_{lower}))$$

$$TimeInfDur = TimeDur + (t^* - t_{last})$$

$$AUCInfDur = AUCDur + AUC^* - y_{lower} * (t^* - t_{last})$$

$$AUCInfLow = AUCLow + y_{lower} * (t^* - t_{last}) + y_{lower} / \lambda Z$$

If y_{last} is in the top window when only one boundary is given, the above procedure is followed and the InfDur parameters become the InfHgh parameters.

If there are two boundaries and y_{last} is in the top window: If λZ is not estimable, all extrapolated parameters are missing; otherwise:

Use the extrapolation rule to get t^* at the upper boundary concentration value. Add time from t_{last} to t^* into TimeInfHgh. Compute AUC* for t_{last} to t^* , add the rectangle that is under the lower boundary into AUCInfLow, add the rectangle that is in the middle window into AUCInfDur, and add the piece that is left into AUCInfHgh. Use the extrapolation rule to get t^{**} at the lower boundary concentration value. Add time from t^* to t^{**} into TimeInfDur. Compute AUC** from t^* to t^{**} , add the rectangle that is under the lower boundary into AUCInfLow, and add the piece that is left into AUCInfDur. Add the extrapolated area in the lower window into AUCInfLow. This is equivalent to:

$$t^* = (1/\lambda Z) * (\alpha - \ln(y_{\text{upper}}))$$

$$t^{**} = (1/\lambda Z) * (\alpha - \ln(y_{\text{lower}}))$$

$$\text{TimeInfHgh} = \text{TimeHgh} + (t^* - t_{\text{last}})$$

$$\text{TimeInfDur} = \text{TimeDur} + (t^{**} - t^*)$$

$$\text{AUCInfHgh} = \text{AUCHgh} + \text{AUC}^* - y_{\text{upper}} * (t^* - t_{\text{last}})$$

$$\text{AUCInfDur} = \text{AUCDur} + (y_{\text{upper}} - y_{\text{lower}}) * (t^* - t_{\text{last}}) + \text{AUC}^{**} - y_{\text{lower}} * (t^{**} - t^*)$$

$$\text{AUCInfLow} = \text{AUCLow} + y_{\text{lower}} * (t^* - t_{\text{last}}) + y_{\text{lower}} * (t^{**} - t^*) + y_{\text{lower}} / \lambda Z$$

Sparse sampling computations

The [Noncompartmental Analysis](#) (NCA) module provides special methods to analyze concentration data with few observations per subject. The NCA treats this sparse data as a special case of plasma or urine concentration data. It first calculates the mean concentration curve of the data, by taking the mean concentration value for each unique time value for plasma data, or the mean rate value for each unique midpoint for urine data. For this reason, it is recommended to use nominal time, rather than actual time, for these analyses. The standard error of the data is also calculated for each unique time or midpoint value.

Using the mean concentration curve, NCA will calculate all of the usual plasma or urine final parameters listed under “[NCA output parameters and their computation formulas](#)” on page 510. In addition, it will use the subject information to calculate standard errors that will account for any correlations in the data resulting from repeated sampling of individual animals.

For plasma data (models 200-202), NCA will calculate the standard error for the mean concentration curve's maximum value (Cmax), and for the area under the mean concentration curve from dose time through the final observed time (AUCall). Standard error of the mean Cmax will be calculated as the sample standard deviation of the y-values at time Tmax divided by the square root of the number of observations at Tmax, or equivalently, the sample standard error of the y-values at Tmax. Standard error of the mean AUC will be calculated as described in [Nedelman and Jia \(1998\)](#), using a modification in [Holder \(2001\)](#), and will account for any correlations in the data resulting from repeated sampling of individual animals. Specifically:

$$SE(\hat{AUC}) = \sqrt{\text{Var}(\hat{AUC})}$$

Since AUC is calculated by the linear trapezoidal rule as a linear combination of the mean concentration values,

$$\hat{AUC} = \sum_{i=0}^m w_i \bar{C}_i$$

where

\bar{C}_i = the sample mean at time i

$$w_i = \begin{cases} (t_1 - t_0)/2, & i = 0 \\ (t_{i+1} - t_{i-1})/2, & i = 1, \dots, m-1 \\ (t_m - t_{m-1})/2, & i = m \end{cases}$$

m = last observation time for AUCall, or time of last measurable (positive) mean concentration for AUClast

it follows that

$$Var(\hat{AUC}) = \sum_{i=0}^m \frac{w_i^2 s_i^2}{r_i} + 2 \sum_{i < j} \frac{w_i w_j r_{ij} s_{ij}}{r_i r_j}$$

where

r_{ij} = number of animals sampled at both times i and j

r_i = number of animals sampled at time i

s_i^2 = sample variance of concentrations at time i

s_{ij} = sample covariance between concentrations C_{ik} and C_{jk} for all animals k that are sampled at both times i and j .

The above equations can be computed from basic statistics, and appear as equation (7.vii) in [Nedelman and Jia \(1998\)](#). When computing the sample covariances in the above, NCA uses the unbiased sample covariance estimator, which can be found as equation (A3) in [Holder \(2001\)](#):

$$s_{ij} = \sum_{k=1}^{r_{ij}} \frac{(C_{ik} - \bar{C}_i)(C_{jk} - \bar{C}_j)}{(r_{ij}-1) + \left(1 - \frac{r_{ij}}{r_i}\right)\left(1 - \frac{r_{ij}}{r_j}\right)}$$

For urine models (models 210-212), the standard errors are computed for Max_Rate, the maximum observed excretion rate, and for AUC_all, the area under the mean rate curve through the final observed rate.

For cases where a non-zero value C_0 must be inserted at dose time t_0 to obtain better AUC estimates (see “[Data checking and pre-treatment](#)” on page 183), the AUC estimate contains an additional constant term: $C^* = C_0(t_1 - t_0)/2$. In other words, w_0 is multiplied by C_0 , instead of being multiplied by 0 as occurs when the point (0,0) is inserted. An added constant in \hat{AUC} will not change $Var(\hat{AUC})$, so SE_AUClast and SE_AUCall also will not change. Note that the inserted C_0 is treated as a constant even when it must be esti-

mated from other points in the data set, so that the variances and covariances of those other data points are not duplicated in the $\text{Var}(\hat{\text{AUC}})$ computation.

For the case in which $r_{ij} = 1$, and $r_i = 1$ or $r_j = 1$, then s_{ij} is set to 0 (zero).

The AUC's must be calculated using one of the linear trapezoidal rules. (Select as described under “NCA settings” on page 179.)

Modeling

Model setup and output

Setup and output for compartmental modeling in WinNonlin are covered under the following headings.

1. “Loading the model”
2. “Model properties” on page 198
3. “Model options” on page 205
4. “Running the model” on page 211
5. “Output” on page 212

WinNonlin provides an extensive library of models, detailed under “[WinNonlin Model Libraries](#)” on page 507. Custom models are covered under “[User Models](#)” on page 219.

Loading the model

The WinNonlin uses two types of models: compiled (machine language) dynamic link library (*.DLL) files and ASCII text models (*.TXT or *.LIB). Both model types can be used in scripts or loaded via the PK/PD/NCA Analysis Wizard. ASCII models are human-readable and easily customized from within the WinNonlin interface. Compiled models provide faster performance. Custom compiled models can be created using FORTRAN or C++.

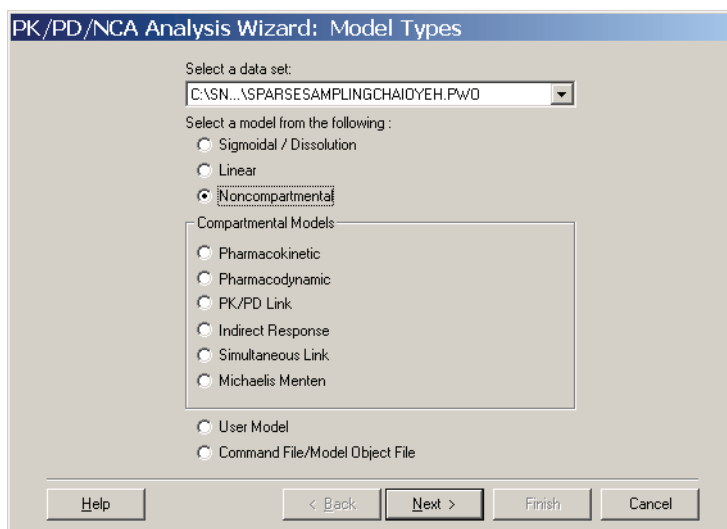
To load a model:

1. Open the data to which the model will be fit in a WinNonlin worksheet.



2. Click the **PK/PD/NCA Analysis Wizard** button on the tool bar, or choose **Tools>PK/PD/NCA Analysis Wizard** from the WinNonlin menus.
3. Make sure that the correct workbook is selected at the top of the dialog. The model will be associated with the active worksheet in that workbook.

Note: Hidden data sets are not available. (See “Hidden windows” on page 18.)



Note: The image above shows the wizard as it appears for a user who has an IVIVC licence installed. Other users will not see the IVIVC models.

4. Select the type of model to load from those listed in [Table 9-1](#). Click the links in the table for a listing of models and model details.

Table 9-1. WinNonlin model types

Model type	Description
Sigmoidal/dissolution models	Available only to users who have purchased and installed a WinNonlin IVIVC license. Dissolutions models to assist in evaluation of <i>in vivo-in vitro</i> correlations for formulation science.
Linear models	Models to perform linear regression.
Noncompartmental analysis	Geometric characterization of drug effect data or blood (plasma) or urine concentration data for IV infusion, IV bolus, or first-order input.

Table 9-1. WinNonlin model types (continued)

Model type	Description
Pharmacokinetic models	1 to 3 compartment models with 0 th or 1 st order absorption, with or without a lag time to the start of absorption.
Pharmacodynamic models	Eight variations on Emax and Sigmoid Emax models.
PK/PD link models	Any combination of WinNonlin's compiled PK and PD models, treating PK parameters as fixed and generating concentrations at the effect sites for use by the PD models.
Indirect response models	Four indirect response models, based on measured drug effect (inhibition or stimulation of response) and parameters defining either the build-up or dissipation of that effect. These models can be based on any WinNonlin PK model, and are available only in compiled form.
Michaelis-Menten models	Four Michaelis-Menten (saturable elimination) models, stored as ASCII library files in WINNONLN\LIB\MM.LIB.
Simultaneous PK/PD link models	Simultaneously fit PK and PD data, stored as ASCII libraries in WINNONLN\LIB. Each includes a different PK model and PD model 105.
User Models	Custom models, written and saved to an ASCII text file or compiled using C++ or FORTRAN, then loaded using this option.
Command file or Model Object	Legacy command files and WinNonlin Model Object (*.WMO) files from prior WNL versions and Pharsight Model Object (*.PMO) files can load ASCII or compiled user models, data and model settings.

Note: Only one model may be loaded in WinNonlin at a time.

5. For pharmacokinetic or pharmacodynamic models, specify whether to use the **ASCII** or a **Compiled** version. ASCII models appear in a WinNonlin text window and are easily customized. Compiled models run faster.
6. Click **Next**.
7. Select the model or, for ASCII models, load the model library file and enter the model number. See also “[Creating a new ASCII model](#)” on page 220.
8. Click **Next**.
9. For compiled user models and new ASCII user models, enter the model information as described under “[User model parameters](#)” on page 221. Click **Next**.

10. The Selection Summary dialog appears, with a description of the model. Click **Finish** to load the selected model. Proceed to set the [Model properties](#).

Model properties

Once a model has been loaded, the following can be set:

- [Data variables](#) tell the model which data columns to use.
- [Model parameters](#) set parameter names and initial parameter estimates.
- [Dosing regimen](#) or [Dosing data variables](#) set dose times and/or amounts, as applicable to the model.
- [Model options](#) select output formats, computational algorithms and units.

Data variables

[Modeling](#) requires that the model variables be identified before other settings.

To specify the roles of data columns in the model:

1. Load the model as described under “[Loading the model](#)” on page 195.
2. Choose **Model>Data Variables** from the WinNonlin menus or click the **Data Variables** button in the WinNonlin tool bar.
3. Select the independent variable in the Variables list and drag it to the X Variable field.
4. Select the dependent variable, and drag it to the Y Variable field.
5. Simulations: No Y variable is needed for simulations; instead check the **Simulate Data** check box and enter units (optional) for the Y variable.
6. Drag and drop to select one or more [sort variables](#) (optional) to identify individual data profiles. For example, a crossover study might include Subject and Treatment as sort variables. WinNonlin performs a separate analysis for each unique combination of sort variable values.
7. Select [carry along variables](#) (optional) to include in the analysis output. For example, Study Number or Gender might be included as a carry-along variable to make it available for post-modeling analyses.



8. Select the *function variable*, if applicable for the model. If a model fits multiple functions simultaneously, a function variable is required to tell WinNonlin which values of the Y variable apply to which function. For example, the data set may include data on two compartments—plasma and urine. The function variable might have the value 1 for plasma data, and 2 for urine data.

Note: Data corresponding to the first (sorted) value of the function variable will be used with the first function; the second, with the second function, etc.

9. Click **OK** and proceed to set the **Model parameters**.

Model parameters

All iterative estimation procedures require initial estimates of the parameters. WinNonlin can compute initial estimates via curve stripping for single-dose compiled PK, PD, PK/PD link, linear or IVIVC models. For all other situations, the user must supply initial estimates or provide boundaries to be used by WinNonlin in creating initial estimates via grid search.

For IVIVC models only (IVIVC license required), some parameters may be given fixed values while others are estimated. The FIXED command can be used for both modeling and simulating. Fixed parameters use a set value, so are no longer considered parameters. This reduces the number of parameters in the model. Statistics such as VIF are not computed for fixed parameters.

To set up the initial parameter estimates:



1. Select **Model>Model Parameters** from the menus or click the **Model Parameters** tool bar button. The Model Parameters tab of the **Model properties** dialog appears.
2. Specify the type of Parameter Calculation and Boundaries.
 - **User Supplied Initial Parameter Values.** Enter the initial estimate for each parameter in the column labeled Initial, and select WinNonlin or user-supplied bounds. When user-supplied boundaries are used, every parameter requires both upper and lower bounds. See the note, below.
 - **WinNonlin Generated Initial Parameter Values with WinNonlin Bounds:** WinNonlin will curve strip for initial estimates, then produce boundaries on the model parameters for model fitting. If curve stripping

fails, the program will terminate, as WinNonlin cannot grid search for initial estimates without user-supplied boundaries.

- **WinNonlin Generated Initial Parameter Values with User-Supplied Bounds:** WinNonlin will do curve stripping for initial estimates, then grid search if that fails.

Note: Parameter boundaries not only provide a basis for grid searching initial parameter estimates, but also limit the estimates during modeling. This can be valuable if the values become unrealistic or the model won't converge. See also [“Notes on initial parameter estimates and bounds”](#) below.

3. Fixed parameters. For IVIVC models, some (but not all) parameters may be set to fixed values rather than estimated. To fix a parameter for a given sort level, click in the cell in the Fixed or Estimated column and choose **Fixed** from the drop-down list. Enter the fixed value in the adjacent cell. To fix that parameter for all sort levels, click the **Fix all <Parameter Name>** button at the lower right.
4. **Propagate Final Estimates.** This option is available when sort variables are included in the [Data variables](#). When it is checked, initial estimates and bounds are entered or calculated only for the first sort level. The final parameter estimates from the first sort level provide the initial estimates for the second, and so forth for each successive sort level.
5. When all values are set, click **Apply** (to leave the dialog open and Proceed to the [Dosing regimen](#)) or **OK** (to close the dialog).
6. The **Save** and **Load** buttons can be used to store and re-use parameter settings in an external ASCII text file. The structure of this file (PARMFILE) is described in the chapter on Scripting. See [“Special file formats”](#) on page 434.

Multiple sort levels

If the data has one or more sort variables, initial estimates must be provided for each level of the sort variable(s) unless Propagate Final Estimates is selected. The same type of parameter calculation and boundaries apply for all sort levels.

To copy model parameter information from the first profile to the remaining profiles:

1. Enter values for the first profile, then select (highlight) all cells for the first profile with the mouse.
2. Hold the cursor in the lower right-hand corner of this selection until the cursor turns into a small black + called a *handle*.
3. Drag the handle over the target cells (to the bottom of the grid).
4. Release the mouse button.

Notes on initial parameter estimates and bounds

- The use of boundaries (user or WinNonlin supplied) is recommended.
- For any model other than a compiled WinNonlin model, if WinNonlin is requested to perform initial parameter estimates (i.e., grid searching is requested), boundaries are required.
- WinNonlin uses curve stripping only for single-dose data. If multiple dose data is fit to a WinNonlin library model and WinNonlin makes initial parameter estimates, a grid search is performed to obtain initial estimates. In this case, boundaries are required.
- For linear regressions, Method 3 (Gauss-Newton (Hartley)) and **Do not use bounds** are recommended.
- The grid used in grid searching includes three values for each parameter. Thus, if four parameters are to be estimated, the grid will comprise $3^4 = 81$ possible points. The point on the grid associated with the smallest sum of squares is used as an initial estimate for the estimation algorithm. Note that this option has been added for convenience, but it may dramatically slow the parameter estimation process if the model is defined in terms of differential equations.
- If the data are to be fit to a micro constant model, WinNonlin will perform curve stripping for the corresponding macro constant model then use the macro constants to compute the micro constants. It is still recommended that the lower and upper boundaries be specified, even when WinNonlin is computing the initial estimates.
- When the Gauss-Newton method is used to estimate the parameters (METHODS 2 and 3), the boundaries are incorporated by applying a normit transform to the parameter space. With the Nelder-Mead option (METHOD 1), the residual sum of squares is set to a large number if any

of the parameter estimates go outside the specified constraints at any iteration, which forces the parameters back within the specified bounds.

- Unlike the linearization methods, the use of bounds does not affect numerical stability when using the Nelder-Mead simplex option. The use of bounds with Nelder-Mead will, however, keep the parameters within the bounds. The use of bounds (either user-supplied or WinNonlin supplied) is always recommended with the Gauss-Newton methods.

Dosing regimen

WinNonlin uses model constants to store dosing information, including the number of doses, amount of the dose(s), and dosing time(s). Specific models require the information described below. For some models the number of constants depends upon the number of doses administered. Note that PD and IVIVC models do not require dosing.

Model Number					
Constant	1, 3-7, 11, 12	2, 9, 10, 19	8, 13, 14, 18	15, 16	17
1	# of doses	# of doses	stripping dose	bolus dose	stripping dose
2	dose 1	dose 1	# doses	IV dose	bolus dose
3	time of dose 1	start time 1	dose 1	infusion length	IV dose
4	dose 2	end time 1	time of dose 1		infusion length
5	time of dose 2	dose 2	dose 2		
6	(etc.)	start time 2	time of dose 2		
7		end time 2	(etc.)		
8		(etc.)			
9					
NCON	$(2 \cdot \text{ND}) + 1$	$(3 \cdot \text{ND}) + 1$	$(2 \cdot \text{ND}) + 2$	3	4

For the table above, NCON = the total number of constants to be specified by the user, and ND = number of administered doses.

WinNonlin dose information can be entered in the Regimen Dialog or read from worksheet columns using the Dosing Data Variables dialog (see [“Dosing data variables” on page 204](#)).

To enter dosing data in the Dosing Regimen dialog:

1. Select **Model>Dosing Regimen** from the WinNonlin menus or click the **Dosing Regimen** button in the tool bar. The Dosing Regimen tab of the **Model properties** dialog appears.

Compiled models

	Subject ID	Formulation	Constant	Value
1	DW	Capsule	Number of Doses	1
2	DW	Capsule	Dose #1	50
3	DW	Capsule	Time of Dose #1	0
4	DW	Tablet	Number of Doses	1
5	DW	Tablet	Dose #1	50
6	DW	Tablet	Time of Dose #1	0
7	GS	Capsule	Number of Doses	1

Dosing Units

Current Units: Restore

Mass Prefix: Mass Unit: Clear

Dose Normalization: Add

Buttons: OK, Cancel, Apply, Help, Copy, Paste, Clear All, Load..., Save...

2. If a sort variable has been specified, enter the constants required for all levels of the sort variable(s).
3. Enter units for the dose under Current Units, or select the unit prefix and unit below and click the **Add** button.

Note: If different profiles require different numbers of doses, enter the largest number, and enter doses of 0 for the extra doses. Include time values that fall within the data but beyond the last real dose for these phantom doses.

4. If dose amount is normalized by subject bodyweight or body mass index, select the appropriate factor in the Dose Normalization drop-list. For example, if doses are in milligrams per kilogram of bodyweight, enter Current Units of mg and Dose Normalization of kg. This will affect output parameter units. For example, if Volume units were L, dose normalization of kg would change those units to L/kg. Dose normalization affects units for all volume

and clearance parameters in NCA and PK models, as well as AUC/D in NCA plasma models, and Percent_Recovered in NCA urine models.

ASCII models

For ASCII models, the Constant column of the Dosing Regimen dialog contains the names of elements of the **CON** array. Check the ASCII text in the Model window to see how many and which constants to enter.

1. Use the **Insert Row**, **Add Row**, and **Delete Row** buttons, if needed, to size the grid appropriately for the number of constants required by the model.
2. If the data set includes multiple profiles, a row will be added for each profile.
3. Enter each constant value for each profile in the column labeled **Value**.

Note: Use the Save and Load buttons to save the dosing values and units to an external ASCII file and reload settings for re-use. The structure of this file is described under “[Special file formats](#)” on page 434

4. Click **OK** or **Apply** to enter the dosing information.

Dosing data variables

Use this dialog to extract dosing information from worksheet columns, instead of entering it manually in the dialog. See also:

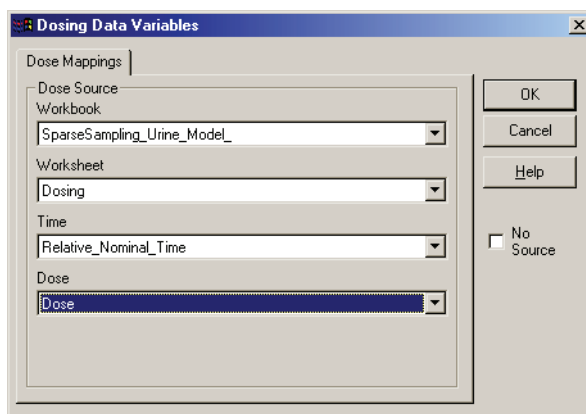
- “[Dosing regimen](#)” on page 202 for compartmental models
- “[Dosing regimen](#)” on page 173 for noncompartmental analysis

To select columns containing dose information:

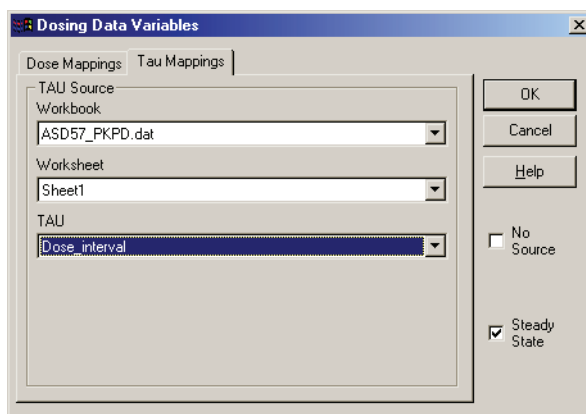
1. Load the data and model in WinNonlin, and set the model variables as described under “[Data variables](#)” on page 198.
2. Open the worksheet with dose information in WinNonlin.
3. Choose **Model>Dosing Data Variables...** from the WinNonlin menus or click the **Dosing Data Variables** tool bar button.
4. Un-check **No Source**.



5. Select the workbook and worksheet containing the dose information under Workbook and Worksheet, respectively.
6. Select the column containing dose times under Time.
7. Select the column containing dose amounts under Dose.



8. For noncompartmental analyses of steady-state data, check **Steady State** and open the Tau Mappings tab. Select the workbook, worksheet, and column containing the dose-interval values (Tau).



9. Click **OK**.

Model options

The options for compartmental modeling output and computational algorithm appear on five tabs in the Model Options dialog.

To set compartmental modeling options:

1. Select **Model Options** from the **Model** menu or click the **Model Options** tool bar button. The Model Options dialog appears.
2. Select the desired options using the following tabs.

Output options	Select text, chart (plots) and/or workbook output forms
Weight	Weighting of observation data.
Title	Text title to appear at the top of modeling output (text and charts)
PK settings	Model fitting algorithm
Units	Preferred output units for parameter estimates

3. Click **OK** or **Apply**.

Output options

The defaults output options are set in the WinNonlin Options dialog (see “WinNonlin options” on page 28). The Output Options tab of the **Model options** dialog, below, overrides the defaults for the current modeling session.

Check or un-check the following options to include or exclude the output.

Workbook: Tabular output in a workbook. For the contents of the workbook output, see “Output” on page 212.

Chart: This option generates plots showing predicted values versus observed values for each level of the ID variable.

Text: This provides an ASCII (text) version of the output, containing all modeling settings and output in one file, as well as any messages or errors that occur during modeling.

Note: When attempting a difficult fit, use the ASCII output to check for errors.

Page Breaks: Checking this starts a new page at each profile in text output.

Weight

For compartmental modeling, there are four ways to assign weights, other than uniform weighting:

1. *Weighted least squares* weights by a power of the observed Y.
2. *Iterative reweighting* sets weights as the predicted value of Y to a specified power.
3. *Include Weight as a variable on the data* file allows specific weight values to be applied to each observed value.
4. Programming statements can be used to define weights as part of a user defined model. See “[User Models](#)” on page 219.

See “[Weighting](#)” on page 243 for a detailed discussion of weighting schemes.

To set the weighting scheme for a modeling session:

1. Open the Weight tab of the [Model options](#) dialog (**Model>Model Options**).
2. Select from the options discussed below, and click **OK**.

Weights on File in Column: to read weights from a column on the data file.

Pre-Selected Scheme: to select from among the most common schemes:

- weight by 1/observed Y
- weight by 1/observed Y²
- weight by 1/predicted Y (iterative reweighting)
- weight by 1/predicted Y² (iterative reweighting)

Observed to the Power n: to use weighted least squares. Input the value of n in the box provided.

Predicted to the Power n: to use iterative reweighting. Input the value of n in the box provided.

Scaling of Weights: When weights are read from the data file or weighted least squares is used, the weights for the individual data values are scaled so that the sum of weights for each function is equal to the number of data values (with non-zero weights). This scaling has no effect on the model fitting, as the weights of the observations are proportionally the same. However, scaling of the weights provides increased numerical stability. See “[Scaling of weights](#)” on page 244 for further discussion.

To turn off scaling of weights, check the **No Scaling of Weights** check box.

Title

Titles can be displayed at the top of each chart and each page of ASCII text output. The title also appears in workbook output, only when it is printed. Unchecking the Title check box suppresses the titles without deleting the title text from the [Model options](#) dialog.

To include titles, check Title and enter the title text below. Enter multiple lines using either **CTRL + ENTER** or **SHIFT + ENTER** to move to the next line.

PK settings

The PK Settings tab of the [Model options](#) dialog (Model>Model Options) provides control over the model fitting algorithm and related settings.

Minimization

Select among the 3 minimization methods:

- [Method 1: Nelder-Mead simplex](#)
- [Method 2: Gauss-Newton with Levenberg and Hartley modification](#)
- [Method 3: Gauss-Newton with Hartley modification](#)

These methods are described in “[Model fitting algorithms and features of WinNonlin](#)” on page 5. The use of bounds is especially recommended with Methods 2 and 3. For linear regressions, use Method 3 without bounds.

Transpose final parameter table

If this option is checked, the Final Parameters output worksheet will show each parameter in a separate column. A second worksheet, called Non-transposed Final Parameters, will present each parameter in a separate row. If this option is not checked, the two worksheets will be reversed, with the Final Parameters worksheet displaying non-transposed data and a second worksheet, called Transposed Final Parameters containing the transposed version. The transposed form is particularly useful when creating final report tables or when summarizing the final parameters using descriptive statistics. See “[Models](#)” on page 29 for examples of transposed and non-transposed output.

Increment for partial derivatives

Nonlinear algorithms require derivatives of the models with respect to the parameters. The program estimates these derivatives using a difference equation. For example:

$$\frac{\partial F}{\partial P(1)} = \frac{F(P(1) + \Delta) - F(P(1))}{\Delta}$$

where Δ = the Increment multiplied by the parameter value.

Number of predicted values

This setting determines the number of points in the predicted data. Use this option to create a data set that will plot a smooth curve. When fitting or simulating multiple dose data, the predicted data plots may be much improved by increasing the number of predicted values. The minimum allowable value is 10; the maximum is 20,000.

If the number of derivatives is greater than 0 (i.e., differential equations are used), this command is ignored and the number of predicted points is equal to the number of points in the data set.

For compiled models without differential equations the default value is 1000. The default for user models is the number of data points in the original data set. This value may be increased only if the model has no more than one independent variable.

Note: To better reflect peaks (for IV dosing) and troughs (for extravascular, IV infusion and IV bolus dosing), the predicted data for the built-in PK models includes dosing times, in addition to the concentrations generated. For all three types, concentrations are generated at dosing times; in addition, for infusion models, data are generated for the end of infusion.

Convergence criterion

Use this option to set the convergence criterion. The default is 0.0001. Convergence is achieved when the relative change in the residual sum of squares is less than the convergence criterion.

Mean square

The variance is normally a function of the weighted residual SS/df, or the Mean Square. For certain types of problems, when computing the variance, the mean square needs to be set to a fixed value.

Iterations

Use this option to change the maximum number of iterations. For Methods 2 and 3 (Gauss-Newton), the default is 50. For Method 1 (Nelder-Mead simplex), the default is 500. For Method 1, each iteration is really a reflection of the simplex. If the number of iterations is set to zero, all output will be use the initial estimates as the final parameters.

Model size

Use this option to alter the default memory requirements when using ASCII models only. Model size may range from 1 to 64; the default size is 4.

Units

Specify the preferred units for output of modeling or simulation. The input data worksheet columns for the X-variable, Y-variable and, where applicable, dosing must contain units to use this option. WinNonlin performs appropriate conversions as needed to report output in the preferred units.

To specify preferred units:

1. Select the Units tab of the [Model options](#) dialog.
2. Select the row with the appropriate parameter name. The default units are displayed in the Default column.

Note: Linear or IVIVC models set preferred units for the independent (x) or dependent (y) variable. Units for output parameters will be derived from these. See [“Linear models” on page 550](#) or [“Sigmoidal/dissolution models” on page 551](#) for model details.

3. Enter the units in the Preferred column for that row. Alternately, click the **Units Builder** button to construct and specify units in the Units Builder.

4. In case of an error the parameters can be reset to their original setting by clicking on the **Reset to Defaults** button.
5. The preferred units can be saved to a (*.TXT) file for later re-use, using the **Save** and **Load** buttons.

Running the model

To start modeling:



1. Select **Model>Start Modeling** from the menus or click the **Start Modeling** tool bar button.

Note: The Select Data for Modeling dialog will appear if the original data set selected for modeling has become un-associated with the model. In this case, select the data set to be used for the modeling, and click **OK**.

Stop modeling

To stop modeling:



1. Click the **Stop Modeling** tool bar button.

Troubleshooting modeling problems

Occasionally when modeling, WinNonlin will terminate abnormally, or terminate normally and converge to unreasonable values of the parameter estimates. If these types of problems occur, apply the following steps in determining what may have gone wrong.

Note that it is possible that a specified model and input values may be correct, but there simply may not be enough information contained in the data to precisely estimate all of the parameters in the model. If that is the case, the only recourse is to obtain additional data.

Check the history file associated with the PK Data Output window. Error messages may appear here as guidance in resolving the problem. If no PK Data Output window appears, check the Text (ASCII) output for error messages.

If the model was defined by the user (i.e., not one of the models in WinNonlin's built-in library), carefully check the model definition statements to make sure that there are no undefined or uninitialized variables. Also make sure that the model has not attempted any illegal arithmetic computations such as taking the logarithm of zero or the square root of a negative number.

Carefully check the input data set and constants to make sure the values are correct. A plot of the data can be very helpful.

Check the initial values of the parameters to see if they are reasonable. One way to do this is to rerun the problem as a simulation. To do this, check the Simulate check box in the Y Variable group box in the Data Variables dialog box. Are the predicted values from the simulation reasonably close to the observed data?

Often error messages occur because WinNonlin is having difficulty fitting the model to the data. In addition, warning messages may also be generated to alert the user that WinNonlin is having to work very hard to obtain the parameter estimates, which may indicate that the model is ill-defined. The data set may be ill-conditioned or the initial parameter estimates may be too far from the true values. If these messages occur, try rerunning the problem and add LOWER and UPPER constraints. If the problem recurs, try using METHOD 1 (Nelder - Mead - via the Settings tab in the Model Options dialog box).

Errors

When modeling is run, error messages and warnings are written to the text output. If an error occurs that does not prevent completion of modeling, it will be written to the text output (if the user requested text output). If an error occurs that prevents completion of modeling, text output will be created, whether or not the user requests it; errors and warnings will be written there.

Note: To locate error messages, search for the word ERROR in the text output.

Output

One of the examples from the *Examples Guide* is used here to discuss the modeling output provided by WinNonlin. This example uses a pharmacokinetic (PK) model.

Note: This section is meant to provide guidance and references to aid in the interpretation of modeling output, and is not a replacement for a PK or statistics textbook.

After Example 1 is run, up to three new child windows appear in the WinNonlin window. They are discussed in the following sections.

- “Text (ASCII) output” on page 213: text version of all model settings and output, including any errors that occurred during modeling.
- “Workbook output” on page 216: worksheets listing input data, modeling iterations and output parameters, as well as several measures of fit.
- “Chart output” on page 217: plots of observed and predicted data, residuals, and other quantities, depending on the model run.

Text (ASCII) output

The text output window contains a complete summary of the modeling for this PK example, and any errors that occurred during modeling.

Model Commands

The first page is a copy of the input commands used to run the model. (See “Commands, Arrays and Functions” on page 561 for an alphabetic catalog of WinNonlin commands.) If the model is defined in a user library, then the statements that define the model are also printed.

Minimization Process

The second page varies, depending which method is used for minimization. If a Gauss-Newton method is used (Methods 2 or 3), this page shows the parameter values and the computed weighted sum of squared residuals at each iteration. In addition, the following two values are printed for each iteration:

- RANK - Rank of the matrix of partial derivatives of the model parameters. If the matrix is of full rank, RANK = # of parameters. If the RANK is less than the number of parameters, then the problem is ill-conditioned. That is, there is not enough information contained in the data to precisely estimate all of the parameters in the model.

- COND - Condition number of the matrix of partial derivatives. COND is the square root of the ratio of the largest to the smallest eigenvalue of the matrix of partial derivatives. If the condition number gets to be very large, say greater than $10e6$, then the estimation problem is very ill-conditioned. When using Methods 2 and 3, using bounds (LOWER and UPPER constraints) often helps reduce the condition number.

If the simplex algorithm is used, the second page only shows the parameter values and the weighted sum of squares for the initial, final (best, or smallest, sum of squares) point, and the next-to-best point.

Final Parameters

The third page of the output lists the parameter estimates, the asymptotic estimated standard error of each estimate, and two confidence intervals based on this estimated standard error. The confidence interval labeled UNIVARIATE is the parameter estimate plus and minus the product of the estimated standard error and the appropriate value of the t-statistic. The confidence interval labeled PLANAR is obtained from the tangent planes to the joint confidence ellipsoid of all the parameter estimates. For a complete discussion of the UNIVARIATE and PLANAR confidence intervals see page 95 of Draper and Smith (1981). The UNIVARIATE confidence intervals ignore the other parameters being estimated, while the PLANAR confidence intervals take into account the correlations among the parameter estimates. Only in the very rare event that the estimates are completely independent (uncorrelated) will the two confidence intervals be equal; usually the PLANAR intervals will be wider than the UNIVARIATE intervals. Both are 95% confidence intervals.

The estimated standard errors and confidence intervals are only approximate, as they are based on a linearization of the nonlinear model. The nearer the model is to being linear, the better is the approximation. See Chapter 10 of Draper and Smith (1981) for a complete discussion of the true confidence intervals for parameter estimates in nonlinear models.

The estimated standard errors are valuable for indicating how much information about the parameters is contained in the data. Many times a model will provide a good fit to the data in the sense that all of the deviations between observed and predicted values are small, but one or more parameter estimates will have standard errors that are large relative to the estimate.

Variance-Covariance Matrix, Correlation Matrix, and Eigenvalues

Page four of the output shows the correlation matrix of the estimates and the eigenvalues of the linearized form of the model. High correlations among one or more pairs of the estimates indicate that the UNIVARIATE confidence limits are underestimates of the uncertainty in the parameter estimates and, although the data may be well fit by the model, the estimates may not be very reliable. Also, data sets that result in highly correlated estimates are often difficult to fit, in the sense that the Gauss-Newton algorithm will have trouble finding the minimum residual sum of squares.

The eigenvalues are another indication of how well the data define the parameters. If the parameter estimates are completely uncorrelated, then the eigenvalues will all be equal. A large ratio between the largest and smallest eigenvalue may indicate that there are too many parameters in the model. Unfortunately, it is usually not possible to drop one parameter from a nonlinear model. For discussion of the use of eigenvalues in modeling see [Belsley, Kuh and Welsch \(1980\)](#).

Residuals

Page five of the output lists the observed data, calculated predicted values of the model function, residuals, weights, the standard deviations (S) of the calculated function values and standardized residuals. Runs of positive or negative deviations indicate non-random deviations from the model and are indicators of an incorrect model and/or choice of weights.

Also on page five are the sum of squared residuals, the sum of weighted squared residuals, an estimate of the error standard deviation, and the correlation between observed and calculated function values. Unlike the linear model case, the correlation is not a particularly good measure of fit; for most problems it will be greater than 0.9, and anything less than 0.8 probably indicates serious problems with the data and/or model. Two measures which have been proposed for comparing competing models, [AIC \(Akaike \(1978\)\)](#) and [SBC \(Schwarz \(1978\)\)](#), are also printed on page five.

If the data have been fit to a WinNonlin library model for extravascular or constant infusion input, area under the curve (AUC) is printed at the end of page five of the text output. The AUC is computed by the linear trapezoidal rule. If the first time value is not zero, then WinNonlin generates a data point with time and concentration values of zero, to compute AUC from zero to the last time. Note that this AUC in the text output differs from the AUC on the

Secondary Parameters tab of the tabular output. The secondary parameter AUC is calculated from the model parameters.

Note: Note that the AUC values may not be meaningful if the data were obtained after multiple dosing or IV dosing (due to time 0 extrapolation problems).

Secondary Parameters

If there are any secondary parameters, they, along with their estimated asymptotic standard errors, are printed on page six. The secondary parameters are functions of the primary parameters whose estimates are given on page three of the output. In the case of multiple-dose data, secondary parameters that depend on dose use the first dose in their computation. The standard errors of the secondary parameters are obtained by computing the linear term of a Taylor series expansion of the secondary parameters. As such, they represent about the third level of approximation and must be regarded with caution.

Workbook output

A PK Workbook window contains summary tables of the modeling data, a summary of the information in the ASCII output. These worksheets present the output in a form that can be used for reporting and further analyses.

Table 9-2. Summary of PK worksheets

Item	Contents
Initial Parameters	Parameter names, initial values, and lower and upper bounds for each level of the sort variables.
Minimization Process	Iteration number, weighted sum of squares, and value for each parameter, for each level of the sort variables.
Final Parameters	Parameter names, units, estimates, standard error of the estimates, CV%, univariate confidence intervals, and planar confidence intervals for each level of the sort variables.
Dosing	The dosing regimen specified for the modeling.
Correlation Matrix	A correlation matrix for the parameters, for each sort level.
Eigenvalues	Eigenvalues for each level of the sort variables.
Condition Numbers	Iteration number, rank and condition number for each sort level.

Table 9-2. Summary of PK worksheets (continued)

Item	Contents
Variance-Covariance Matrix	A variance-covariance matrix for the parameters, for each sort level.
Summary Table ^a	The sort variables, X, Y, transformed X, transformed Y, predicted Y, residual, weight, standard error of predicted Y, standardized residuals, for each sort level. For link models, also includes C_P and C_E . For indirect response models, also includes C_P .
Diagnostics	Diagnostics for each function in the model and for the total: corrected sum of squared observations (CSS), weighted corrected sum of squared observations (WCSS), sum of squared residuals (SSR), weighted sum of squared residuals (WSSR), estimate of residual standard deviation (S) and degrees of freedom (DF), the correlation between observed Y and predicted Y, the weighted correlation, and two measures of goodness of fit: the Akaike Information Criterion (AIC) and Schwarz Bayesian Criterion (SBC).
Differential Equations	Values of the differential equations at each time in the data set.
Partial Derivatives	The value of the partial derivatives for each parameter at each time point for each value of the sort variables.
Predicted Data	Time and predicted Y for multiple time points, for each sort level.
Secondary Parameters	Secondary parameter name, units, estimate, standard error of the estimate, and CV% for each sort level.
User Settings	Title, model number, weighting scheme, minimization method, convergence criterion, and maximum number of iterations allowed.

- a. If there are no statements to transform the data, then X and Y will equal X(obs) and Y(obs).

Note: Worksheets produced will depend on the analysis type and model settings.

These output worksheets are presented together in one workbook window. Use the tabs at the bottom of the window to select the worksheet to use.

Chart output

This PK analysis produces a chart window with five graphs for each level of the sort variable. Select graph type using the tabs at the bottom of the window.

X vs. Observed Y and Predicted Y: plots the predicted curves as a function of X , with the Observed Y overlaid on the plot. Used for assessing the model fit.

Observed Y vs. Weighted Predicted Y: plots weighted predicted Y against observed Y . Scatter should lie close to the 45 degree line.

Weighted Predicted Y vs. Weighted Residual Y: used to assess whether error distribution is appropriately modeled throughout the range of the data.

X vs. Weighted Residual Y: used to assess whether error distribution is appropriately modeled across the range of the X variable.

Partial Derivatives: plots the partial derivative of the model with respect to each parameter as a function of the x variable. If $f(x; a, b, c)$ is the model as a function of x , based on parameters a , b , and c , then

$$df(x; a, b, c)/da, df(x; a, b, c)/db, df(x; a, b, c)/dc$$

are plotted versus x . Each derivative is evaluated at the final parameter estimates. Data taken at larger partial derivatives are more influential than those taken at smaller partial derivatives for the parameter of interest.

User Models

Creating custom models using WinNonlin command language, FORTRAN, or C++

In addition to its library of standard PK and PK/PD models, WinNonlin supports two types of custom models: ASCII and compiled user models. Either type can be loaded via WinNonlin's PK/PD/NCA Analysis Wizard, and used in Pharsight Model Objects and scripts.

- **ASCII user models** are written using WinNonlin commands and programming language, and saved as text (*.TXT) or model library files (*.LIB).
- **Compiled user models** are created and compiled in Fortran or C++ into a dynamic link library (*.DLL) file. Compiling a model can significantly speed model fitting, especially for models with differential equations.

Example user models appear in this chapter and the WinNonlin *Examples Guide*. In addition, the ASCII and compiled models in the WinNonlin \LIB and \USERCOMP subdirectories serve as examples and templates. See also [Gabrielsson and Weiner \(2001\)](#).

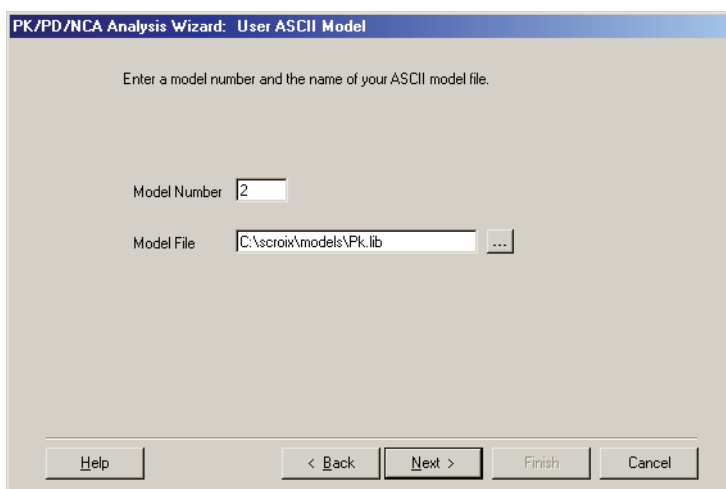
ASCII user models

ASCII user models are custom models with or without differential equations, written in WinNonlin commands. One or more models can be saved together in an ASCII text file (*.TXT) or library (*.LIB). Individual models may be up to 24 KB in size. An ASCII library cannot exceed 30 KB.

Within an ASCII model file, each model begins with a MODEL *n* statement identifying the model by number, and ends with an end-of-model (EOM) statement. For example, a library with two models might contain the following.

```
MODEL 1
<model statements>
EOM
MODEL 2
<model statements>
EOM
```

To load the second model into WinNonlin, you would use the PK/PD/NCA Analysis Wizard (see [“Loading the model” on page 195](#)), choose to load a User Model and Existing ASCII Model in the first two screens of the wizard, then enter the model number and the full file path as shown below.



Details on model templates and structure appear under:

- [“Creating a new ASCII model” below](#)
- [“Model structure” on page 222](#)
- [“ASCII model examples” on page 228](#)

Creating a new ASCII model

ASCII user models can be created in any text editor, including a WinNonlin text window or Notepad. To use a WinNonlin library model as a template, copy the library (*.LIB) file from the WinNonlin \LIB subdirectory and edit the copy as needed. Alternately, create a template with the appropriate number of parameters and equations using WinNonlin’s PK/PD/NCA Wizard as follows.

To create a new ASCII model from a template:

1. Launch the PK/PD/NCA Analysis Wizard by clicking on the **PK/PD/NCA Analysis Wizard** tool bar button or choosing **Tools>PK/PD/NCA Analysis Wizard** from the WinNonlin menus.
1. Select **User Model** in the Model Types dialog and click **Next**. The ASCII Model Selection dialog appears.
2. Select **Create New ASCII Model** and click **Next**. The Model Parameters dialog appears.
3. Enter the model information as detailed under “[User model parameters](#)”.
4. Click **Next**. The Selection Summary dialog appears.
5. Review the settings. Click **Back** to change settings or **Finish** to accept them.
6. WinNonlin will create a new text window containing a template for the model. Edit as needed using the structure, commands and syntax detailed in this chapter and under “[Commands, Arrays and Functions](#)” on page 561.
7. To save the model, save it as an ASCII file (*.LIB). The default model library file name is USERASCII.LIB. It is often convenient to save under a new name that describes the model. Several models can be saved to one .LIB file, so long as each uses a unique model number. To do this, open the library file in a text window, and copy and paste the new model text into the existing file.

User model parameters

In order to create a new ASCII model template or use a compiled user model, WinNonlin’s PK/PD/NCA Analysis Wizard needs to know the number of algebraic and differential equations in the model, and the number and names of estimated and secondary parameters. This information is entered in the User Model Parameters dialog of the Wizard as follows.

Note: To ascertain this information for compiled user models, look at the FORTRAN or C++ code from which the user model .DLL was compiled.

To complete the model parameters settings:

1. For ASCII models, enter the model number, an integer between 1 and 32767, to identify the model within a library. (See [Table 10-1](#) on page 223.)

2. Enter the number of algebraic functions in the model.
3. If the model includes differential equations, check **Include differential equations** and enter the number of differential equations.
4. Enter the primary (estimated) and secondary (optional) parameters in the grid. Parameter names can include up to 8 alphanumeric characters. For compiled models, enter parameters in the order in which they appear in the model.

Model structure

The following questions need to be considered in building custom models:

1. Is the model expressible as an algebraic nonlinear function, a system of differential equations, or a combination of the two?

Examples of algebraic nonlinear models	Examples of differential equations
$Y = \exp(-a \cdot t)$	$dz/dt = -k_1 t$
$Y = a \cdot \cos(t) + b \cdot \sin(t)$	$dz/dt = k_1 w - k_2 y$

2. Does the data set contain all the necessary information, or will the model require new variables that are functions of the input data?
3. Does the model change with values of the data or estimated parameters?
4. How many parameters and constants will be needed to define the model?
5. If the model uses differential equations, what are their initial values?

Information that is fixed across modeling sessions, such as the number and names of model parameters, should be included within the model to reduce the amount of information that must be supplied to WinNonlin when using the model. In contrast, use variables to represent model quantities that may vary across uses—e.g., dosage—to make the model as general as possible.

Note: The maximum number of variables and operators in a model is 2000.

An ASCII model structures this information in blocks, outlined in [Table 10-1](#).

Table 10-1. Model structure

Block:	Description
MODEL <i>N</i>	Required. Marks the beginning of the model. The model number, <i>N</i> , is required to load the model via the PK/PD/NCA Wizard, call it in a script, or store it in a multi-model library. The number must be an integer from 1 to 32767 and unique for each model in a library.
TRANSFORM statements END	Optional statements to alter data values and/or create new variables before the model is fit.
COMMANDS statements END	Optional WinNonlin commands to load the model, alter model settings, and begin analysis. Useful for automated (scripted) analyses.
TEMPORARY statements END	Optional commands to create global variables, i.e., variables that will be used in more than one model block. Variables that are defined within another block persist only within that block.
FUNCTION <i>N</i> statements END	Required. A function block must be included for each set of algebraic equations to be fit to data. Number each consecutively. The function number is required even if there is only one FUNCTION block.
STARTING statements END	Required if differential equations are included. Sets initial conditions for the system of differential equations.
DIFFERENTIAL statements END	Required if the model is a function of one or more differential equations. Statements defining the differential equations are included in this block.
SECONDARY statements END	Optional statements to define parameters that are functions of the estimated parameters.
EOM	Required. Marks the end of the model.

At minimum, a model requires a MODEL statement, FUNCTION block and EOM statement. It must begin with MODEL and end with EOM. The other blocks may appear in any order. Each block must end with an END statement.

Any block or command name can be represented by its first four characters or its full name. For example, either FUNCTION 1 or FUNC 1 signifies that Function 1 is being defined. Either REMARK or REMA indicates that the rest of that paragraph contains a note and is to be ignored by WinNonlin.

Note: Comments can appear within or between any of the blocks. Each individual comment line must begin with REMA (or [REMARK](#)).

In general, the blocks use WinNonlin programming statements and command language to define relationships between input data and parameter estimates, using special variables that WinNonlin makes available to hold the model input and output. Details appear under:

- “WinNonlin variables” on page 233
- “The WinNonlin programming language” on page 235
- “Commands, Arrays and Functions” on page 561

TRANSFORM

This block of statements is called once for each observation being modeled, prior to starting the estimation procedure. This block often defines the dependent variable (**Y**) or the weight (**WT**) as a function of other variables from the input data set. Independent variables can also be derived or transformed here. See the user models in EXP3.PMO, EXP4.PMO, and EXP9.PMO, stored in the \EXAMPLES directory, for examples that use the [TRANSFORM](#) block.

Note: When **WT** is specified in a model file, it will override any weighting specified via the Data Variables dialog. See also “[Weights in ASCII models](#)” on [page 245](#).”

If this block transforms a variable from the input data set, the output will include both the original variable and the transformed values. New variables created in the transform block will not be included. For example, if a data set contains the variables X8 and Y, and the following statements are included:

```
X8 = SQRT(X8)
Y=log(Y)
```

then the output worksheet will include the columns: X8 and Y (transformed values), and X8(Obs) and Y(Obs) (original values). However, if it included:

```
LOGY = log(Y)
```

then the variable LOGY will not be included in the output worksheet.

COMMANDS

The **COMMANDS** block associates WinNonlin commands with a model, to define model-related values such as NPAR (number of parameters). This block cannot include WinNonlin programming statements.

This block provides backward-compatibility with WinNonlin command files. Instead of using a **COMMAND** block, these items can be set using the WinNonlin interface, and saved in a Pharsight model object.

The **NPARAMETERS** command specifies the number of parameters to be estimated. **INITIAL**, **LOWER** and **UPPER** set initial estimates and bounds, and must include the number of values indicated by NPARAM.

The **NCONSTANTS** command sets the number of (dosing) constants in the model. Each constant must be initialized via either the **CONSTANTS** command or the WinNonlin dosing interface. Some models allow a variable number of constants. For example, the multiple dosing models in WinNonlin's library allow a variable number of doses per subject. In such cases, the constants should be specified at run time using WinNonlin's dosing interface.

The **NDERIVATIVES** command tells WinNonlin the number of differential equations in the model. **NFUNCTIONS** specifies the number of equation sets associated with observations, i.e., the number of **FUNCTION** blocks.

If a model will fit multiple functions that share one or more parameters, it may be more efficient to model the functions simultaneously, that is, within the same MODEL using multiple **FUNCTION** blocks. For example, when modeling both plasma and urine data for a one compartment IV bolus model it would be more efficient to model these simultaneously, since the equations for both involve K10. The equations for each compartment would appear in a separate **FUNCTION** block: one for plasma data, one for urine. The variable K10 would be defined in a **TEMPORARY** block for use in both functions.

When more than one function is defined (NFUN>1) it is important to define which observation data belong to each **FUNCTION** block. The input data set should contain all observations (Y) in one column. A *function* variable in a separate column indicates which observations belong with which function; its values must correspond to the **FUNCTION** numbers, i.e., 1 for FUNC 1, 2 for FUNC 2, etc. This variable is identified via WinNonlin's Data Variables dialog, or using the **FNUMBER** *N* command, where *N* is the column number.

The **NSECONDARY** command indicates the number of secondary parameters to be computed as functions of the estimated parameters. For example:

```
COMMANDS
NSEC 1
END
SECO
S(1)=0.693/K10
END
```

The **PNAMES**, and **SNAMES**, commands name the primary and secondary parameters. These names provide column labels in the output, and may be used in programming statements.

The **DNAMES** command may be used to define names for the columns in the input data. These names can then be used in the model definition statements.

The **PUNITS** and **SUNITS** commands supply optional units for the primary and secondary parameters. For user models, units are not included in calculation but are carried along to the output.

```
COMM
DNAMES 'SUBJECT', 'FORM', 'TIME', 'CONC'
NPARM 3
PNAMES 'K10', 'VOLUME', 'K1E'
PUNIT '1/hr', 'ml', '1/hr'
NSEC 1
SNAMES 'CL'
SUNIT 'ml/hr'
END
SECO
CL =0.693/K10
END
```

TEMPORARY

Statements in the optional **TEMPORARY** block define global variables; that is, variables that can be used by any block in the model. Variables that are used only within a given block can be defined within that block. Variable names include up to eight (8) letters, numbers or underscores, but must begin with a letter.

FUNCTION

Each set of equations that will be fit to a body of data requires a **FUNCTION** block. Most models include only one. The reserved variable F should be assigned the function value as:

- an algebraic function, e.g., $F = (D/V) \exp(-K1.0t)$, or
- a function of one or more differential equations defined in one or more **DIFFERENTIAL** blocks, e.g., $F = Z(1)$ or $F = Z(1) + 3Z(2)$.

```
FUNC 1
F=A*EXP(-ALPHA*T)
END
FUNC 2
F=Z(1)
END
```

This section can also define weights by setting **WT** equal to the weight.

STARTING

When a model includes differential equations (i.e., **NDERIVATIVES** > 0), a **STARTING** block is required to set initial values for each. Assign initial values to elements of the **Z** array as shown below. In most instances, starting values will equal the value of the dependent variable when $X = 0$. If the initial values do not occur at $X = 0$, then **X** should also be assigned accordingly. Note that the initial values could be estimated as parameters, such as **C0** below.

```
START
Z(1)=C0
Z(2)=0
END
```

DIFFERENTIAL

This block defines a system of differential equations using the array: **DZ(1)**, **DZ(2)**, etc. A **DIFFERENTIAL** block is required when **NDERIVATIVES** > 0.

```
DIFF
DZ(1)=-K12*Z(1)
DZ(2)=K12*Z(1) - K20*Z(2)
END
```

SECONDARY

In many instances, interest is not in the model parameters *per se*, but in functions of them. For example, a model may be defined micro constants, though clearance is of greater interest. The **SECONDARY** block addresses this.

```
SECO
REMARK Note that D and AUC must be defined above
S(1) = D/AUC
END
```

Assign secondary parameter values to the **S** array. S(1) denotes the first secondary parameter, S(2) the second, etc. Alternately, use **SNAMES** as follows.

```
COMM
NSEC 1
SNAME 'CL'
END
SECO
CL = D/AUC
END
```

ASCII model examples

Example 1

To fit a set of data to the following model:

$$C(t) = A\exp(-\alpha t) + B\exp(-\beta t)$$

A , α and β are parameters to be estimated, t denotes time and $C(0)$ is constrained to be zero (i.e., $A + B = 0$). B is not estimated, as $C(0) = 0$, implying that $B = -A$. This model will also estimate the half lives of α and β .

$$-\text{LN}(0.5)/\alpha \text{ and } -\text{LN}(0.5)/\beta$$

The following statements define the model and secondary parameters:

```
MODEL 1
FUNCTION 1
F=P(1)*EXP(-P(2)*X)-P(1)*EXP(-P(3)*X)
END
SECONDARY
S(1)=-LOGE(*.5)/P(2)
S(2)=-LOGE(*.5)/P(3)
END
EOM
```

The FUNCTION - END block defines function one (the only function). F denotes the concentration predicted by the model, as a function of X (which is time in this example) and three parameters A , α , and β , to be estimated.

The SECONDARY - END block defines secondary parameters (functions of the model parameters). $S(1)$ and $S(2)$ are the secondary parameters.

The above model will work fine. However, descriptive parameter names and comments could make the model easier to read, as follows.

```

MODEL 1
remark: Example problem
COMMAND
NPARM 3
NSEC 2
PNAMES 'A', 'ALPHA', 'BETA'
SNAMES 'ALPHA_HL', 'BETA_HL'
END
FUNC 1
B=-A
remark: Model is constrained so that C(0)=0
TIME=X
E1=EXP(-ALPHA*TIME)
E2=EXP(-BETA*TIME)
F=A*E1 + B*E2
END
remark: Secondary parameters
SECO
ALPHA_HL=-LOGE(*.5)/ALPHA
BETA_HL=-LOGE(*.5)/BETA
END
EOM

```

The above models are functionally equivalent. In the second model, the COMMAND - END block holds model commands. The PNAMES and SNAMES statements define the parameter and secondary parameter names that are used in the programming statements and the output.

It is useful to also store NSEC and NPAR with the model, so they needn't be set when loading the model to WinNonlin. The following statements would appear after the MODEL statement:

```

COMMANDS
NPAR 2
NSEC 2
END

```

Example 2

Consider a model characterized by the following differential equations.

$$\frac{dZ_1}{dt} = -K_{12} \cdot Z(1)$$

$$\frac{dZ_2}{dt} = K_{12} \cdot Z(1) - K_{20} \cdot Z(2)$$

with initial conditions $Z_1(0) = D$ and $Z_2(0) = 0$. The parameters to be estimated are K_{12} , K_{20} and D . As in the previous example, first use a PNAMES statement to assign meaningful names to the parameters and facilitate model definition.

```
MODEL 1
COMMAND
NPARM 3
PNAMES 'K12', 'K20', 'D'
END
```

Next, define the initial values for the differential equations.

```
START
Z(1) = D
Z(2) = 0
END
```

A FUNCTION block is required to define the model associated with the observed data. In this case Z_2 represents plasma concentration.

```
FUNC 1
F=Z(2)
END
```

The next set of statements completes the model specification.

```
DIFF
DZ(1) = -K12 * Z(1)
DZ(2) = K12 * Z(1) - K20 * Z(2)
END
EOM
```

Example 3

This example shows how to use the predefined variables to alter a model based on the current observation. Suppose there are two sets of data to fit simultaneously. Both data sets have the same model and parameters except that the second set requires a time lag. Assume the following data:

0.5	0.03	1
1.0	0.7	1
2.0	0.9	1
...	
0.5	0	2
1.0	0	2
2.0	0	2
3.0	0.02	2

The first column is time. The second is concentration. The third is an indicator variable. Its value indicates to which set of data the observations belong. To access its value during the model-fitting process, use the [DTA\(\)](#) variable. Since the third column contains the indicator values, the statement:

$$I = \text{DTA}(3)$$

assigns *I* the value of the indicator variable. The model reads as follows.

```

MODEL 1
COMM
NPAR 5
PNames 'A', 'B', 'ALPHA', 'BETA', 'LAG'
END
FUNC 1
I = DTA(3)
IF I = 1 THEN                                :First Set of Data
  T = X
  F = A*EXP(-ALPHA*T) + B*EXP(-BETA*T)
ELSE                                          :Second Set of Data
  T = X-LAG
  F = A*EXP(-ALPHA*T) + B*EXP(-BETA*T)
ENDIF
END
EOM

```

When $I = 1$ the function without the time lag is fit. When I is not 1, that is, $I = 2$, the function with the time lag is fit.

There is another way to fit these data. The two functions can be defined separately, in two FUNCTION blocks, using the **FNUMBER** command.

```
MODEL 1
COMM
NPAR 5
PNAMES 'A', 'B', 'ALPHA', 'BETA', 'LAG'
NFUNC 2
FNUM 3
END
FUNC 1
  T = X
  F = A*EXP(-ALPHA*T) + B*EXP(-BETA*T)
END
FUNC 2
  T = X-LAG
  F = A*EXP(-ALPHA*T) + B*EXP(-BETA*T)
END
EOM
```

These approaches produce the same parameter estimates but different output. The first creates one summary table; the second creates two, one per function.

WinNonlin variables

WinNonlin provides access to input data and parameter estimates during the model-fitting process through special variables, listed below. This allows “adaptive” models—that is, models that can be altered based on the current observation or value of the parameter estimates.

- | | |
|--------|--|
| DTA(N) | This array contains values of all columns from the input data file (in the same order) corresponding to the current observation. Thus, for a given observation,
DTA(1) = value of 1st variable (column)
DTA(2) = value of 2nd variable (column) etc.
These variables can also be accessed using DNAMEs. |
| F | The value of the function evaluated at the current observation, and current values of the parameters to be estimated, P(1), P(2), etc. |

X	Value of the independent variable at which the functions or differential equations are to be evaluated. This value is identical to DTA(XNUM) (See the XNUM command, p. 56.) If the model is a function of more than one independent variable, the independent variables can be accessed via the DTA() array, or using DNames.
P(N)	This array contains the current values of the parameters to be estimated. These variables can also be accessed using PNames.
CON(N)	This array contains the constants that have been input by the user (e.g., dose).
Z(N)	This array contains the current integrated values corresponding to a system of differential equations. These values are computed by WinNonlin (except when starting values for the system of differential equations are defined by the user).
DZ(N)	This array contains the current values of the differential equations evaluated at X, P() and Z().
S(N)	This array contains the values of the estimated secondary parameters. These variables can also be accessed using SNames.
WT	Weight assigned to the current observation. Values assigned to WT will override any weighting specified via the Data Variables dialog.

Temporary variables

Most models require other variables to define the model or parameters. These are created as global variables in the Temporary block or as local variables within any block. Variable names can include 1-8 letters, numbers and/or underscores and must start with a letter. Subscripts are not allowed, e.g., TIME(2) is invalid.

Reserved variable names

Variable names must not begin with:

- any WinNonlin keyword (e.g., TRAN, TEMP, FUNC, SECO, STAR, DIFF, COMM, OBJF, END, or EOM)
- any WinNonlin command or data array (e.g., CARR, CON, DTA, DTAARRAY(i,j), DTIM, F, NCAR, NDER, NCON, NOBS, NPARM, NPAU, NSEC, NTOT, NVAR, OBJFN, OBSN, P, PAUC, STEA, X, XARRAY(j), YARRAY(j), WTARRAY(j), Z).

Aliases

The commands [PNAMES](#), [SNAMES](#), and [DNAMES](#) can define aliases for parameters, secondary parameters, constants, and columns from the data file.

The WinNonlin programming language

WinNonlin provides a flexible programming language for ASCII user models. It provides control statements, mathematical functions and loop control. It is possible to define new variables for easier programming.

Valid mathematical operators are +, -, *, **, and /. Expressions are evaluated using standard operator precedence. Note that ** denotes exponentiation.

Statements

GOTO	Action	Transfers processing to a labeled statement.
	Syntax	GOTO <i>label</i> Where <i>label</i> is the statement label of an executable statement.
	Remarks	There should not be any blank spaces between GO and TO.
	Example	GOTO GREEN
LABEL	Action	Denotes a label. Control is transferred here via a GOTO statement.
	Syntax	The label must end with a colon (:).
	Example	GREEN:
IF THEN ELSE	Action	If <i>expression</i> is true, statements in the IF block are executed; if <i>expression</i> is false, control is transferred to the next ELSE, ELSE IF, or ENDIF statement at the same IF level.
	Syntax	IF <i>expression</i> THEN <i>statements</i> ELSE <i>statements</i> Where <i>expression</i> is a logical expression.
	Example	IF I>4 THEN GOTO GREEN ELSE GOTO BLUE ENDIF

	Remarks	IF statements can be nested up to 10 deep with the default model size (2), and up to 30 for model sizes 3 or greater.
ELSE	Action	Marks the beginning of an ELSE block.
	Syntax	ELSE This statement is optional.
	Remarks	An ELSE block consists of any executable statements between the ELSE statement and the next ENDIF statement at the same IF level.
	Example	See IF - THEN.
ENDIF	Action	Terminates an IF block.
	Syntax	
	Remarks	Each block of statements corresponding to each IF or ELSEIF statement must end with an ENDIF.
	Example	See IF - THEN.
DO ... NEXT	Action	Repeatedly executes the statements following the DO statement, through the NEXT statement that ends the loop.
	Syntax	DO <i>var</i> = <i>exp1</i> TO <i>exp2</i> [STEP <i>exp3</i>] Where <i>var</i> is a variable to be incremented during the looping. <i>exp1</i> is an expression whose value is the initial value of <i>var</i> . <i>exp2</i> is an expression whose value is the limit for the loop. STEP <i>exp3</i> is optional. If omitted, <i>exp3</i> is assumed to be 1. <i>exp3</i> is an expression whose value is the incrementor for the loop.
	Remarks	If <i>exp3</i> is positive, <i>exp2</i> is an upper limit for the loop. If <i>exp3</i> is negative, <i>exp2</i> is a lower limit for the loop. <i>Var</i> is always initialized to <i>exp1</i> , even if the loop is not executed (because the value of <i>exp1</i> is beyond the limit placed by <i>exp2</i>). After executing the last round through the loop, <i>var</i> has the value that caused the loop to exit (i.e., it is beyond the limit set by <i>exp2</i>). IF statements can be nested up to 10 deep with the default model size (2), and up to 30 for model sizes 3 or greater.
	Example	DO I=1 TO 3 DO J=1 TO 5 <i>statements</i> NEXT NEXT

Comparison operators

Table 10-2.

Operator	Definition	
AND		and
OR		or
GT	>	greater than
LT	<	less than
EQ	=	equal to
GE	>=	greater than or equal to
LE	<=	less than or equal to

Functions

Function Name	Result
LAG(<i>var</i>)	Returns the value of <i>var</i> at the last execution. <i>Var</i> can be X, Y, WT, or any temporary variable.
INT(<i>var</i>)	Truncates <i>var</i> to an integer value. E.G.: INT(3.9) = 3
EXP (X), DEXP (X)	e^x
ALOG (X), DLOG (X), LOGE (X)	natural log of X: ln(X)
ALOG10 (X), DLOG10 (X) LOG10 (X)	log base 10 of X: $\log_{10}(X)$
SQRT (X), DSQRT (X)	\sqrt{X}
SIN (X), DSIN (X)	sine (X)
COS (X), DCOS (X)	cosine (X)
MAX (X, Y), DMAX1 (X, Y)	Returns the maximum of X and Y.
MIN (X, Y), DMIN1 (X, Y)	Returns the minimum of X and Y.
ABS (X), DABS (X)	Returns the absolute value of X.
ATAN (X), DATAN (X)	$\tan^{-1}(X)$
ATAN2 (Y, X), DATAN2 (Y, X)	$\tan^{-1}(Y/X)$

Function names that are grouped together are interchangeable. For example, both EXP(X) and DEXP(X) will produce the same result, namely e^x .

Bioassay functions

Function name	Result
NORMIT (R,N)	<p>Let p denote R/N. This function returns the normit of p. That is, it returns the value NOR such that</p> $p = \int_{-\infty}^{NOR} \frac{1}{\sqrt{2\pi}} \exp\left(-Z^2/2\right) dz$ <p>Note: If R=0, then p is set to 1/2N. If R=N, then p is set to 1-1/2N</p>
WTNORM (NOR,N)	<p>This function returns the wt, WTNOR, associated with the normit (NOR) and a sample size N.</p> $WTNOR = N * Z^2 / (p * (1 - p))$ <p>where $Z = \frac{1}{\sqrt{2\pi}} \exp\left(-NOR^2/2\right)$</p> <p>and $p = \int_{-\infty}^{NOR} \frac{1}{\sqrt{2\pi}} \exp\left(-X^2/2\right) dx$</p>
LOGIT(P)	$\text{LOGIT} = \ln \frac{P}{(1-P)}$

Compiled user models

For [User Models](#) defined in terms of several differential equations, the model fitting process can be greatly accelerated by compiling the model into a 32-bit Windows dynamic link library (DLL) using either Compaq Visual FORTRAN 6.0 or Microsoft Visual C++ 6.0.

Each model requires two files:

1. The source code file (*.F90 or *.C).
2. The compiled model, *.DLL.

The source code file contains one subroutine specifying the model. This subroutine must be named `USRMOD ()`. Note, however, that the project should not be named `USRMOD`; `USRMOD.DLL` is a reserved name in WinNonlin.

Using FORTRAN

Creating a source file

The format of this file is very similar to that of WinNonlin [ASCII user models](#). The same blocks are included, but implemented in FORTRAN or C++ instead of WinNonlin programming language.

Two examples of the files required to build a DLL in FORTRAN have been installed to the \USERCOMP subdirectory of the WinNonlin installation directory. Either of these f90 files can be used as a template. The files include:

File	Purpose
F_UEXP6.F90	FORTRAN source file for a system of two differential equations with data on both compartments (same model as <code>EXAMPLES\EXP6.PMO</code>).
F_UEXP6.DLL	Compiled .DLL for the above FORTRAN source file.
F_UEXP15.F90	FORTRAN source file for a PK/PD link model (same model as <code>EXAMPLES\EXP15.PMO</code>).
F_UEXP15.DLL	Compiled .DLL for the above FORTRAN source file.

Example using Digital (now Compaq) Visual FORTRAN 6.0

To create a compiled user model in FORTRAN:

1. Open Visual Fortran.
2. Choose **File>New** from the menus.
3. Select **Fortran Dynamic Link Library** as the kind of product to create.
4. Enter a project name at the top left.
5. Click **OK**.

6. In the next dialog select **Empty dll application**.
7. Click **Finish**.
8. Click **OK**.
9. Choose **Project>Add to Project...>New** from the Visual FORTRAN menus.
10. Select **Fortran Free Format Source File**.
11. Enter a file name. This example uses `modelName.f90`.

Note: The source file stored on disk can be assigned any name, but the subroutine must be named `USRMOD()`.

12. Click **OK** to open a frame for the Fortran file.
13. Enter the model code. To use one of the provided templates, open `USER-COMP\F_UEXP6.F90` or `USERCOMP\F_UEXP6.F90` (located in the WinNonlin installation directory) in a text editor. Copy/paste the content into your new FORTRAN file and edit as needed. See the following for model structure and syntax:
 - [“Model structure” on page 222](#)
 - [“WinNonlin variables” on page 233](#)
 - [“Commands, Arrays and Functions” on page 561](#)
14. Choose **Build>Set Active Configuration...** from the menus. Select **Win32 Release** and click **OK**. This minimizes the size of the file.
15. Choose **Build>Build mymodel.dll** from the menus. The .DLL is saved to the Release subdirectory in the current directory; for this example:
`D:\MYPROJECTS\MYMODEL\RELEASE\MYMODEL.DLL`.

Using C++

Creating a source file

The format of this file is very similar to a WinNonlin ASCII model. The same blocks are included, with a C++ implementation. An example of this file is provided for Microsoft® Visual C++ 6.0.

The files required to build a DLL in Visual C++ have been installed to the \USERCOMP subdirectory. Two sets of files have been provided. Either can be used as a template. These files include:

File	Purpose
C_UEXP6.C	C source file for modeling example 6.
C_UEXP6.DLL	Compiled .DLL file for modeling example 6.
C_UEXP15.C	C source file for modeling example 15.
C_UEXP15.DLL	Compiled .DLL file for modeling example 15.

Note: To demonstrate the benefits of compiling models, run Example 15 in both the ASCII form and in the compiled form.

Example using Microsoft Visual C++ 6.0

To compile a user model:

1. Open Microsoft® Visual C++ v. 6.0.
2. Select **New** from the **File** menu.
3. Select **Win32 Dynamic Link Library**.
4. Enter a project name.
5. Press **OK**.
6. In the next dialog select **An empty dll project**.
7. Click **Finish**.
8. Click **OK**.
9. On the **Project** menu select **Add to Project...New**.
10. Select **C++ Source File**.
11. Enter a file name. This example uses `modelname.c`.

Note: It is important to use the `.c` file extension rather than the `.cpp` extension.

12. A frame opens for the program. Enter the code for the model. Use the templates provided (see table above).
13. Select from the **Build** menu **Set Active Configuration...Win32 Release**. This is not essential, but it builds a smaller DLL file.
14. Select from the **Build** menu **Build mymodel.dll**. The DLL is built in the specified directory (D:\MYPROJECTS\MYMODEL\RELEASE\MYMODEL.DLL)

Note: Statements in the Visual C++ source file with // starting in the first column are comments.

To alter this example for another model:

1. C++ makes no assumptions about types of variables. All integers should be declared as INT and all real numbers should be declared as DOUBLE.
2. Replace “k12 = p[0]; ... etc.” with the parameters for the model as they would be stated in a Temporary block.
3. For each block, other than the Command block, include the corresponding block from the template file.

The source file stored on disk can be assigned any name, but the subroutine must be called USRMOD ().

Weighting

Weighted nonlinear regression

WinNonlin provides flexibility in performing weighted nonlinear regression and using weighted regression. Weights are assigned after loading a model, via the Model Options dialog (see “[Model options](#)” on page 205 for compartmental models and “[Model options](#)” on page 175 for NCA).

There are three ways to assign weights, other than uniform weighting, through the WinNonlin user interface:

1. [Weighted least squares](#): weight by a power of the observed value of Y
2. [Iterative reweighting](#): weight by a power of the predicted value of Y
3. [Reading weights from the data file](#): include weight as a column in the data

For a user ASCII model, programming statements may also be used to define weights as part of the model.

Weighted least squares

When using weighted least squares, WinNonlin weights each observation by the value of the dependent variable raised to the power of n, i.e., $WEIGHT = Y^n$. For example, selecting this option, and setting $n = -0.5$ instructs WinNonlin to weight the data by the square root of the reciprocal of observed Y, i.e.,

$$1/(\sqrt{Y})$$

If n has a negative value, and one or more of the Y values are less than or equal to zero, then the corresponding weights are set to zero.

The program scales the weights such that the sum of the weights for each function equals the number of observations with non-zero weights. See “[Scaling of weights](#)” on page 244.

Iterative reweighting

Iterative Reweighting redefines the weights for each observation to be F^n , where F is the predicted response. For example, selecting this option, and setting $n = -0.5$ instructs WinNonlin to weight the data by the square root of reciprocal of the predicted value of Y , i.e.,

$$1/(\sqrt{Y})$$

As with [Weighted least squares](#), if N is negative, and one or more of the predicted Y values are less than or equal to zero, then the corresponding weights are set to zero.

Iterative reweighting differs from weighted least squares in that for weighted least squares the weights are fixed. For iteratively reweighted least squares the parameters change at each iteration, and therefore the predicted values and the weights change at each iteration.

For certain types of models, iterative reweighting can be used to obtain maximum likelihood estimates (see the article by Jennrich and Moore in the References).

Reading weights from the data file

It is also possible to have a variable in the data file that has as its values the weights to use. The weights should be the reciprocal of the variance of the observations. As with Weighted Least Squares, the program will scale the weights such that the sum of the weights for each function equals the number of observations with non-zero weights (see “[Scaling of weights](#)”).

Scaling of weights

When weights are read from the data file or when weighted least squares is used, the weights for the individual data values are scaled so that the sum of the weights for each function is equal to the number of data values (with non-zero weights).

The scaling of the weights has no effect on the model fitting as the weights of the observations are proportionally the same. However, scaling of the weights provides increased numerical stability.

Consider the following example:

X	Y	Y^2	Weight assigned by WinNonlin
1	6	0.0278	1.843
10	9	0.0123	0.819
100	14	0.0051	0.338
Sum		0.0452	3.000

Suppose weighted least squares with the power -2 was specified (i.e., weighting by $1/Y*Y$). The corresponding values are as shown in the third column. Each value of Y^{-2} is divided by the sum of the values in the third column, and then multiplied by the number of observations, e.g.,

$$(0.0277778/0.0452254) * 3 = 1.8426238, \text{ or } 1.843 \text{ after rounding.}$$

Note that $0.0278/0.0051$ is the same proportion as $1.843/0.338$.

Note: Scaling can be turned off by selecting the **No Scaling of Weights** option on the Weight tab of the Model Options dialog or by using the NOSCALE command in a command file.

Weights in ASCII models

When writing a custom ASCII model, it is possible to define the weights using programming statements at the same time the model is defined. *WT* is a special variable name in WinNonlin. If the variable *WT* is defined with programming statements, this variable overrides any weighting specified in the WinNonlin Model Options dialog. Consider the following example:

```
MODEL 1
TRANSFORM
IF X > 10 THEN
WT = 1/Y
ELSE
WT = 1/(Y*Y)
ENDIF
END
(additional statements here)
```

The above statements would set the weight of the observation equal to $1/Y$ if the value of X is greater than 10, and $1/Y^2$ if X is less than or equal to 10. As the values of Y are constant across iterations, the corresponding weights are also constant. However, it is also possible to use programming statements to iteratively reweight the observations, as shown in the following example:

```
MODEL 1
FUNC 1
F = A*EXP(-ALPHA*X) - A*EXP(-BETA*X)
WT = 1/(F*F)
END
(additional statements here)
```

This is equivalent to specifying Predicted to power $n = -2$. (See [“Iterative reweighting” on page 244.](#))

Each function to be fit can use a different weighting scheme. For example:

```
MODEL 1
TEMP
(additional statements here)
END
FUNC 1
(additional statements here)
WT = (some function)
END
FUNC 2
(additional statements here)
WT = (a different function)
END
(additional statements here)
```

The WinNonlin Toolbox

Nonparametric Superposition, Semicompartmental Modeling and Crossover Design

The WinNonlin toolbox includes the following tools:

- **Nonparametric superposition:** used to predict drug concentrations after multiple dosing at steady state.
- **Semicompartmental modeling:** estimates effect-site concentration given plasma concentration and a value for K_{e0} .
- **Crossover design:** performs nonparametric statistical tests; calculates confidence intervals for treatment median and median difference between treatments, and estimates relevance of direct, residual, and period effects.
- **Numeric deconvolution:** evaluating *in vivo* drug release and delivery based on data for a known drug input

Example are provided in the *Examples Guide*.

Nonparametric superposition

In pharmacokinetics it is often desirable to predict the drug concentration in blood or plasma after multiple doses, based on concentration data from a single dose. This can be done by fitting the data to a compartmental model with some assumptions about the absorption rate of the drug. An alternative method is based on the principle of superposition, which does not assume any pharmacokinetic (PK) model.

WinNonlin's nonparametric superposition function is used to predict drug concentrations after multiple dosing at steady state, and is based on noncompartmental results describing single dose data. The predictions are based upon an accumulation ratio computed from the terminal slope (λ_Z). The feature allows predictions from simple (the same dose given in a constant inter-

val) or complicated dosing schedules. The results can be used to help design experiments or to predict outcomes of clinical trials when used in conjunction with the semicompartamental modeling function. See [“Performing nonparametric superposition” on page 250](#) for stepped instructions

Data and assumptions

Nonparametric Superposition assumes that each dose of a drug acts independently of every other dose; that the rate and extent of absorption and average systemic clearance are the same for each dosing interval; and that linear pharmacokinetics apply, so that a change in dose during the multiple dosing regimen can be accommodated.

In order to predict the drug concentration resulting from multiple doses, one must have a complete characterization of the concentration-time profile after a single dose. That is, it is necessary to know $C(t_i)$ at sufficient time points t_i , ($i=1,2,\dots,n$), to characterize the drug absorption and elimination process. Two assumptions about the data are required: independence of each dose effect, and linearity of the underlying pharmacokinetics. The former assumes that the effect of each dose can be separated from the effects of other doses. The latter, linear pharmacokinetics, assumes that changes in drug concentration will vary linearly with dose amount.

The required input data are the time, dosing, and drug concentration. The drug concentration at any particular time during multiple dosing is then predicted by simply adding the concentration values as shown below.

Note: User-defined terminal phases apply to all sort keys. In addition, dosing schedules and doses are the same for all sort keys.

Computation method

Given the concentration data points $C(t_i)$ at times t_i , ($i=1,2,\dots,n$), after a single dose D , one may obtain the concentration $C(t)$ at any time t through interpolation if $t_1 < t < t_n$, or extrapolation if $t > t_n$. The extrapolation assumes a log-linear elimination process; that is, the terminal phase in the plot of $\log(C(t))$ versus t is approximately a straight line. If the absolute value of that line's slope is λ_z and the intercept is $\ln(\beta)$, then $C(t) = \beta \exp(-\lambda_z t)$ for $t > t_n$.

The slope λ_z and the intercept are estimated by least squares from the terminal phase; the time range included in the terminal phase is specified by the user. If

the user does not specify the terminal phase, estimates from the best linear fit (based on adjusted R -square as in noncompartmental analysis) will be used. The half life is $\ln(2)/\lambda_z$.

Suppose there are m additional doses $D_j, j = 1, \dots, m$, and each dose is administered after τ_j time units from the first dose. The concentration due to dose D_j will be:

$$C_j(t) = (D_j / D) * C(t - \tau_j)$$

where t is time since the first dose and $C(t - \tau_j) = 0$ for $t \leq \tau_j$.

The total predicted concentration at time t will be:

$$\text{Conc}(t) = \sum_j C_j(t), \quad j = 1, 2, \dots, m$$

If the same dose is given at constant dosing intervals τ , and τ is sufficiently large that drug concentrations reflect the post-absorptive and post-distributive phase of the concentration-time profile, then steady state can be reached after sufficient time intervals. Let the concentration of the first dose be $C_1(t)$ for $0 < t < \tau$, so τ is greater than t_n . Then the concentration at time t after n^{th} dose (i.e., t is relative to dose time) will be:

$$\begin{aligned} C_n(t) &= C_1(t) + \beta \exp[-\lambda(t+\tau)] + \beta \exp[-\lambda(t+2\tau)] \\ &\quad + \dots + \beta \exp[-\lambda(t+(n-1)\tau)] \\ &= C_1(t) + \beta \exp[-\lambda(t+\tau)] \{1 - \exp[-\lambda(n-1)\tau]\} / [1 - \exp(-\lambda\tau)] \end{aligned}$$

As $n \rightarrow \infty$, the steady state (ss) is reached:

$$C_{ss}(t) = C_1(t) + \beta \exp[-\lambda(t+\tau)] / [1 - \exp(-\lambda\tau)]$$

To display the concentration curve at steady state, WinNonlin assumes steady state is at ten times the half life.

For interpolation, WinNonlin offers two methods: linear interpolation and log-interpolation. Linear interpolation is appropriate for log-transformed concentration data and is calculated by:

$$C(t) = C(t_{i-1}) + [C(t_i) - C(t_{i-1})] * [t - t_{i-1}] / [t_i - t_{i-1}],$$

$$t_{i-1} < t < t_i$$

Log-interpolation is appropriate for original concentration data, and is evaluated by:

$$C(t) = \exp \{ \log(C(t_{i-1})) + [\log(C(t_i)) - \log(C(t_{i-1}))] * (t - t_{i-1}) / (t_i - t_{i-1}) \}, \quad t_{i-1} < t < t_i$$

For additional reading see Appendix E of [Gibaldi and Perrier \(1982\)](#).

Performing nonparametric superposition

Note: All quantities in the Nonparametric Superposition dialog should be entered in the units of the input data set. If units appear in the input worksheet, those units are applied in nonparametric superposition, and appear in the output.

To perform nonparametric superposition:

1. Open the data set providing the single-dose concentrations.
2. Choose **Tools>Nonparametric Superposition** from the WinNonlin menus. The Nonparametric Superposition dialog appears. Make sure the appropriate data set appears under Dataset, and its variables appear under Variables.

3. *Time for First Dose*: Drag the Time column to this field.
4. *Concentration for First Dose*: Drag the Concentration column to this field.
5. *Sort variables*: If using sort variables to define individual data profiles, drag one or more columns to the sort variables field. A separate analysis is performed for each unique combination of sort variable values.
6. *Number of output data points*: Enter the number of predicted values to output.

7. *Method for computations*: Select linear or logarithmic interpolation and extrapolation. Generally, select **Linear** for log-transformed concentration data; **Log** for untransformed data.
8. *Regular dosing*: For dosing at a regular time interval:
 - a. Enter the initial dose in the **Initial Dose** field of the Regular Dosing tab.
 - b. Enter the maintenance dose.
 - c. Enter the value for **Tau**, the dosing interval, in time units matching those of the time variable in the data set.
 - d. Select whether to display the chart at steady state or at a particular dose.
9. *Variable Dosing*: If the time between doses varies:
 - a. Enter individual doses and dose times on the Variable Dosing tab.
 - b. For a repeating dose regimen, enter doses and times for one cycle, check **Repeat every n time units** and enter the repeat time, n .
 - c. Enter time range for which to output predicted data.
10. *Define Terminal Phase*: (Optional) To set the time range for Lambda Z calculations, check this check box and enter start and end times for the terminal elimination phase.
11. Click **Calculate**. WinNonlin checks for duplicate time values in the data and prompts for sort variables if they have not been specified. WinNonlin performs the calculations and generates two new windows, a Nonparametric Superposition Workbook and a Nonparametric Superposition Chart.

Output

Nonparametric Superposition generates a workbook containing predicted concentrations and Lambda Z values, and a chart window containing plots of predicted concentration over time for each sort level.

The workbook contains two worksheets: Concentrations and Lambda Z. The Concentrations worksheet contains times and predicted concentrations for each level of the sort variables. If a regular dosing schedule is selected, this output represents times and concentrations between two doses at steady state. If a variable dosing schedule is selected, the output includes the times and concentrations within the supplied output range. The Lambda Z worksheet contains the Lambda Z value and the half-life for each sort key.

Semicompartmental modeling

Semicompartmental modeling was proposed by [Kowalski and Karim \(1995\)](#) for modeling the temporal aspects of the pharmacokinetic-pharmacodynamic relationship of drugs. This model was based on the effect-site link model of [Sheiner, Stanski, Vozeh, Miller and Ham \(1979\)](#) to estimate effect-site concentration C_e by using a piecewise linear model for plasma concentration C_p rather than specifying a PK model for C_p . The potential advantage of this approach is reducing the effect of model misspecification for C_p when the underlying PK model is unknown.

WinNonlin's semicompartmental modeling estimates effect-site concentrations for given times and plasma concentrations and an appropriate value of k_{eo} . This function should be used when a counterclockwise hysteresis is observed in the graph of effect versus plasma concentrations. The hysteresis loop collapses with the graph of effect versus effect-site concentrations.

A scientist developing a PK/PD link model based upon simple IV bolus data can use this function to compute effect site concentrations from observed plasma concentrations following more complicated administration regimen without first modeling the data. The results can then be compared to the original data sets to determine if the model suitably describes pharmacodynamic action after the more complicated regimen. See [“Performing semicompartmental modeling” on page 254](#) for stepped instructions

Data and assumptions

Drug concentration in plasma C_p for each subject is measured at multiple time points after the drug is administered. To minimize the bias in estimating C_e , the time points need to be adequately sampled such that accurate estimation of the AUC by noncompartmental methods can be obtained. Effect-site link model is used and the value of the equilibration rate constant k_{eo} that accounts for the lag between the C_p and the C_e curves is required. A piecewise log-linear model is assumed for C_p .

Computation method

In the effect-site link model ([Sheiner, Stanski, Vozeh, Miller and Ham \(1979\)](#)), a hypothetical effect compartment was proposed to model the time lag between the PK and PD responses. The effect site concentration C_e is related to C_p by first-order disposition kinetics and can be obtained by solving the differential equation:

$$\frac{dC_e}{dt} = k_{e0}(C_p - C_e)$$

where k_{e0} is a known constant. In order to solve this equation, one needs to know C_p as a function of time, which is usually given by compartmental PK models.

Here, use a piecewise linear model for C_p :

$$C_p(t) = C_{p_{j-1}} + \lambda_j (t - t_{j-1}) \quad t_{j-1} < t < t_j$$

where $\lambda_j = (C_{p_j} - C_{p_{j-1}})/(t_j - t_{j-1})$ and $C_{p_{j-1}} = C_p(t_{j-1})$.

Using the above two equations can lead to a recursive equation for C_e :

$$C_{e_j} = C_{e_{j-1}} e^{-k_{e0}(t_j - t_{j-1})} + \left(C_{p_{j-1}} - \frac{\lambda_j}{k_{e0}} \right) [1 - e^{-k_{e0}(t_j - t_{j-1})}] + \lambda_j (t_j - t_{j-1})$$

The initial condition is $C_p(0) = C_e(0) = 0$.

One can also assume a log-linear piecewise PK model for C_p

$$C_p(t) = C_{p_{j-1}} \exp[-\lambda_j (t - t_{j-1})] \quad t_{j-1} < t < t_j$$

here $\lambda_j = (\ln C_{p_{j-1}} - \ln C_{p_j})/(t_j - t_{j-1})$, and:

$$C_{e_j} = C_{e_{j-1}} e^{-k_{e0}(t_j - t_{j-1})} + C_{p_{j-1}} \frac{k_{e0}}{k_{e0} - \lambda_j} [e^{-\lambda_j(t_j - t_{j-1})} - e^{-k_{e0}(t_j - t_{j-1})}]$$

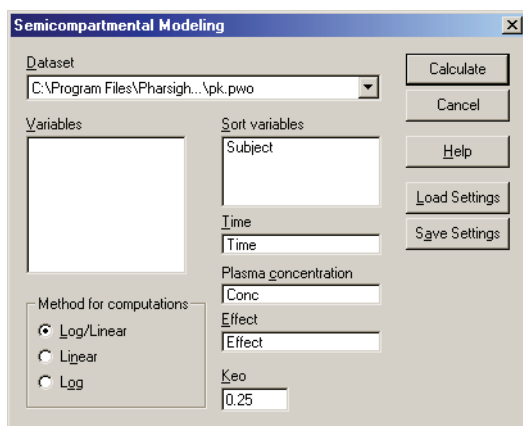
λ_1 and C_{e_1} are estimated by a linear regression, assuming $C_e(0) = C_p(0) = 0$.

WinNonlin provides three methods. The 'linear' method uses linear piecewise PK model; the 'log' method uses log-linear piecewise PK model; the default 'log/linear' method uses linear to T_{max} and then log-linear after T_{max} . The Effect field is not used for the calculation of C_e but when it is provided, the $E-C_p$ and $E-C_e$ plot will be plotted.

Performing semicompartmental modeling

To perform semicompartmental modeling:

1. Open a data set containing plasma concentrations and times, at minimum.
2. Choose **Tools>Semicompartmental modeling** from the WinNonlin menus. The Semicompartmental Modeling dialog appears. Make sure the appropriate data set appears under Dataset, and its columns appear under Variables.
3. *Sort variables*: If using sort variables to define individual data profiles, drag one or more columns to the sort variables field. A separate analysis is performed for each unique combination of sort variable values.
4. *Time*: Drag the column containing times to this field.
5. *Plasma Concentration*: Drag the column containing blood concentrations here.
6. *Effect*: Drag a column containing drug effect data to this field in order to include concentration-effect plots in the output. This column is not used in estimating effect site concentration.
7. *Keo*: Enter the value for the equilibration rate constant.

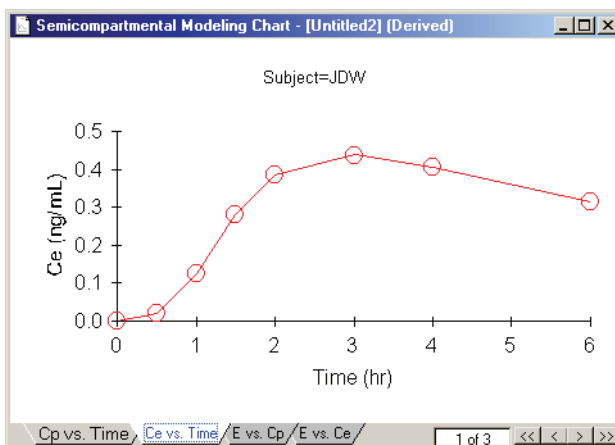
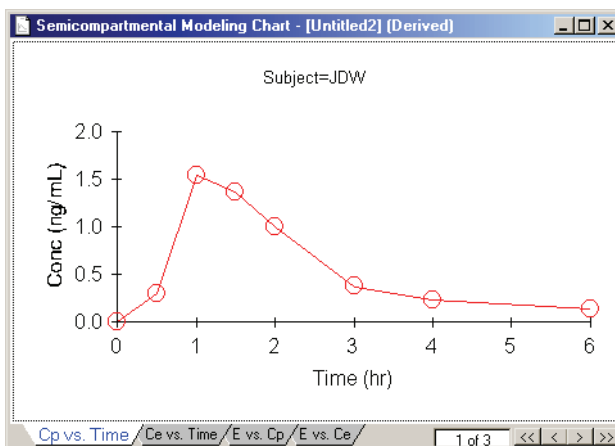


8. *Method for computations*: Select **Linear** to use a linear piecewise PK model; **Log** for a log-linear piecewise PK model; or the default **Log/linear** for linear to Tmax and then log-linear after Tmax.
9. Click **Calculate**. WinNonlin performs the computations and generates a new workbook and chart window.

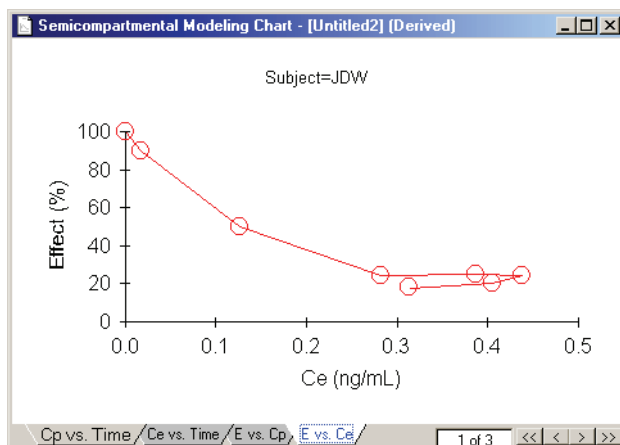
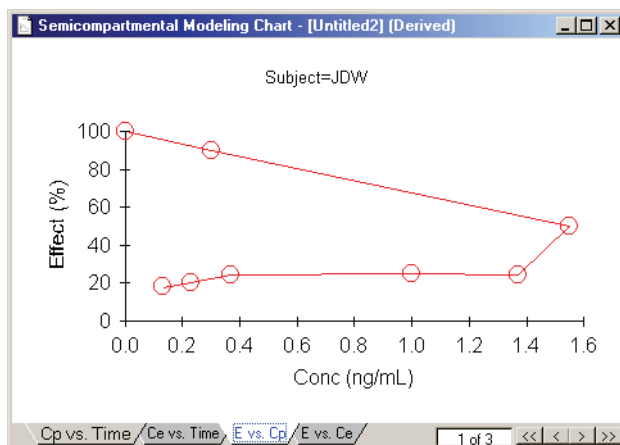
Output

Workbook: WinNonlin creates a new workbook with the following data: any sort variables, Time, Conc, C_e (estimated effect site concentration), and Effect (if included).

Chart: The chart window contains plots of C_p versus Time and C_e versus Time. If Effect is included in the data set, plots of Effect versus C_p and Effect versus C_e are also created. Examples are provided below.



Note: Any units associated with the Time and Concentration columns in the input data are carried through to the semicompartmental modeling output.



When the data set has multiple sort levels, a separate plot appears for each.

Crossover design

The two-period crossover design is common for clinical trials in which each subject serves as his or her own control. In many situations the assumptions of variance homogeneity and normality, relied upon in parametric analyses, may not be justified due to small sample size, among other reasons. The nonparametric methods proposed by Koch (1972) can be used to analyze such data.

WinNonlin's Crossover Design tool performs statistical analysis of data arising from subjects when variance homogeneity and normality may not necessarily apply. This tool performs nonparametric statistical tests on variables such as Tmax and provides two categories of results. First, it calculates confi-

dence intervals for each treatment median. It then calculates the median difference between treatments and the corresponding confidence intervals. Finally, it estimates the relevance of direct, residual, and period effects. See “Using the Crossover Design tool” on page 260 for stepped instructions.

Data and assumptions

Consider a trial testing the effects of two treatments, Test (T) and Reference (R). Suppose that n subjects are randomly assigned to the first sequence, TR, in which subjects are given treatment T first and then treatment R following an adequate washout period. Similarly, m subjects are assigned to the RT sequence. Y represents the outcome of the trial (for example Cmax or Tmax) and y_{ijk} is the value observed on sequence i ($i=1,2$), subject j ($j = 1,2,\dots,n, n+1,\dots,n+m$) and period k ($k = 1,2$). Treatment is implied by sequence i and period k , for example $i=1$ and $k=1$ is treatment T; $i=1$ and $k=2$ is treatment R. The data are listed in columns 1 through 4 of Table 12-1 below, and column 5 gives the within-subject difference in Y between the two periods.

Table 12-1. Representation of the data

Sequence (i)	Subject (j)	Y_{ij1}	Y_{ij2}	D_{ij}
TR	1	y_{111}	y_{112}	d_{11}
...	
TR	n	y_{1n1}	y_{1n2}	d_{1n}
RT	$n+1$	$y_{2(n+1)1}$	$y_{2(n+1)2}$	$d_{2(n+1)}$
...	
RT	$n+m$	$y_{2(n+m)1}$	$y_{2(n+m)2}$	$d_{2(n+m)}$

$$d_{ij} = y_{ij1} - y_{ij2}.$$

Then the $n*m$ crossover differences $(d_{1i} - d_{2j})/2$ are computed, along with the median of these $n*m$ points. This median is the Hodges-Lehmann estimate of the difference between the median for T and the median for R.

Crossover Design supports two data layouts: stacked in same column or separate columns. For “stacked” data, all measurements appear in a single column, with one or more additional columns flagging which data belong to which treatment. The data for one treatment must be listed first, then all the data for

the other. The alternative is to place data for each treatment in a separate column.

Descriptive analysis

The median response and its confidence interval are calculated for Y of each treatment and the crossover difference between treatments, C . Let $X_{(1)}, X_{(2)}, \dots, X_{(N)}$ be the order statistic of samples X_1, X_2, \dots, X_N . The median Xm is defined as:

$$Xm = [X_{(N/2)} + X_{(N/2+1)}]/2, \quad \text{if } N \text{ is even}$$

$$= X_{[(N+1)/2]}, \quad \text{if } N \text{ is odd}$$

The 100% confidence interval (CI) of Xm is defined as follows.

- For $N \leq 20$, the exact probability for a binomial distribution is used:

$$X_l = X_{(L)}, \text{ where } L = \max \left(i: \sum_{i=1}^N \binom{N}{i} 2^{-N} < (1-p)/2 \right) + 1$$

$$X_u = X_{(U)}, \text{ where } U = \min \left(i: \sum_{i=1}^N \binom{N}{i} 2^{-N} > (1+p)/2 \right) - 1$$

$$\text{The exact probability is } \sum_{i=L}^U \binom{N}{i} 2^{-N}.$$

- For $N > 20$, normal approximation is used:

$$X_l = X_{(L)}, \text{ where } L = \text{int} \left(\frac{N}{2} - z_{\frac{1+p}{2}} \frac{\sqrt{N}}{2} \right) + 1$$

$$X_u = X_{(U)}, \text{ where } U = N - \text{int} \left(\frac{N}{2} - z_{\frac{1+p}{2}} \frac{\sqrt{N}}{2} \right)$$

where $\text{int}(X)$ returns the largest integer less than or equal to X .

Hypothesis testing

Four hypotheses are of interest:

1. no sequence effect (or drug residual effect);
2. no treatment effect given no sequence effect;
3. no period effect given no sequence effect; and
4. no treatment and no sequence effect.

Hypothesis 1 above can be tested using the Wilcoxon statistic on the sum S . If $R(S_l)$ is the rank of S_{ij} in the whole sample, $l = 1, \dots, n+m$. The test statistic is:

$$T = \min \left(\sum_{l=1}^n R_l, \sum_{l=n+1}^{n+m} R_l \right)$$

The p-value is evaluated by using normal approximation (Conover, 1980):

$$[T - n(n+m+1)/2] / \sqrt{nm(n+m+1)/12} \sim N(0,1)$$

Similarly, hypothesis 2 can be tested using the Wilcoxon statistic on the difference D ; hypothesis 3 can be tested using the Wilcoxon statistic on the crossover difference C . The statistics are in the form described above.

Hypothesis 4 can be tested using the bivariate Wilcoxon statistic on (Y_{ij1}, Y_{ij2}) . For each period k , let R_{ijk} equal:

$$\text{rank} \{ y_{ijk}: y_{11k}, \dots, y_{1nk}, y_{2(n+1)k}, \dots, y_{2(n+m)k} \}$$

The average rank for each sequence is

$$\bar{R}_{ik} = \sum_{j=1}^{n_i} R_{ijk} / n_i, \text{ where } j=1, \dots, n_i, n_i=n \text{ for } i=1, \text{ and } n_i=m \text{ for } i=2.$$

Thus, the statistic to be used for testing hypothesis 4 is:

$$L = (n+m-1)[nU_1^T \mathbf{S}^{-1} U_1 + mU_2^T \mathbf{S}^{-1} U_2]$$

Where U_i is a 2x1 vector:

$$[\bar{R}_{i1} - M, \bar{R}_{i2} - M]^T, M = (n+m+1)/2$$

\mathbf{S} is the 2x2 covariance matrix:

$$\mathbf{S} = \Sigma_i \Sigma_j \begin{bmatrix} (R_{ij1} - M)^2 & (R_{ij1} - M)(R_{ij2} - M) \\ (R_{ij1} - M)(R_{ij2} - M) & (R_{ij2} - M)^2 \end{bmatrix}$$

and $L \sim \text{chi-square}(2)$, so the p-value can be evaluated.

Using the Crossover Design tool

Crossover design supports two data formats: data for both treatments stacked in one column, or each treatment placed in a separate column.

Note: See the *WinNonlin Examples Guide* for examples of crossover design.

To use crossover design:

1. Make sure that the data set to use is open and active.
2. Choose **Tools>Crossover Design** from the WinNonlin menus to open the Crossover Design dialog.
3. *Sort variables:* If using sort variables in addition to subject IDs, treatment and sequence to define individual data profiles, drag one or more columns to the sort variables field. Each unique combination of sort variable values for each subject and treatment represents one data profile.
4. *Subject:* Drag the variable representing subject identifiers from the Variables list box to the Subject field.
5. *Sequence:* Drag the variable representing treatment sequence to this field.
6. *Treatment data:* Select whether the treatment data are in separate columns or stacked in one column. An example of each appears below.

Separate columns:

The screenshot shows the 'Crossover Design' dialog box. The 'Dataset' field contains 'C:\Program Files\Pha...\separate.pwo'. The 'Variables' list is empty. The 'Sort variables' list is empty. The 'Treatment data' section has two radio buttons: 'Separate columns' (selected) and 'Stacked in same column'. The 'Subject' field contains 'Subject'. The 'Treatment A:' field contains 'Trit_G'. The 'Treatment B:' field contains 'Trit_H'. The 'Sequence' field contains 'Sequence'. The 'Calculate', 'Cancel', 'Help', 'Load Settings', and 'Save Settings' buttons are on the right.

7. If data for each treatment appears in a separate column, drag the column containing response data for each treatment to the Treatment A and Treatment B fields as shown above.

Stacked in one column:

The screenshot shows the 'Crossover Design' dialog box. The 'Dataset' field contains 'C:\Program Files\Pha...\stacked.pwo'. The 'Variables' list contains 'PERIOD'. The 'Sort variables' list contains 'PARAMETER'. The 'Treatment data' section has two radio buttons: 'Separate columns' and 'Stacked in same column' (selected). The 'Subject' field contains 'SUBJECT'. The 'Treatment:' field contains 'TREATMENT'. The 'Response:' field contains 'ESTIMATE'. The 'Sequence' field contains 'SEQUENCE'. The 'Calculate', 'Cancel', 'Help', 'Load Settings', and 'Save Settings' buttons are on the right.

8. If response data for both treatments are stacked in a single column:
 - a. *Treatment:* Drag the column the identifies each treatment to this field.
 - b. *Response:* Drag the column containing response data to this field.
9. Click **Calculate** to perform the analysis.

Output

Crossover Design creates a new workbook with two worksheets. The first sheet lists the confidence intervals (at 80%, 90%, and 95%) for the treatment medians in the crossover data and the treatment difference median.

	Variables	Median	Confidence_Level	Exact	CI_Lower	CI_Upper
1	Treatment_Diff_(trt_G - trt_H)	-2.175	0.80	0.8000	-4.1000	-0.9250
2	Treatment_Diff_(trt_G - trt_H)	-2.175	0.90	0.9000	-4.2000	-0.6250
3	Treatment_Diff_(trt_G - trt_H)	-2.175	0.95	0.9500	-4.8750	-0.3250
4	trt_G	0.35	0.80	0.8906	0.3000	1.5000
5	trt_G	0.35	0.90	0.8906	0.3000	1.5000
6	trt_G	0.35	0.95	0.9785	0.2500	1.5500
7	trt_H	1.85	0.80	0.8906	0.7500	7.1000
8	trt_H	1.85	0.90	0.8906	0.7500	7.1000
9	trt_H	1.85	0.95	0.9785	0.6300	7.2000

Note: Units associated with the Response variable in the input data set units are carried through to the crossover design output. The output workbooks display units in the median, lower, and upper value parameters.

The second worksheet, called Effects, lists the test statistic and p-value associated with each of four tests. It includes tests for an effect of sequence, treatment and period, as well as treatment and residual simultaneously.

	Test	T_stat	L_stat	p_Value
1	Sequence	22		0.2506
2	Treatment (SEQ1=SEQ2)	16		0.0163
3	Period (SEQ1=SEQ2)	33		0.2506
4	Treatment & Residual (simultaneous)		5.97	0.0505

Numeric deconvolution

Deconvolution evaluates in vivo drug release and delivery based on data for a known drug input. Deconvolution calculations and stepped usage instructions appear under the following headings.

- “Deconvolution methodology”

- [“Using deconvolution” on page 271](#)

Note: WinNonlin version 5.1 established several enhancements to numeric deconvolution. Scripts written prior to WinNonlin 5.1 should operate in later versions, but code to process deconvolution output may need modification to account for enhancements to the workbook output.

Deconvolution methodology

Deconvolution is used to evaluate *in vivo* drug release and delivery when data from a known drug input are available. Depending upon the type of reference input information available, the drug transport evaluated will be either a simple *in vivo* drug release (e.g., gastro-intestinal release) or a composite form, typically consisting of an *in vivo* release followed by a drug delivery to the general systemic circulation. See [“Numeric deconvolution” on page 262](#) for instructions on performing deconvolution.

Perhaps the most common application of deconvolution is in the evaluation of drug release and drug absorption from orally administered drug formulations. In this case, the bioavailability is evaluated if the reference input is a vascular drug input. Similarly, gastro-intestinal release is evaluated if the reference is an oral solution (oral bolus input).

The Deconvolution feature is not limited in scope to drug delivery via the oral route. It considers other types of delivery, including transdermal release and delivery, drug delivery from implanted devices, etc.

Deconvolution provides automatic calculation of the drug input. It also allows the user to direct the analysis to investigate issues of special interest through different parameter settings and program input.

Linearity assumptions

The methodology is based on linear system analysis with linearity defined in the general sense of the linear superposition principle. It is well recognized that classical linear compartmental kinetic models exhibit superposition linearity due to their origin in linear differential equations. However, all kinetic models defined in terms of linear combinations of linear mathematical operators (e.g. differentiation, integration, convolution, deconvolution, etc.) constitute linear systems adhering to the superposition principle. The scope and generality of the linear system approach thus ranges far beyond linear com-

partmental models. To fully appreciate the power and generality of the linear system approach, it is best to depart from the typical, physically structured modeling view of pharmacokinetics. To cut through all the complexity and objectively deal with the present problems, it is best to consider the kinetics of drug transport in a non-parametric manner and simply use stochastic transport principles in the analysis.

Convolution/deconvolution - stochastic background

The convolution/deconvolution principles applied to evaluate drug release and drug input (absorption) can be explained in terms of point A to point B stochastic transport principles. For example, consider the entry of a single drug molecule at point A at time 0. Let B be a sampling point (a given sampling space or volume) anywhere in the body (e.g. a blood sampling) that can be reached by the drug molecule entering at A. The possible presence of the molecule at B at time t is a random variable due to the stochastic transport principles involved. Imagine repeating this one molecule experiment an infinite number of times and observing the fraction of times that the molecule is at B at time t . That fraction represents the probability that a molecule is at point B given that it entered point A at time 0. Denote this probability by the function $g(t)$.

Next, consider a simultaneous entry of a very large number (N) of drug molecules at time 0, e.g., a bolus injection. Let it be assumed that there is no significant interaction between the drug molecules, such that the probability of a drug molecule's possible arrival at sampling site B is not significantly affected by any other drug molecule. Then the transport to sampling site B is both random and independent. In addition the probability of being at B at time t is the same and equals $g(t)$ for all of the molecules. It is this combination of independence and equal probability that leads to the superposition property that in its most simple form reveals itself in terms of dose-proportionality. For example, in the present case the expected number of drug molecules to be found at sampling site B at time t ($N_B(t)$) is equal to $Ng(t)$. It can be seen from this that the concentration of drug at the sampling site B at the arbitrary time t is proportional to the dose (N is proportional to the dose and $g(t)$ does not depend on the dose due to the independence in the transport).

Now let's further assume that the processes influencing transport from A to B are constant over time. The result is that the probability of being at point B at time t depends on the elapsed time since entry at A, but is otherwise independent of the entry time. Thus the probability of being at B for a molecule that enters A at time t_A is $g(t-t_A)$. This assumed property is called time-invariance. Consider again the simultaneous entry of N molecules, but now at time t_A

instead of 0. Combining this property with those of independent and equal probabilities, i.e., superposition, results in an expected number of molecules at B at time t given by $N_B(t) = Ng(t-t_A)$.

Suppose that the actual time at which the molecule enters point A is unknown. It is instead a random quantity t_A distributed according to some probability density function $h(t)$, i.e.,

$$Pr(t_A \geq t) = \int_0^t h(u) du$$

The probability that such a molecule is at point B at time t is the average or expected value of $g(t-t_A)$ where the average is taken over the possible values of t_A according to

$$\bar{g}(t) = \int_{-\infty}^{\infty} g(t-t_A)h(t_A)dt_A$$

Causality dictates that $g(t)$ must be 0 for any negative times, i.e., a molecule can't get to B before it enters A. For typical applications drug input is restricted to non-negative times so that $h(t) = 0$ for $t < 0$. As a result the quantity $g(t-t_A)h(t_A)$ is nonzero only over the range from 0 to t and the equation becomes

$$\bar{g}(t) = \int_0^t g(t-t_A)h(t_A)dt_A$$

Suppose again that N molecules enter at point A but this time they enter at random times distributed according to $h(t)$. This is equivalent to saying that the expected rate of entry at A is given by $N'_A(t) = Nh(t)$. As argued before, the expected number of molecules at B at time t is the product of N and the probability of being at B at time t , i.e.,

$$N_B(t) = N\bar{g}(t) = N \int_0^t g(t-t_A)h(t_A)dt_A = \int_0^t g(t-t_A)N'_A(t_A)dt_A$$

Converting from number of molecules to mass units:

$$M_B(t) = \int_0^t g(t-t_A)f(t_A)dt_A$$

where $M_B(t)$ is the mass of drug at point B at time t and $f(t)$ is the rate of entry in mass per unit time into point A at time t .

Let V_s denote the volume of the sampling space (point B). Then the concentration, $c(t)$, of drug at B is

$$c(t) = \int_0^t f(t-u)c_\delta(u)du \equiv f(t)*c_\delta(t)$$

where “*” is used to denote the convolution operation and

$$c_\delta(t) = g(t)/V_s$$

The above equation is the key convolution equation that forms the basis for the evaluation of the drug input rate, $f(t)$. The function $c_\delta(t)$ is denoted the unit impulse response (a.k.a. characteristic response or disposition function). The process of determining the input function $f(t)$ is called deconvolution because it is required to deconvolve the convolution integral in order to extract the input function that is embedded in the convolution integral.

The unit impulse response function (c_δ) provides the exact linkage between drug level response $c(t)$ and the input rate function $f(t)$. c_δ is simply equal to the probability that a molecule entering point A at $t = 0$ is present in the sampling space at time t divided by the volume of that sample space.

The function representation

WinNonlin models the input function as a piecewise linear “precursor” function $f_p(t)$ convolved with an exponential “dispersion” function $f_d(t)$. The former provides substantial flexibility whereas the latter provides smoothness. The piecewise linear component is parameterized in terms of a sum of hat-type wavelet basis functions, $h_j(t)$:

$$f_p(t) = \sum x_j h_j(t) \quad x_j \geq 0$$

$$h_j(t) = \frac{t - T_j}{T_{j+1} - T_j}, \quad T_j < t < T_{j+1}$$

$$h_j(t) = \frac{t - T_j}{T_{j+1} - T_j}, \quad T_{j+1} < t < T_{j+2}$$

$$h_j(t) = 0, \quad \text{otherwise}$$

where T_j are the wavelet support points. The hat-type wavelet representation enables discontinuous, finite duration drug releases to be considered together with other factors that result in discontinuous delivery, such as stomach emptying, absorption window, pH changes, etc. The dispersion function provides the smoothing of the input function that is expected from the stochastic transport principles governing the transport of the drug molecules from the site of release to subsequent entry to and mixing in the general systemic circulation.

The wavelet support points (T_j) are constrained to coincide with the observation times, with the exception of the very first support point that is used to define the lag-time, if any, for the input function. Furthermore, one support point is injected halfway between the lag-time and the first observation. This point is simply included to create enough capacity for drug input prior to the first sampling. Having just one support point prior to the first observation would limit this capacity. The extra support point is well suited to accommodate an initial “burst” release commonly encountered for many formulations.

The dispersion function $f_d(t)$ is defined as an exponential function:

$$f_d(t) = \frac{e^{-t/\delta}}{\delta}$$

where δ is denoted the dispersion or smoothing parameter. The dispersion function is normalized to have a total integral ($t = 0$ to ∞) equal one, which explains the scaling with the δ parameter. The input function, $f(t)$, is the convolution of the precursor function and the dispersion function:

$$f(t) = f_p(t) * f_d(t) \equiv \int_0^t f_p(t-u) f_d(u) du$$

The general convolution form of the input function above is consistent with stochastic as well as deterministic transport principles. The drug level profile, $c(t)$, resulting from the above input function is, according to the linear disposition assumption, given by:

$$c(t) = f_p(t) * f_d(t) * c_d(t)$$

where “*” is used to denote the convolution operation.

Deconvolution through convolution methodology

WinNonlin deconvolution uses the basic principle of deconvolution through convolution (DTC) to determine the input function. The DTC method is an iterative procedure consisting of three steps. First, the input function is adjusted by changing its parameter values. Second, the new input function is convolved with $c_\delta(t)$ to produce a calculated drug level response. Third, the agreement between the observed data and the calculated drug level data is quantitatively evaluated according to some objective function. The three steps are repeated until the objective function is optimized. DTC methods differ basically in the way the input function is specified and the way the objective function is defined. The objective function may be based solely on weighted or unweighted residual values (observed–calculated drug levels). The purely residual-based DTC methods ignore any behavior of the calculated drug level response between the observations.

The more modern DTC methods, including WinNonlin's approach, consider both the residuals and other properties such as smoothness of the total fitted curve in the definition of the objective function. The deconvolution method implemented in WinNonlin is novel in the way the regularization (smoothing) is implemented. Regularization methods in some other deconvolution methods are done through a penalty function approach that involves a measurement of the smoothness of the predicted drug level curve (e.g. integral of squared second derivative). The WinNonlin deconvolution method instead introduces the regularization directly into the input function through a convolution operation with the dispersion function, $f_d(t)$. In essence, a convolution operation acts like a “washout of details”, i.e. a smoothing due to the mixing operation inherent in the convolution operation. Consider, for example, the convolution operation that leads to the drug level response $c(t)$. Due to the stochastic transport principles involved, the drug level at time t is made up of a mixture of drug molecules that started their journey to the sampling site at different times and took different lengths of time to arrive there. Thus, the drug level response at time t depends on a mixture of prior input. It is exactly this mixing in the convolution operation that provides the smoothing. The convolution operation acts essentially as a low pass filter with respect to the filtering of the input information. The finer details (higher frequencies) are attenuated relative to the more slowly varying components (low frequency components).

Thus, the convolution of the precursor function with the dispersion function results in an input function, $f(t)$, that is smoother than the precursor function. WinNonlin allows the user to control the smoothing through the use of the smoothing parameter δ . Decreasing the value of the dispersion function parameter δ results in a decreasing degree of smoothing of the input function.

Similarly, larger values of δ provide more smoothing. As δ approaches 0, the dispersion function becomes equal to the so-called Dirac delta “function”, resulting in no change in the precursor function.

The smoothing of the input function, $f(t)$, provided by the dispersion function, $f_d(t)$, is carried forward to the drug level response in the subsequent convolution operation with the unit impulse response function (c_δ). Smoothing and fitting flexibility are inversely related. Too little smoothing (too small a δ value) will result in too much fitting flexibility that results in a “fitting to the error in the data.” In the most extreme case, the result is an exact fitting to the data. Conversely, too much smoothing (too large a δ value) results in too little flexibility so that the calculated response curve becomes too “stiff” or “stretched out” to follow the underlying true drug level response.

Cross validation principles

WinNonlin's deconvolution function determines the optimal smoothing without actually measuring the degree of smoothing. The degree of smoothing is not quantified but is controlled through the dispersion function to optimize the “consistency” between the data and the estimated drug level curve. Consistency is defined here according to the cross validation principles. Let r_j denote the differences between the predicted and observed concentration at the j -th observation time when that observation is excluded from the data set. The optimal cross validation principle applied is defined as the condition that leads to the minimal predicted residual sum of squares (PRESS) value, where PRESS is defined as:

$$\text{PRESS} = \sum_{j=1}^m r_j^2$$

For a given value of the smoothing parameter δ , PRESS is a quadratic function of the wavelet scaling parameters x . Thus, with the non-negativity constraint $x > 0$, the minimization of PRESS for a given δ value is a quadratic programming problem that has a unique solution that can be determined numerically. Let PRESS (δ) denote such a solution. The optimal smoothing is then determined by finding the value of the smoothing parameter δ that minimizes PRESS (δ). This is a one variable optimization problem with an embedded quadratic-programming problem.

User control over execution

WinNonlin's deconvolution function permits the user to override the automatic smoothing by manual setting of the smoothing parameter. The user may also specify that no smoothing should be performed. In this case the input rate function consists of the precursor function alone.

Besides controlling the degree of smoothing, the user may also influence the initial behavior of the estimated input time course. In particular the user may choose to constrain the initial input rate to 0 ($f(0) = 0$) and/or constrain the initial change in the input rate to 0 ($f'(0) = 0$). By default WinNonlin does not constrain either initial condition. Leaving $f(0)$ unconstrained permits better characterization of formulations with rapid initial “burst release,” e.g., extended release dosage forms with an immediate release shell. This is done by optionally introducing a bolus component (or “integral boundary condition”) for the precursor function so the input function becomes:

$$\begin{aligned} f(t) &= (f_p(t) + x_\delta f_d(t)) * f_d(t) \\ &= f_p(t) * f_d(t) + x_\delta f_d(t) \end{aligned}$$

where the $f_p(t) * f_d(t)$ is defined as before. The difference here is the superposition of the extra term $x_\delta f_d(t)$ that represents a particularly smooth component of the input. The magnitude of this component is determined by the scaling parameter x_δ , which is determined in the same way as the other wavelet scaling parameters previously described.

The estimation procedure is not constrained with respect to the relative magnitude of the two terms of the composite input function given above. Accordingly, the input can “collapse” to the “bolus component” $x_\delta f_d(t)$ and thus accommodate the simple first order input commonly experienced when dealing with drug solutions or rapid release dosage forms. A drug suspension in which a significant fraction of the drug may exist in solution should be well described by the composite input function option given above. The same may be the case for dual release formulations designed to rapidly release a portion of the drug initially and then release the remaining drug in a prolonged fashion. The prolonged release component will probably be more erratic and would be described better by the more flexible wavelet-based component $f_p(t) * f_d(t)$ of the above dual input function.

Constraining the initial rate of change to 0 ($f'(0) = 0$) introduces an initial lag in the increase of the input rate that is more continuous in behavior than the usual abrupt lag time. This constraint is obtained by constraining the initial value of the precursor function to 0 ($f_p(0) = 0$). When such a constrained pre-

cursor is convolved with the dispersion function, the resulting input function has the desired constraint.

The extent of drug input

The extent of drug input in the Deconvolution module is presented in two ways. First, the amount of drug input is calculated as:

$$\text{Amount input} = \int_0^t f(t) dt$$

Second, the extent of drug input is given in terms of fraction input. If test doses are not supplied, the fraction is defined in a non-extrapolated way as the fraction of drug input at time t relative to the amount input at the last sample time, t_{end} :

$$\text{Fraction input} = \frac{\int_0^t f(t) dt}{\int_0^{t_{\text{end}}} f(t) dt}$$

The above fraction input will by definition have a value of 1 at the last observation time, t_{end} . This value should not be confused with the fraction input relative to either the dose or the total amount absorbed from a reference dosage form. If dosing is entered, as detailed under [“Deconvolution dosing” on page 274](#), the fraction input is relative to the dose amount.

Using deconvolution

[Numeric deconvolution](#) is used to evaluate *in vivo* drug release and delivery when data from a known drug input are available. WinNonlin’s Deconvolution tool can estimate the cumulative amount and fraction absorbed over time for individual subjects, given PK profile data and dose per subject.

To use numeric deconvolution:

1. Obtain the unit impulse response function (UIR).

For example, fit time and concentration data in WinNonlin using PK model 1, 8, or 18 (for $N=1, 2$, or 3 , respectively), and adjust to reflect a single unit dose. The UIR must take the form:

$$c_{\delta}(t) = \sum_{j=1}^N A_j e^{-\alpha_j t}$$

Since the UIR assumed the response from 1 dose unit, for model 1 (one compartment), take V (volume) from the 1-compartment IV model output and let $A = 1/V$. Make units corrections if needed (see below). For model 8 (two compartment), take the A and B from the model 8 output and divide by the Strip-ping Dose to get A1 and A2 for the UIR. Again, perform the units conversion if needed. Use model 18 (three compartment) similarly model 8.

If the dose units differ from the concentration mass units in the deconvolution input dataset, take care in entering the A values: A values must use the units of the concentration data divided by the units of the dose used to determine the UIR. (See the example immediately below.) Alternately, use the Column Properties dialog to convert the concentration data to use the same mass units as the dose, or convert the dose amounts to use the concentration mass units.

For example, suppose the oral concentrations use ng/mL, and a 1 mg dose was used in determining the unit impulse response. In the one-compartment case, $A = 1/V$, so the units for 'A' in the model 1 output equal the inverse of the volume units. 'A' values must be converted to ng/mL/mg before they are used in the Deconvolution tool, i.e., $A = 10^6/V$.

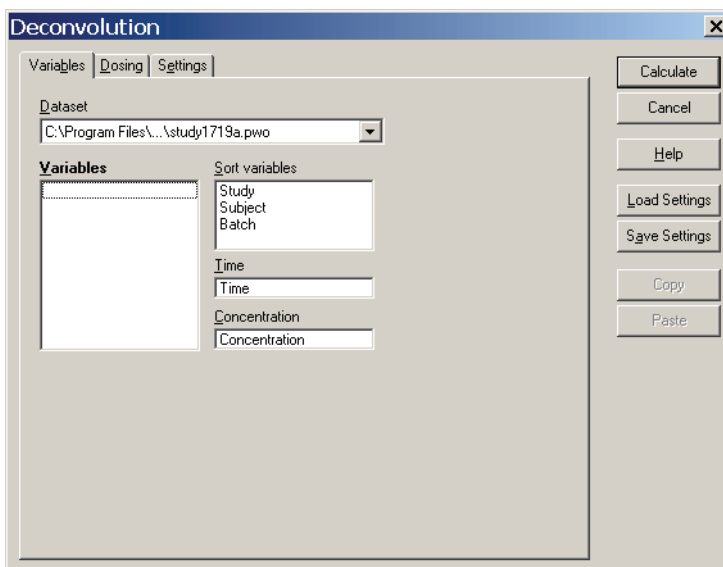
2. Once the UIR is established and units adjustments are complete, open a data set containing individual PK profiles in a WinNonlin workbook.
3. Choose **Tools>IVIVC>Deconvolution** (IVIVC license) or **Tools>Deconvolution** (no IVIVC license) from the WinNonlin menus.
4. Define the variable mappings, dosing and model settings as detailed under “Deconvolution variables” below, “Deconvolution dosing” on page 274 and “Deconvolution settings” on page 275.
5. Click **Calculate**. Deconvolution generates a new chart and workbook, discussed under “Deconvolution output” below.

Deconvolution variables

Numeric deconvolution requires a data set containing time-concentration data, with sort variables identifying individual profiles.

To select variables for numeric deconvolution:

1. Open the Variables tab of the Deconvolution dialog (**Tools>IVIVC>Deconvolution** (IVIVC license) or **Tools>Deconvolution** (no IVIVC license)).



2. *Dataset*: If needed, select the data set containing individual PK profiles.
3. *Sort variables*: If one or more variables contain values that differentiate individual data profiles, drag those columns to the Sort Variables field.
4. *Time*: Drag the variable representing time (or other independent variable) to this field.
5. *Concentration*: Drag the dependent variable to this field.
6. Use the **Save Settings** and/or **Load Settings** buttons to save or load all settings in the Deconvolution dialog via an external file.
7. Proceed to “**Deconvolution dosing**” below.

Deconvolution dosing

Numeric deconvolution supports input of test dose information directly to the Deconvolution dialog, so that the fraction input can be computed relative to the administered dose. Dose information is optional; without it, deconvolution will run correctly, with the fraction input approaching a final value of 1.

Dose time is assumed to be zero for all sort levels. If a dose value is entered for any sort level, values are required for all sort levels (all or none).

Deconvolution

Variables | **Dosing** | Settings

Dose units (Input & UIR)
ug

Dose amounts for time / concentration data (optional)

	Study	Subject	Batch	Dose
1	ST001	1	7	100
2	ST001	1	9	
3	ST001	2	7	
4	ST001	2	9	
5	ST001	3	7	
6	ST001	3	9	
7	ST002	4	7	
8	ST002	4	9	
9	ST002	5	7	
10	ST002	5	9	
11	ST002	6	7	
12	ST002	6	9	

Calculate
Cancel
Help
Load Settings
Save Settings
Copy
Paste

To define dosing for numeric deconvolution:

1. Open the Dosing tab of the Deconvolution dialog (**Tools>IVIVC>Deconvolution** (IVIVC license) or **Tools>Deconvolution** (no IVIVC license)).
2. **Dose units:** (Optional) Enter units for all doses here. The same units must apply to the UIR and the test (oral) doses entered in this dialog. Dose units are applied only if the Time and Concentration columns in the data set have units. See [Table 3-2](#) and [Table 3-3](#) on page 54 for accepted units abbreviations.
3. **Dose:** Enter the dose amount for each sort level in the input data.
4. Use the **Save Settings** and/or **Load Settings** buttons to save or load all settings in the Deconvolution dialog via an external file.
5. Proceed to “**Deconvolution settings**” below.

Deconvolution settings

Numeric deconvolution in WinNonlin assumes a unit impulse response function (UIR) of the form:

$$c_s(t) = \sum_{j=1}^N A_j e^{-\alpha_j t}$$

The analysis can use the observed times from the input data set, or *in vivo* times from an entirely separate workbook.

Deconvolution

Variables | Dosing | Settings

Number of exponential terms in the unit impulse response: 1

	Study	Subject	Batch	A (1/ug)	alpha (1/hr)
1	ST001	1	7		
2	ST001	1	9		
3	ST001	2	7		
4	ST001	2	9		
5	ST001	3	7		
6	ST001	3	9		

☒ Output from 0 to last time point
☐ Output from [] to []
 Number of output data points: 101
☐ Use observed times from input workbook
☐ Use times from a workbook column
 Workbook: []
 Time column: []

Smoothing: Automatic
 Smoothing parameter: []
☐ Initial rate is 0
☐ Initial change in rate is 0

Buttons: Calculate, Cancel, Help, Load Settings, Save Settings, Copy, Paste

To define model settings for numeric deconvolution:

1. Open the Settings tab of the Deconvolution dialog (**Tools>IVIVC>Deconvolution** (IVIVC license) or **Tools>Deconvolution** (no IVIVC license)).
2. *Number of exponential terms in the unit impulse response*: Select the number of exponential terms (N in the equation above, with $1 \leq N \leq 9$).
3. *A and alpha*: For each profile, enter (or copy/paste) values for each exponential term in the unit impulse response function, e.g., values obtained by fitting WinNonlin PK model 1, 8, or 18 (for N=1, 2, or 3, respectively).

CAUTION: The A and alpha values must use the units shown in the A or alpha column headers (if any), to match the units used in the input data and dosing. In particular, A values must match the units of the concentration data. This may require conversion of A values if dose and concentration units differ. See [“Using deconvolution” on page 271](#) for further discussion.

4. *Smoothing*: Select one of the following choices.

- **Automatic** allows WinNonlin to find the optimal value for the dispersion parameter *delta*.
- **None** disables smoothing that comes from the dispersion function. If selected, the input function is the piecewise linear precursor function.
- **User-Specified** allows manual entry of the value of the dispersion parameter *delta* as the Smoothing Parameter, where $\delta > 0$. Increasing *delta* increases the amount of smoothing.

5. *Initial Rate is Zero*: Check this option to constrain the estimated input rate to be zero at the initial time (lag time).

6. *Initial Change in Rate is Zero*: Check this option to constrain the derivative of the estimated input rate to be zero at the initial time (lag time) (requires initial rate of zero for automatic or user-specified smoothing).

7. *Output data points*: at the lower left area of the dialog, select the number and times for output data points by selecting from among the following options.

- a. **Output from 0 to last time point**: WinNonlin will generate output at even intervals from time 0 to the time of the final input data point for each profile, inclusive. Enter the total number of data points (≤ 1001) below.
- b. **Output from X to Y**: WinNonlin will generate output at even intervals from time X to time Y, inclusive. Enter the total number of data points (≤ 1001) below.
- c. **Use observed times from input workbook**: WinNonlin will generate output at each time point in the input data set for each profile.
- d. **Use times from a workbook column**: Select the workbook and time column. The worksheet containing the time values must be open in a WinNonlin workbook window, and can contain no more than 1001 times.

8. Click **Calculate**. Deconvolution generates a new chart and workbook, discussed under [“Deconvolution output”](#) below.

Deconvolution output

WinNonlin numeric deconvolution generated workbook and chart output. The workbook output includes one workbook with the following six worksheets.

Table 12-2. Deconvolution workbook output

Worksheet	Content
Values	Time, input rate, cumulative amount (Cumul_Amt, using the dose units) and fraction input (Cumul_Amt / test dose or, if no test doses are given, then fraction input approaches 1) for each profile
Parameters	The smoothing parameter <i>delta</i> and absorption lag time for each profile
Exponential Terms	Values for the UIR exponential terms for each profile
Fitted Values	Predicted data for each profile
Settings	Input settings for smoothing, number of output points and start and end times for output
History	History of operations on the workbook. See “History worksheet” on page 37 .
Dosing	If user-specified test doses were entered in the Deconvolution dosing dialog, a Dosing worksheet is included in the output, listing the doses and dose units.

Deconvolution generates one chart window with three charts for each profile:

- Fitted Curve: observed time-concentration data vs. the predicted curve
- Input Rate: rate of drug input vs. time
- Cumulative Input: cumulative drug input vs. time
- Fa-Avg: Fabs (subject * formulation) grouped by subject with Fabs_avg (formulation) overlaid, vs. time, and sorted by formulation
- Fa-Avg by profile: Fabs (subject * formulation) grouped by formulation, vs time, and sorted by subject

The IVIVC Toolkit

In vivo-in vitro correlation and validation tools for formulation development

Users who have a WinNonlin *In Vivo-In Vitro* Correlation IVIVC Toolkit License can access additional tools for evaluating correlations between *in vitro* drug properties, such as dissolution, and *in vivo* responses, such as fraction absorbed, in order to predict PK responses such as maximum concentration or area under the curve. WinNonlin IVIVC features are detailed under the following headings:

- “[Numeric deconvolution](#)” on page 262: Users with or without an IVIVC Toolkit License can use deconvolution to evaluate drug absorption from time-concentration data, based on the unit impulse response.
- “[Wagner-Nelson and Loo-Riegelman methods](#)” on page 280: Estimate drug absorption assuming a one-compartment (Wagner-Nelson) or two-compartment (Loo-Riegelman) PK model.
- “[Convolution](#)” on page 295: Estimate or verify PK profiles based on the unit impulse response function and expected absorption profile.
- “[Levy plots](#)” on page 302: Comparing predicted and observed concentrations over time as a measure of correlation model fit.
- “[Sigmoidal/dissolution models](#)” on page 551: Sigmoidal models, including Hill, Weibull, Double Weibull and Makoid-Banakar, for modeling dissolution.
- “[The IVIVC Wizard](#)” on page 304: Structured environment for building, executing, saving and reloading common IVIVC work flows.

See the *WinNonlin Examples Guide* for examples demonstrating common IVIVC inquiries.

Wagner-Nelson and Loo-Riegelman methods

Users with a WinNonlin IVIVC Toolkit License can access Wagner-Nelson and Loo-Riegelman methods to estimate fraction of drug absorbed.

Wagner-Nelson inputs and calculations

The Wagner-Nelson method estimates the fraction of drug absorbed over time, relative to the total amount to be absorbed, following the method described in [Gibaldi and Perrier \(1975\)](#) pages 130 to 133. It uses as a basis AUC values computed for each time point in your time-concentration data.

Note: WinNonlin Wagner-Nelson computations assume single-dose PK data with a concentration value of 0 at dose time. If no concentration value exists at dose time, WinNonlin will insert a concentration value of 0.

You can choose to calculate AUC using either the linear, linear/log, or linear up/log down trapezoidal rule. (See “[Calculation methods for AUC](#)” on page 179 and “[AUC calculation and interpolation formulas](#)” on page 185 for definitions and calculation methods.) Note that the two linear non-compartmental analysis (NCA) methods are equivalent, as no interpolation is used in the Wagner-Nelson method.)

Similarly, you choose the method for computing Lambda Z: best fit, user-specified range, or user-specified value. If you specify a Lambda Z value, you may also enter an intercept or have WinNonlin compute it using the last positive concentration and associated time value:

$$intercept = \text{Lambda Z} * t_{last} + \ln(C_{last})$$

If you do not enter a value for AUCINF, it will be computed as for WinNonlin NCA:

$$AUCINF = AUC_{last} + C_{last} / \text{Lambda Z}$$

You choose whether to use the observed or predicted value for C_{last} , where $C_{last_pred} = \exp(intercept - \text{Lambda Z}(t_{last}))$.

The cumulative amount absorbed at time t , normalized by the central compartment volume VI , is computed as:

$$\text{Cumul_Amt_Abs_V}(t) = C(t) + \text{Lambda Z} * AUC(t)$$

Therefore, Cumul_Amt_Abs_V at $t = \text{infinity}$ is:

$$\text{Cumul_Amt_Abs_V}(\text{inf}) = \text{Lambda Z} * \text{AUCINF}$$

The relative fraction absorbed at time t is then:

$$\text{Rel_Fraction_Abs}(t) = \text{Cumul_Amt_Abs_V}(t) / \text{Cumul_Amt_Abs_V}(\text{inf})$$

The percent remaining at time t is computed as:

$$\text{Percent remaining}(t) = 100 * (1 - \text{Rel_Fraction_Abs}(t))$$

Data points with a missing value for either time or the concentration will be excluded from the analysis and will not appear in the output.

Loo-Riegelman inputs and calculations

Like Wagner-Nelson, the Loo-Riegelman method estimates the fraction of drug absorbed as a function of time, relative to the total amount to be absorbed, following the method described in [Gibaldi and Perrier \(1975\)](#) pages 136 to 142. It computes AUC values for each time point in your time-concentration data set, as in Wagner-Nelson. For Loo-Riegelman, you must provide estimates for K10, K12, and K21, as may be obtained by running WinNonlin PK model 7 on separate IV data.

Note: WinNonlin Loo-Riegelman computations assume single-dose PK data with a concentration value of 0 at dose time. If no concentration value exists at dose time, WinNonlin will insert a concentration value of 0.

You may supply a value for AUCINF or allow WinNonlin to compute it as for non-compartmental analysis:

$$\text{AUCINF} = \text{AUC}_{\text{last}} + C_{\text{last}} / \text{Lambda Z}$$

You can choose to use the observed or predicted value for C_{last} , where $C_{\text{last_pred}} = \exp(\text{intercept} - \text{Lambda Z}(t_{\text{last}}))$. As in the Wagner-Nelson method, you choose the method for computing Lambda Z: best fit, user-specified range, or user-specified value. You may specify an intercept for the last case. If the intercept is not specified, WinNonlin will compute it using the last positive concentration and associated time value:

$$\text{intercept} = \text{Lambda Z} * t_{\text{last}} + \ln(C_{\text{last}})$$

Note that the Loo-Riegelman method uses Lambda Z only to compute AUCINF; it uses the intercept only to compute C_{last_pred} .

The cumulative amount absorbed at time t , normalized by the central compartment volume VI , is:

$$\text{Cumul_Amt_Abs_V}(t) = \text{Conc}(t) + K_{10} * \text{AUC}(t) + \text{Amt_Peripheral_V}(t)$$

The amount in the peripheral compartment at time t , normalized by VI is computed sequentially as:

$$\begin{aligned} \text{Amt_Peripheral_V}(t) = & (\text{Prior_Amt_Peripheral_V}) * \exp(-K_{21} * \delta t) \\ & + (K_{12} * \text{Prior_Conc} / K_{21}) * (1 - \exp(-K_{21} * \delta t)) \\ & + (K_{12} * \delta t * \delta \text{Conc} / 2) \end{aligned}$$

See [Gibaldi and Perrier \(1975\)](#) pages 138-139 for the derivation of this equation.

Cumul_Amt_Abs_V at time infinity is:

$$\text{Cumul_Amt_Abs_V}(\text{inf}) = K_{10} * \text{AUCINF}$$

The fraction absorbed at time t is computed as:

$$\text{Rel_Fraction_Abs}(t) = \text{Cumul_Amt_Abs_V}(t) / \text{Cumul_Amt_Abs_V}(\text{inf})$$

The percent remaining at time t is computed as:

$$\text{Percent remaining}(t) = 100 * (1 - \text{Rel_Fraction_Abs}(t))$$

Data points with a missing value for either time or the concentration will be excluded from the analysis and will not appear in the output.

Using the Wagner-Nelson or Loo-Riegelman method

To use the Wagner-Nelson or Loo-Riegelman method:

1. Open the data set containing individual PK profiles in a WinNonlin workbook.
2. For deconvolution using a one-compartment PK model: choose **Tools>IVIVC Tools>Wagner-Nelson** from the WinNonlin menus.

3. For deconvolution using a two-compartment PK model: choose **Tools>IVIVC Tools>Loo-Riegelman** from the WinNonlin menus.

A model (text) window will appear, representing the newly-loaded model.

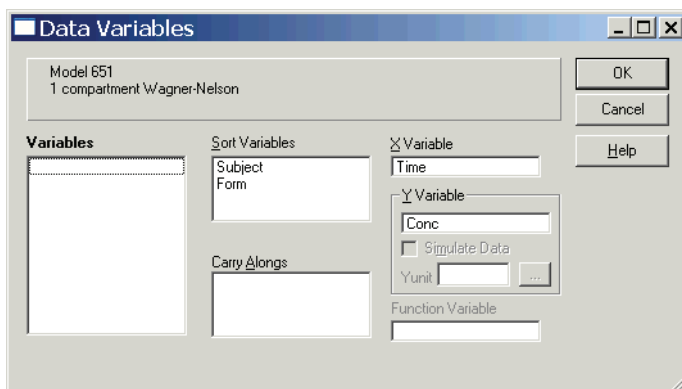
4. Define the variable mappings, model properties and options as detailed under the following headings:
 - “Data variables” on page 283
 - “Dosing regimen” on page 284
 - “Loo-Riegelman model parameters” on page 285 (Loo-Riegelman deconvolution only)
 - “Lambda Z ranges” on page 286
 - “Output options” on page 289
 - “Weighting” on page 290
 - “Title” on page 292
 - “Settings” on page 293
5. Run the analysis by choosing **Model>Start Modeling** from the WinNonlin menus.
6. The model and settings may be saved, loaded and refreshed as for any WinNonlin model, using the options on the **File** menu.

Data variables

To define the data variables to use for the Wagner-Nelson or Loo-Riegelman method:

1. Load the data and model as detailed under “Using the Wagner-Nelson or Loo-Riegelman method” on page 282.
2. Choose **Model>Data Variables** from the WinNonlin menus.
3. *Sort Variables*: Drag the variables needed to identify individual profiles to the Sort Variables list.
4. *X Variable*: Drag the independent variable (generally, time) to this field.
5. *Y Variable*: Drag the dependent variable (concentration) to this field.

6. *Carry Alongs*: Drag variables to be included in the output workbook (but not used in calculations) to this list.



7. Click **OK** and proceed to set the **Dosing regimen**.

Dosing regimen

If the data were loaded from PKS, the dosing should also have been loaded. In that case, skip to “[Loo-Riegelman model parameters](#)” on page 285 for Loo-Riegelman or “[Lambda Z ranges](#)” on page 286 for Wagner-Nelson.

To define the dosing:

1. Load the data and model as detailed under “[Using the Wagner-Nelson or Loo-Riegelman method](#)” on page 282, and set the data variables (**Model>Data Variables**).
2. Choose **Model>Dosing Regimen** from the WinNonlin menus.
3. *Value*: For each profile, enter the dose time, or use the **Load** button to load previously-saved dosing information from a file.
4. Use the **Save** button to save the dosing information for later re-use.

Note: If no concentration value exists at dose time for a given profile, WinNonlin will insert a concentration value of 0 at the specified dose time.

Model Properties

Model 651
1 compartment Wagner-Nelson

Dosing Regimen | Lambda z Ranges

	Subject	Form	Constant	Value
1	1	Capsule	Time of Last Dose	0
2	1	Tablet	Time of Last Dose	0
3	2	Capsule	Time of Last Dose	0
4	2	Tablet	Time of Last Dose	0
5	3	Capsule	Time of Last Dose	0
6	3	Tablet	Time of Last Dose	0
7	4	Capsule	Time of Last Dose	0
8	4	Tablet	Time of Last Dose	0
9	5	Capsule	Time of Last Dose	0
10	5	Tablet	Time of Last Dose	0
11	6	Capsule	Time of Last Dose	0
12	6	Tablet	Time of Last Dose	0

Buttons: OK, Cancel, Apply, Help, Copy, Paste, Clear All, Load..., Save...

5. Click **OK**.
6. For Loo-Riegelman models, proceed to set the [Loo-Riegelman model parameters](#). For Wagner-Nelson, skip to the [Lambda Z ranges](#).

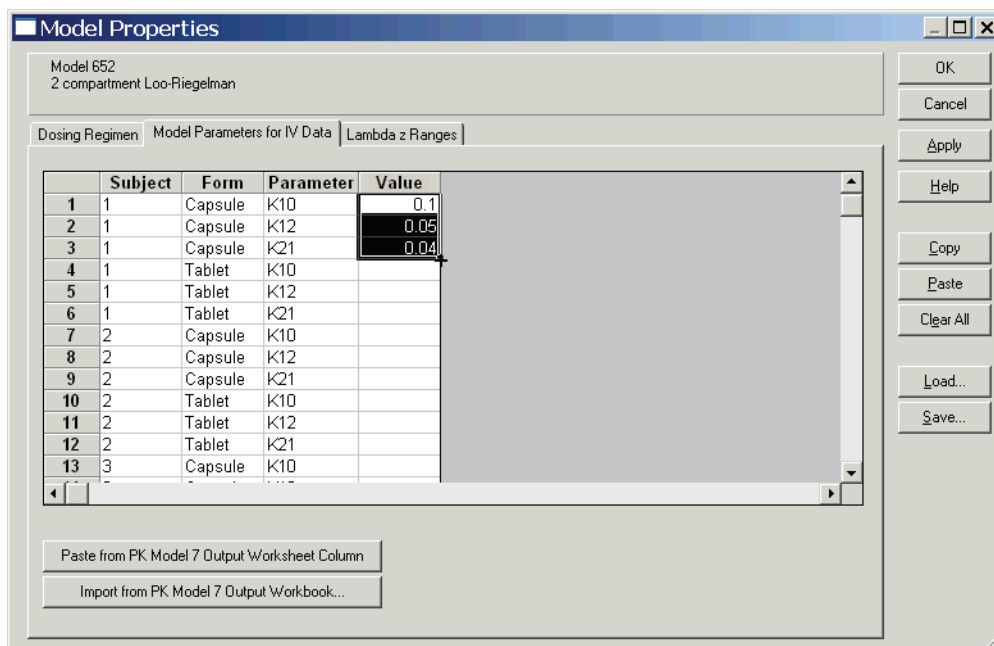
Loo-Riegelman model parameters

The Loo-Riegelman method requires initial estimates for the model parameters describing the two-compartment model that fits the IV data. Parameter estimates may be obtained by running WinNonlin PK model 7 on the data prior to using the Loo-Riegelman tool.

To set up initial model parameters for the Loo-Riegelman method:

1. Load the data and model as detailed under “Using the Wagner-Nelson or Loo-Riegelman method” on page 282, and set the data variables (**Model>Data Variables**).
2. Choose **Model>Model Parameters** from the WinNonlin menus.
3. *Value*: For each profile, enter initial parameter estimates for each parameter. Alternately, use the buttons at the bottom of the dialog to load the parameter

values from a PK output workbook obtained by running WinNonlin PK model 7 on the data prior to this analysis.

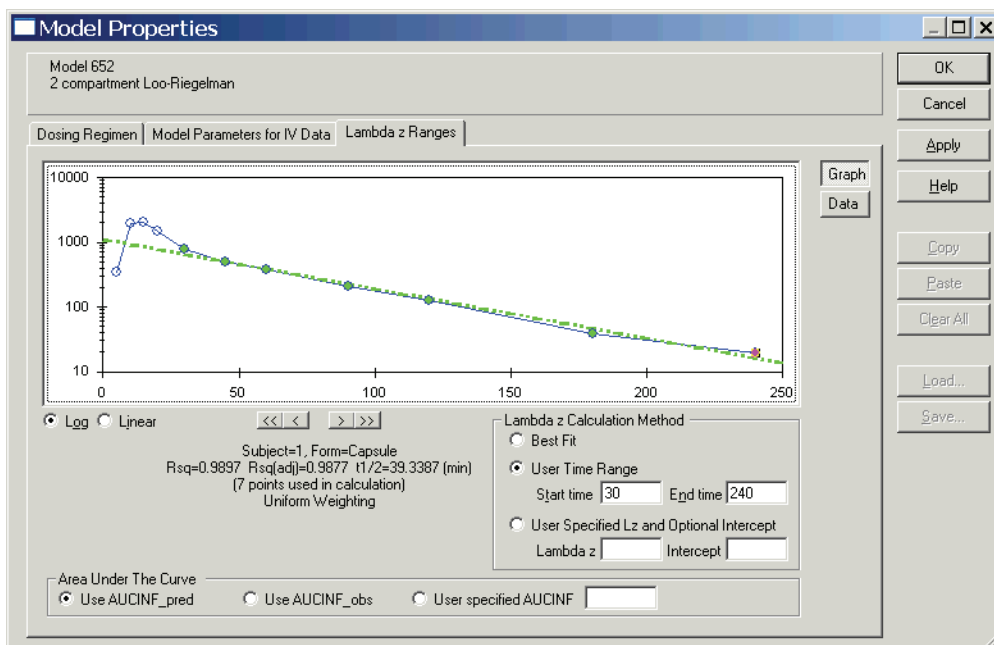


4. Click **OK** and proceed to set the **Lambda Z** ranges.

Lambda Z ranges

To set Lambda Z ranges for the Wagner-Nelson or Loo-Riegelman method:

1. Load the data and model as detailed under “Using the Wagner-Nelson or Loo-Riegelman method” on page 282, and set the data variables (**Model>Data Variables**).
2. Choose **Model>Lambda z Ranges** from the WinNonlin menus.
3. Choose whether to enter Lambda Z information using a plot of each profile or by entering tabular data, using the **Graph** and **Data** buttons at the top right.
4. For each profile, select Lambda Z and AUC settings as follows. Use the arrow buttons directly below the plot to move between profiles in the Plot view.



5. *Area Under the Curve*: Select one of the following options for calculation of area under the curve to time = infinity:

Use AUCINF_pred: WinNonlin will calculate the area under the curve as:
$$\text{AUCINF} = \text{AUC}_{\text{last}} + C_{\text{last}} / \text{Lambda } Z$$
, where C_{last} is the last predicted concentration.

Use AUCINF_obs: WinNonlin will calculate the area under the curve as:
$$\text{AUCINF} = \text{AUC}_{\text{last}} + C_{\text{last}} / \text{Lambda } Z$$
, where C_{last} is the last observed concentration.

User specified AUCINF: Enter a value for the area under the curve. Note that for the Loo-Riegelman method, the Lambda Z options apply only for the AUCINF computation. If AUCINF is a user-specified value, then the other Lambda Z settings are unnecessary for the Loo-Riegelman method.

Note: If observed data does not extend significantly into the terminal phase (i.e. absorption phase must be completed) then use of calculated **AUCINF** may significantly underestimate the actual **AUCINF**, resulting in scaling errors in the computation of fraction absorbed.

6. *Lambda Z Calculation Method*: Select one of the following options:

Best fit: If this option is selected in the Plot view, or Start Time, End Time and Lambda Z are empty in the Data view, WinNonlin will determine the Lambda Z range as detailed under “[Lambda Z or slope range selection](#)” on page 164.

User Time Range: Define the terminal elimination phase as follows:

- a. If using the Plot view, select the axis scale for the graphical representation below the lower left corner of the plot: **Linear** or **Log Linear**.
- b. Select the first data point to be included: Left-click on a data point or enter a time value under Start Time in the Data view.
- c. Select the last data point to be included: Hold down the **Shift** key while left-clicking on a data point or enter a time value under End Time.
- d. Exclusions (optional): To exclude a data point from Lambda Z calculations, hold down the **Ctrl** key and left-click on that point in the Plot view or enter the time value under Exclusions (or in the next column to the right, for multiple exclusions) for that profile in the Data view. Repeat to reverse an exclusion. These exclusions apply only to Lambda Z calculations; the excluded data points will be included in computation of AUC's.

User Specified Lz and Optional Intercept: (Available for AUC calculated as AUC_pred only.) Enter a value for Lambda Z and, optionally, for the intercept. If no intercept value is entered, WinNonlin will compute it as: $intercept = \text{Lambda Z} * t_{last} + \ln(C_{last})$.

Since no specific points will be used in the computation of Lambda Z, the Lambda Z table in the output will give Predicted and Residual values for all Time and Conc values.

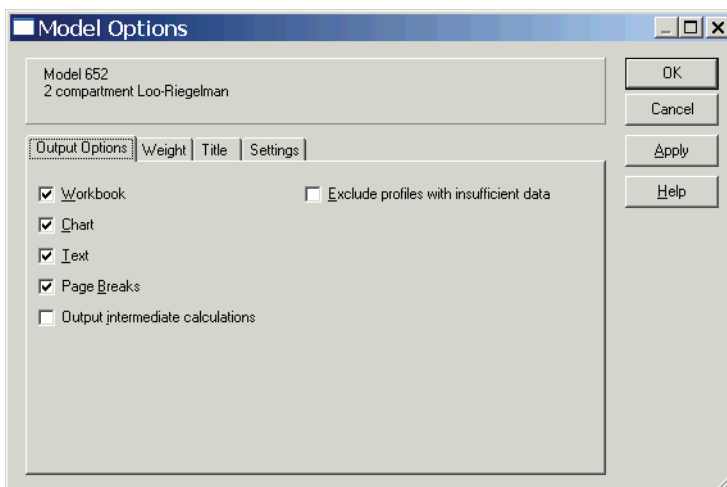
Note: Any AUC, Lambda_z, Intercept and K values entered must be greater than 0.

7. When done, click **Apply** to save the settings and move to another tab of the Model Properties dialog. To save the settings and close the dialog, click **OK**.
8. Proceed to set the [Output options](#).

Output options

To define the output options for the Wagner-Nelson or Loo-Riegelman method:

1. Load the data and model as detailed under “Using the Wagner-Nelson or Loo-Riegelman method” on page 282, and set the data variables (**Model>Data Variables**).
2. Choose **Model>Model Options** from the WinNonlin menus and open the **Output Options** tab.



3. Check the options to apply from among the following choices. After the analysis is run, one or more new windows is generated, with content as selected in the following options.

Note: Set default output options via **Tools>Options** in the WinNonlin menus.

Workbook: Check this option to generate an output workbook. The workbook contains the worksheets summarized in Table 13-1. These worksheets present the same information as the text output, but in tabular form.

Table 13-1. Summary of output worksheets

Sheet name	Contents
Wagner-Nelson	Wagner-Nelson method only. Profile estimates for each level of the sort variable, including time, concentration, cumulative AUC and cumulative amount and relative fraction absorbed. Units are those of the input data.
Loo-Riegelman	Loo-Riegelman method only. Profile estimates for each level of the sort variable, including time, concentration and cumulative AUC, amount in the peripheral compartment, amount absorbed, and relative fraction absorbed. Units are those of the input data.
Lambda Z Fit	Details for fitting Lambda Z. Not included for Loo-Riegelman with user-specified AUCINF.
User Settings	User defined settings
History	History of actions done in and to the workbook. “History worksheet” on page 37.

Chart: If workbook output is selected, check this item to generate a Chart window with two plots per profile: a linear plot of Relative_Fraction_Absorbed vs. Time, and a plot of the Lambda Z fit (excluded for Loo-Riegelman with user-specified AUCINF). If Lambda Z is user-specified, the latter plot is excluded.

Text: This option provides an ASCII text version of the analysis results, containing the same information as the workbook output. Modeling errors are also reported in this file.

Page Breaks: Check this box to include page breaks in the ASCII text output.

4. Click **OK** and proceed to set the [Weighting](#).

Weighting

The weighting options for the Loo-Riegelman and Wagner-Nelson methods apply to the calculation of Lambda Z (see [“Lambda Z ranges” on page 286](#)).

To define the weighting scheme to use in fitting Lambda Z:

1. Load the data and model as detailed under [“Using the Wagner-Nelson or Loo-Riegelman method” on page 282](#), and set the data variables (**Model>Data Variables**).

2. Choose **Model>Model Options** from the WinNonlin menus and open the **Weight** tab.
3. Select the desired weighting scheme. Common schemes are detailed below.

Note that it is not the weights themselves that are important; rather it is the relative proportions of the weights. See “[Weighting](#)” on page 243 for discussion of weighting schemes.

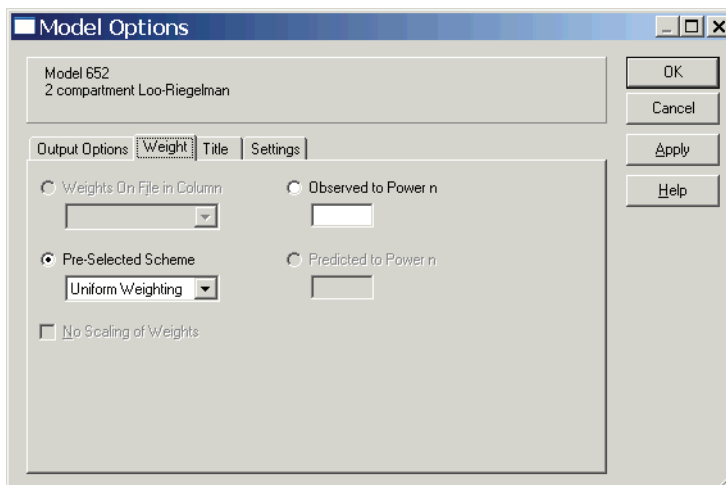
CAUTION: When a log-linear fit is done, it already uses weighting implicitly, approximately equal to $1/\hat{Y}^2$.

To use uniform weighting:

1. Select the **Pre-selected Scheme** radio button.
2. Select **Uniform Weighting** from the pull-down list.

To use weighted least squares:

1. Select **Observed to the Power n** button and enter the value of n , or use the **Pre-selected scheme** button to select the most common weighted least square options: $1/Y$ and $1/Y^2$.
2. When weighting by $1/Y$ and using the Linear/Log trapezoidal rule, one could argue that the weights should be $1/\text{Log}Y$, rather than $1/Y$. However, if this were the case, concentrations between 0 and 1 would have negative weights, and therefore could not be included.



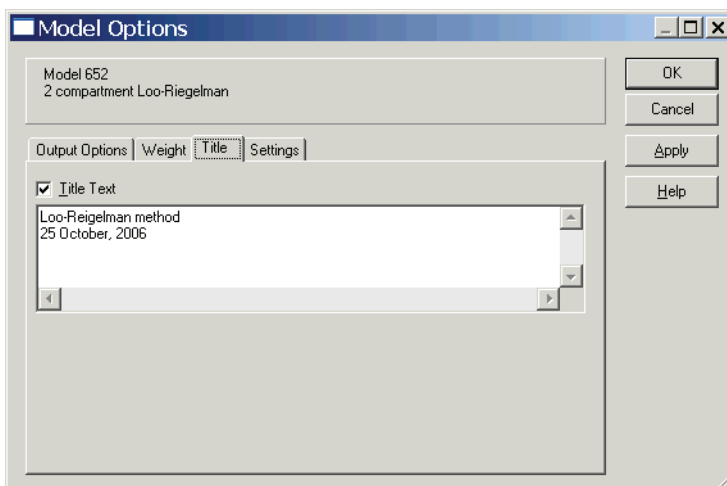
3. Click **OK** and proceed to set the **Title**.

Title

If included, the title is displayed at the top of each printed page in ASCII text output, on the User Settings worksheet in the workbook output, and at the top of each chart in chart output. It may include up to 5 lines of text.

To define and include the title:

1. Load the data and model as detailed under “Using the Wagner-Nelson or Loo-Riegelman method” on page 282, and set the data variables (**Model>Data Variables**).
2. Choose **Model>Model Options** from the WinNonlin menus and open the **Title** tab.
3. Make sure that **Title Text** is checked. If it is not, the title will not display even if text is entered below.
4. Enter the title text below. To include a line break use **Shift + Enter**. The title will be centered on the page.

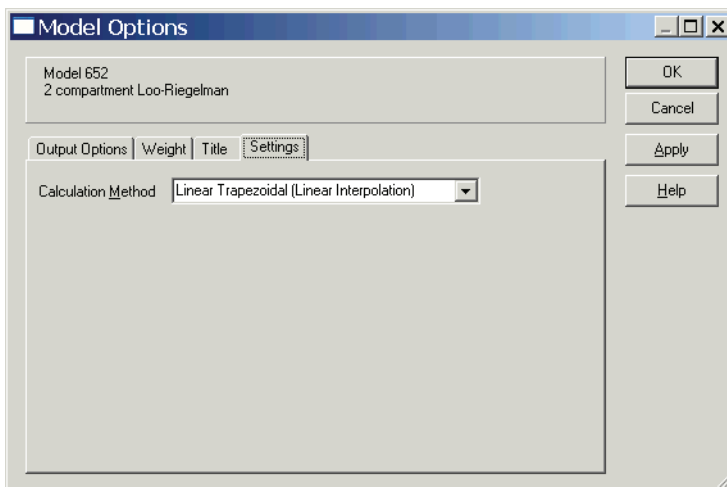


5. Click **OK** and proceed to the IVIVC [Settings](#) tab.

Settings

To set the AUC calculation method for Wagner-Nelson or Loo-Riegelman:

1. Load the data and model as detailed under “Using the Wagner-Nelson or Loo-Riegelman method” on page 282, and set the data variables (**Model>Data Variables**).



2. Choose **Model>Model Options** from the WinNonlin menus and open the **Settings** tab.

3. *Calculation Method*: Select the method for computing area under the curve.

Four methods are available. The method chosen applies to all AUCINF computations. All methods reduce to the log trapezoidal rule and/or linear trapezoidal rule, but differ with regard to when these rules are applied.

- Linear/Log Trapezoidal Method (method 1). Uses the log trapezoidal rule (given under “[AUC calculation and interpolation formulas](#)” on page 185) after Cmax, or after C0 for bolus (if C0 > Cmax), and linear trapezoidal otherwise. If Cmax is not unique, then the first maximum is used.
- Linear Trapezoidal rule (Linear Interpolation). This method (method 2) uses the linear trapezoidal rule, given under “[AUC calculation and interpolation formulas](#)” on page 185, applied to each pair of consecutive points in the data set that have non-missing values, and sums up these areas.
- Linear Up/Log Down Method (method 3). The linear trapezoidal rule is used any time that the concentration data is increasing, and the logarithmic trapezoidal rule is used any time that the concentration data is decreasing.
- Linear Trapezoidal Method with Linear/Log Interpolation (method 4). Equivalent to Linear Trapezoidal rule, since interpolation is not used in Loo-Riegelman and Wagner-Nelson methods.

Note: Methods 1 and 4 are the same for Wagner-Nelson and Loo-Riegelman, since no interpolation is done.

Note: The Linear/Log Trapezoidal, the Linear Up/Log Down, and the Linear Trapezoidal (with Linear/Log Interpolation) methods all apply the same exceptions in area calculation and interpolation: If a Y value (concentration, rate or effect) is less than or equal to zero, the program defaults to the linear trapezoidal or interpolation rule for that point. Similarly, if adjacent Y values are equal to each other, the program defaults to the linear trapezoidal or interpolation rule.

4. Click **OK**.

Convolution

The WinNonlin Convolution tool supports analytic evaluation of the convolution integral,

$$F(t) \cdot g(t) = \int_0^t f(t-u)g(u)du$$

where $f(t)$ and $g(t)$ are constrained to be either polyexponentials or piecewise polynomials (splines), in any combination. Polyexponentials must take the form:

$$f(t) = \sum_{i=1}^N A_i e^{-(\alpha_i \cdot t)}$$

where $N \leq 9$ and $t \geq 0$. To specify a polyexponential, you must supply the polyexponential coefficients, i.e., the A's and alpha's.

Splines must be piecewise constant or linear splines, i.e., they must take the form:

$$f(t) = \sum_{i=1}^N b_{ij}(t-t_j)^{i-1}, t \in (t_j, t_j + 1), j = 1, 2, \dots, m$$

$$= 0, \text{ otherwise}$$

where N is restricted to be 1 or 2.

For a spline function, you must supply a dataset to be fit with the splines. You can choose to convolve either this spline function, e.g., for input rate data, or the derivative of this spline function, e.g., for cumulative input data (so the derivative is input rate data). For the first case, the Convolution tool will find the spline with the requested degree of polynomial splines, and will use these polynomial splines in the convolution. For the derivative of a spline, the Convolution tool will fit the data with the requested degree of polynomial splines, differentiate the splines to obtain polynomial splines of one degree lower, and use the derivatives in the convolution. For example, if cumulative input data is fit with linear splines, then the derivatives are piecewise constant, so the piecewise constant function is convolved with the user's other specified function.

Linear splines are the lines that connect the (x,y) points. For $t \in (t_i, t_{i+1})$,

$$y = y_i + m_i (t - t_i),$$

where m_i is the slope $(y_{i+1} - y_i) / (t_{i+1} - t_i)$.

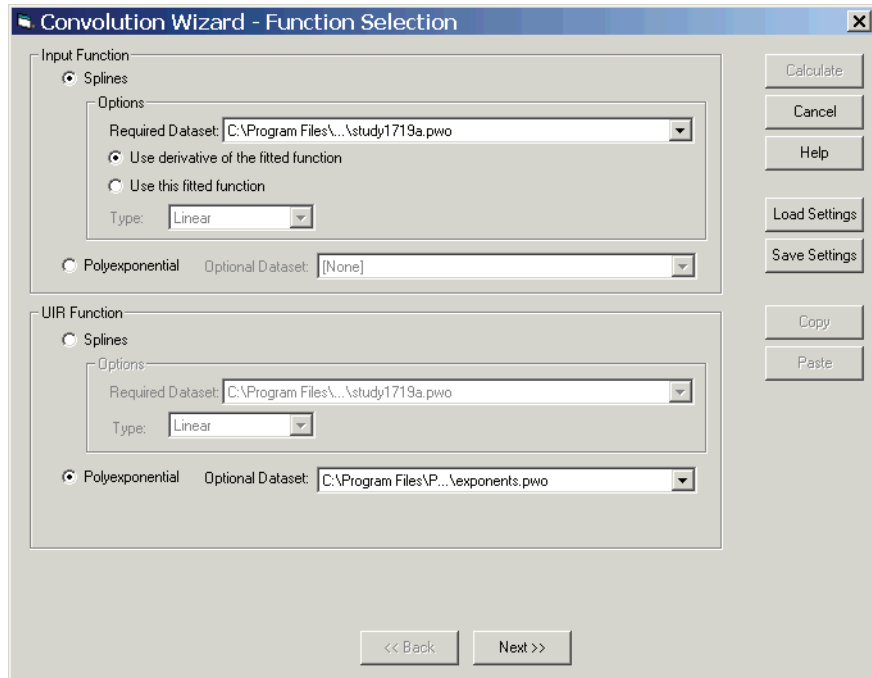
The derivative of a linear spline function is a piecewise constant function, with the constants being the slopes m_i . For input rate data, each piecewise constant spline will have the value that is the average of the y-values of the endpoints. This option is not available for cumulative input data since the derivative will be zero.

To use convolution:

1. Open the data set(s) to be convolved in a WinNonlin workbook.
2. Choose **Tools>IVIVC Tools>Convolution** from the WinNonlin menus.

The Convolution Wizard appears. The first dialog defines the functions $f(t)$ and $g(t)$ described above. Note that $f(t)$ and $g(t)$ are interchangeable; either could represent the input function or unit impulse response function.

3. Select the appropriate function type and settings for the Input function and Unit Impulse Response (UIR) function.
 - *Splines*: Convolution will find the polynomial coefficients. Select the workbook containing time concentration data to be fit under Required Dataset. Select **Use derivative of the fitted function** to fit the data with the derivative of the piecewise polynomial (first of two polynomials only), as for cumulative input data. Select **Use this fitted function** to fit the data with the piecewise polynomial, as for input rate data, and set the degree of polynomial splines (**Linear** or **Constant**). Note that 'linear splines' are required for cumulative input data.
 - *Polyexponential*: Optionally, you may select a dataset containing 1 - 9 sets of coefficient ("A's") and exponent ("alphas") values per subject. If two polyexponentials are to be convolved, none of the alpha values for the first polyexponential can be the same as any of the alpha values for the second polyexponential.
 - See the above discussion for detail on function types and data requirements.



4. Use the **Save** or **Load** button to save or load convolution settings to/from an external file.
5. Click **Next** to proceed to “[Data variables](#)” or, if you selected two polyexponentials without optional input data sets, to proceed to “[Convolution settings](#)”.

Data variables

The options in this dialog vary, depending on the function types selected in the Convolution Function dialog, described under “[Convolution](#)” on [page 295](#). The example below shows the setup for one polynomial and one polyexponential with an optional input data set.

Note: When both functions use an input data set: 1) the data will be joined on any sort keys that have identical names in both data sets; 2) WinNonlin will use the cross-product of any unmatched sort keys.

To set up a polynomial function (splines):

1. *Sort Variables*: Drag variables needed to identify individual profiles to this field. See the above note about sort keys.
2. *Time*: Drag the independent variable to this field. Note that the data set may contain no more than one time record for each profile (unique combination of Sort variable values).
3. *Cumulative Input*: Drag the dependent variable to this field.
4. *Carry Alongs*: Drag data columns to include in the output to this field. These columns will be carried as-is to the results; they are not used in calculations.

Convolution Wizard - Data Variables Selection

Input Function (Splines)
Dataset: C:\Program Files\...\study1719a.pwo

Variables: [Empty]

Sort Variables: Subject, Batch

Time: Time

Cumulative Input: Concentration

Carry Alongs: Study

UIR Function (Polyexponential)
Dataset: C:\Program Files\...\exponent.pwo
Number of exponential terms: 2

Variables: A, alpha, B, beta

Sort Variables: Study, Subject

Carry Alongs: [Empty]

Term	A	Alpha
1	A	alpha
2	B	beta

<< Back Next >>

To set up a polyexponential function that uses a dataset for exponentials:

1. Choose the number of coefficient/exponent pairs under Number of exponential terms.
2. *Sort Variables*: Drag the variables needed to identify individual profiles to this field. See the note about sort keys, above.

3. *Terms Variables*: Drag each coefficient and exponent pair to the A and Alpha fields, respectively, in the same row. All A's must use the same units, as must all Alphas.
4. *Carry Alongs*: Drag data columns to include in the output to this field. These columns will be carried as-is to the results; they are not used in calculations.
5. Click **Next** to proceed to “[Convolution settings](#)”.

Convolution settings

The options in this dialog vary, depending on the function types selected in the Convolution Function dialog, described under “[Convolution](#)” on [page 295](#). The example below shows the setup for one polynomial and one polyexponential without an optional input data set.

To set the number and times of output data points (optional):

1. Enter the total number of output data points under Number of output data points.
2. Select the time option:
 - *Use default times*: (not available for two polyexponentials) Output will include time points from 0 (zero) to one of the following:
 - For two polynomials, the last time value in one of the data sets plus the last time value in the other dataset
 - For one polynomial, twice the last time value in the data set
 - *Output from X to Y*: Output will spread the number of data points evenly across times from *X* to *Y*, inclusive.

Convolution Wizard - Settings

Input Function (Splines)
Dataset: Q:\...\deconvolution_output_2_Subj.pwo
Time column is: Time
Cumulative Input column is: Cumulative_Input

UIR Function (Polyexponential)
Dataset not chosen
Maximum number of exponential terms: 2

	Term	A	Alpha
1	1	1.2	
2	2		

A Units:
Alpha Units: 1/hr

Number of output data points: 101
☒ Use default times
☐ Output from 0 to to

<< Back Next >>

Calculate
Cancel
Help
Load Settings
Save Settings
Copy
Paste

To set up a polyexponential function that did not use a dataset for exponentials:

1. Choose the number of coefficient/exponent pairs to enter under Maximum number of exponential terms.
2. Optionally, enter units for the coefficients (A) and exponents (alphas) to the right. Use the abbreviations detailed in [Table 3-2 on page 54](#). Note that alpha units must equal 1/(Time units). The units can be blank (no units). If the A's and alphas are drawn from a workbook, their column units will apply.
3. *A's and Alphas*: Enter values for each coefficient and exponent pair together in the same row.
4. Click **Calculate** to run the convolution. The output is described under “[Convolution output](#)”.

Convolution output

Convolution generates workbook and chart output. The workbook contains the worksheets listed in [Table 13-2](#). The chart output includes one plot of the convolved data over time for each level of the sort variables. Units in the output are described under “[Convolution units](#)” on page 301.

Table 13-2. Convolution workbook output

Worksheet name	Content
Convolution	Two columns: Time and the convolved data points for each level of the sort variables
Settings	Settings and input workbook names for the convolution
History	Record of operations on the workbook. "History worksheet" on page 37.

Convolution units

The units in convolution output differ with the functions used, as follows.

Table 13-3. Convolution units

Functions	Units in output
Splines and polyexponential, with option "use this fitted function"	If the Time and Input Rate columns have units, and you have provided units for the A's, then the convolved data have the following units. Note that the alpha units must equal 1/(Time units). Time units * Input Rate units * A units
Splines and polyexponential, with option "use derivative of this fitted function"	If the Time and Cumulative Input columns have units, and you have provided units for the A's, then the convolved data have units: Cumulative Input units * A units
Splines and splines, with option "use this fitted function"	The two Time columns must use the same units (if any). If all input columns have units, then the convolved data has units: Time units * Input Rate units * Y units
Splines and splines, with option "use derivative of this fitted function"	The two Time columns must use the same units (if any). If all input columns have units, then the convolved data has units: Cumulative Input units * Y units
Polyexponential and polyexponential	All alpha units must be the same; all A units must also be the same. If alpha and A units are provided, and if the alpha units are 1 over a time unit, then the convolved data has units: Time units from alpha units * A units (1st poly.) * A units (2nd poly.)

Levy plots

Users with a WinNonlin IVIVC Toolkit License can create Levy plots to compare *in-vivo* absorption versus *in-vitro* dissolution data visually, as time-versus-time plots at matched values for absorption and dissolution. The plots include a linear fit of the data points with intersection at origin, and the correlation (R^2) value and the slope for the model line, as well as a line at unity.

Levy plots require two separate workbooks, one containing *in-vivo* and another containing *in-vitro* data. WinNonlin uses these data sets to plot *in-vivo* versus *in-vitro* times at user-specified Y values (typically, fraction absorbed or dissolved), using linear interpolation to calculate data points as needed. Matched time pairs will be sampled from the data within the range from zero to the maximum observed dissolution or deconvolved absorption value, whichever is smaller.

To create one or more Levy plots:

1. Open the *in-vivo* and *in-vitro* data sets in separate WinNonlin workbooks.
2. Choose **Tools>IVIVC Tools>Levy Plots...** from the WinNonlin menus to open the Levy Plots dialog.
3. *Dataset*: Select the appropriate data sets, one each for the *in-vivo* (vertical axis) and *in-vitro* (horizontal axis) profiles, in the corresponding Dataset fields.
4. *Sort Variables* (optional): Drag any variables required to identify separate plots from the Variables list to the Sort Variables field for one or both data sets. The two data sets may contain the same and/or different sort variables. For sort variables that exist in both data sets (identical column name), WinNonlin will include only those sort values that exist in both data sets. For non-matching sort variables, WinNonlin will use the cross-product of sort values to perform the plotting and modeling.
5. *Independent*: For each data set, drag the variable containing sample times from the Variables list to the Independent field.
6. *Values*: For each data set, drag the variable containing sample data from the Variables list to the Values field. Units for these variables, if any, must match in the two data sets.
7. *Formulation* (optional): For each data set, drag the variable identifying different drug formulations from the Variables list to the Formulation field. WinNonlin will group data by formulation within each Levy Chart. Each

formulation will appear in a different color with a corresponding entry in the plot legend. WinNonlin will use only those formulation values that appear in both data sets. Data for formulation values that appear in only one data set will not appear in the chart or workbook output (for a given profile, if sort keys are used). If none of the formulation values match between data sets for any sort key, no output will appear at all.

Note: The summary line on the Levy plot is fit to all formulations combined.

8. *Carry Alongs* (optional): For each data set, drag any additional variables to include in the output workbook from the Variables list to the Carry Alongs field. These variables are not used in the plots.
9. *Plot labels*: If desired, edit the chart title and axis labels in the appropriate fields at the bottom of the dialog box. To enter more than one title line, use **Ctrl+Enter** to insert line breaks.
10. *Values to Match*: WinNonlin will plot *in-vivo* versus *in-vitro* times corresponding to the Y values selected here. The same selections apply across all sort levels.
 - a. *Plot Values at every N*: Select this radio button to plot interpolated time values at regular intervals across the sample data. Enter the interval to the right, in the units of the Y variables. The first value will appear at Y=0, then at the specified interval up to the end of the sample data range. For example, an interval value of 10 would generate plot values at Y = 0, 10, 20, 30, etc. up to the end of the input data. The maximum Y value shown will be lesser of the maximum observed fraction dissolved and the calculated fraction absorbed. WinNonlin will not plot any points that would be extrapolated from one of the data sets.
 - b. *Plot at these values*: Select this radio button and enter a list of Y values to plot interpolated times at those values.
 - c. *Show datapoint labels*: Check this box to display the numeric time values with each data point in the charts as (T(*in-vitro*), T(*in-vivo*)).
 - d. *Datapoints are fractional*: Check this box if the data are expressed as fractions (e.g., 0.02) rather than percents (e.g., 2). WinNonlin will multiply the values by 100 before plotting.

Note: When a Y value to be plotted exists in the data set, WinNonlin will use the data; all other values are computed using simple linear interpolation.

11. Use the **Save Settings** button to save entries in the dialog to an external file for later use. Use the **Load Settings** button to load values from such a file.
12. When all settings are complete, click the **Create Chart** button to generate the Levy plots and accompanying workbook.

Output

The Levy Plot tool outputs one workbook and one plot window. The plot window contains one Levy plot for each level of the sort variables. The workbook contains the following worksheets:

Table 13-4. Levy workbook output

Worksheet	Content
Plotted Values	Values used in the plots, including interpolated and original data values (where interpolation wasn't necessary), including: sort variables (if included) and for each unique combination of sort variable values: Formulation (if included), Matched_Y, Vitro_Time, Vivo_Time, and Carry Along variables (if included).
Predicted Values	Predicted data based on linear regression, including: sort variables, if any, and for each unique combination of sort variable values: X, Y and Carry Along variables (if included).
Parameters	Output from fitting model 502, including estimates and statistics for the Y-intercept (INT) and slope.
History	History of operations on the workbook. See "History worksheet" on page 37 .

The IVIVC Wizard

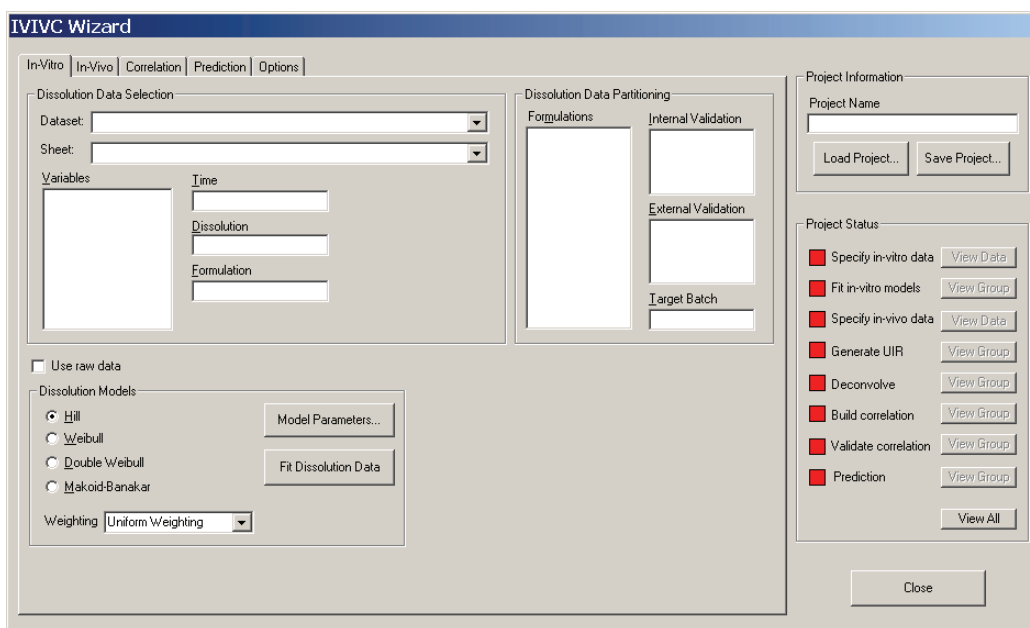
The IVIVC Wizard provides an organized environment in which to build and manipulate *in-vitro in-vivo* correlation (IVIVC) models using the underlying functionality described under ["The IVIVC Toolkit" on page 279](#), plus a feature to automatically estimate the unit impulse response (UIR) for one or more profiles given an oral, IV or IR formulation. The Wizard supports setup and running of all or a subset of operations required to create an IVIVC, validate it and use it to make predictions. It provides workflow support for complete, two-stage IVIVC development. Any set of operations set up in the Wizard can be saved and reloaded as an IVIVC project (*.IVC), including the

related data sets. See the *WinNonlin Examples Guide* for examples demonstrating a full IVIVC workflow.

Note: To maximize IVIVC Wizard operation, set your monitor resolution to a minimum setting of 1024 x 768.

To use the IVIVC Wizard:

1. Open the necessary data set(s) in WinNonlin workbook windows, e.g. one workbook for dissolution data, with one profile per formulation, and another with PK profiles, either individual or average. Note that *in-vitro* and *in-vivo* data must appear in separate workbooks.
2. Close any models that are open in WinNonlin.
3. Choose **Tools>IVIVC>IVIVC Wizard...** from the WinNonlin menus to launch the IVIVC Wizard. You must have a valid WinNonlin license and a valid WinNonlin IVIVC Toolkit License installed in order to access this menu.



Individual functions are run from the separate tabs of the IVIVC Wizard as detailed in the following sections.

- “In-vitro tab” on page 308

- “In-vivo tab” on page 309
- “Correlation tab” on page 313
- “Prediction tab” on page 316
- “Options” on page 318

Note: Output generated from within the IVIVC wizard should be refreshed by saving, reloading and re-running the IVIVC project. The Dependencies dialog will not work for IVIVC Wizard operations and output.

Minimizing the IVIVC Wizard

To avoid potential issues with other WinNonlin dialogs, the IVIVC Wizard remains in the Windows foreground so long as it is open and WinNonlin is not minimized. To minimize the IVIVC Wizard (and leave it open), minimize the main WinNonlin window using the third button from the right end of the WinNonlin title bar.

Project Status

The right hand pane of the wizard shows the status of steps that may be completed in the process of creating, testing and using an IVIVC. Each item appears next to a square that can take the following colors and meanings.

Table 13-5. Project status color codes

Color	Meaning
Green	Process has been run successfully and is up to date.
Yellow	Process is ready to run, but not up to date due to changes to the data in WinNonlin or the settings in the IVIVC Wizard.
Red	Process is not ready to run; one or more required settings or data items is not available.

Use the buttons next to the items to view related data or settings. Note that there is no requirement to complete all items.

Saving and loading IVIVC projects

You may save any collection of settings, input data and output in the IVIVC Wizard as an IVIVC project. Note that any open windows must be closed in WinNonlin before an IVIVC project can be loaded.

Once a user saves an .IVC file, all the WinNonlin objects associated with the IVIVC project will be saved to the same directory.

Note: If the project files must be moved to another location, be sure to move all files (.IVC + workbooks, text, charts) together. It is not possible to load a saved .IVC file if any of the associated files are not present in the same directory.

The WinNonlin objects associated with the IVIVC project can be easily recognized because they have had the IVIVC project name prefix associated with their filenames.

To load a saved IVIVC project:

1. Choose **Tools>IVIVC>IVIVC Wizard...** from the WinNonlin menus.
2. Click the **Load Project** button at the top right corner of the wizard. If any windows are open in WinNonlin, the Wizard will warn you that those objects will be closed. Click **Yes** to proceed.
3. Locate and select the settings (*.IVC) file.
4. Click **Open** to load the project into the IVIVC Wizard.

To save an IVIVC project to your hard drive:

1. Set up the desired project settings in the IVIVC Wizard either manually or by loading a previously-saved project. Run any analyses for which you would like to save data as part of the project. Keep the wizard open.
2. Click the **Save Project...** button at the top right corner of the wizard.
3. In the Save Project dialog, navigate to an appropriate directory and enter any Windows-compatible filename, with or without the file extension (*.IVC).
4. Click **Save**.

Note: Use **PKS>Save** or **PKS>Save As** to save an IVIVC project and, optionally, output, to the PKS.

In-vitro tab

The **IVIVC Wizard**, *In-Vitro* tab includes settings to identify your dissolution data, test and reference (target) formulations, and to fit a dissolution model to smooth the data if desired.

IVIVC Wizard

In-Vitro | In-Vivo | Correlation | Prediction | Options

Dissolution Data Selection

Dataset: C:\Progra...\Phst1426_tabs_diss.dat

Sheet: Sheet1

Variables

Variables	Time	Dissolution	Formulation
	time	diss	form

☐ Use raw data

Dissolution Models

☒ Hill ☐ Weibull ☐ Double Weibull ☐ Makoid-Banakar

Model Parameters...
Fit Dissolution Data

Weighting: Uniform Weighting

Dissolution Data Partitioning

Formulations	Internal Validation	External Validation	Target Batch
2 3	1		

Project Information

Project Name: _____

Load Project... Save Project...

Project Status

- ☒ Specify in-vitro data View Data
- ☒ Fit in-vitro models View Group
- ☒ Specify in-vivo data View Data
- ☒ Generate UIR View Group
- ☒ Deconvolve View Group
- ☒ Build correlation View Group
- ☒ Validate correlation View Group
- ☒ Prediction View Group

View All

Close

Dissolution data selection

Dataset: Select the workbook containing your dissolution data, as fraction dissolved over time, from among those open in WinNonlin.

Sheet: Select the worksheet containing the percent dissolution data.

Variables: Drag column names from the Variables list to the appropriate field on the right to identify those containing time, dissolution and formulation identifiers.

Dissolution data partitioning

Drag and drop a formulation to one or more of the following to identify:

Internal validation (required): Formulation(s) to be used in building the IVIVC model(s) and for validation on the Correlation tab. (See [“Correlation tab” on page 313.](#))

External validation (optional): Formulation(s) to be used in model validation.

Target batch (optional): Formulation to be used as the comparator in predictions, via the Predictions tab. (See [“Prediction tab” on page 316.](#))

Dissolution model

Optionally, in order to provide smoothing you may fit the dissolution data to a sigmoidal dissolution model. Select one of the IVIVC models described under [“Sigmoidal/dissolution models” on page 551](#), then set the following.

Model parameters: 1.3.2.1.1. Depending on the model, you may wish to fix certain parameters to known values. For example, the asymptotic fraction dissolved, F_{inf} , can be fixed to 1.0 to prevent incomplete measurements from biasing other parameter estimates. See [“Model parameters” on page 199](#) for instructions.

Weighting: Select how to weight data during modeling, optionally based on observed Y or predicted Y (\hat{Y}) values. See also [“Weight” on page 207](#).

Click **Fit Dissolution Data** to run the model or linear interpolation and generate output.

Alternately, to use linear interpolation on the raw data instead of modeling, check **Use raw data** above the models.

In-vivo tab

The IVIVC Wizard, *In-Vivo* tab includes settings to identify your time-concentration or absorption data, formulations and dosages. Use this tab to fit a unit impulse response (UIR) function to reference concentration data, and/or to deconvolve an existing UIR and concentration data in order to estimate the fraction of drug absorbed over time.

IVIVC Wizard

In-Vitro **In-Vivo** Correlation Prediction Options

PK Data Selection

Dataset: C:\Documents...\ivivc_vivo_subj.pwo
Sheet: Sheet1

Variables

Variables	Sort Variables	Independent	Values	Formulation
	Subj	Time	Cp	Form

Formulation Information

Reference Formulation: IV
Reference Data Type: IV
Duration of Infusion:
Dose Units: mg

Form	Dose
CR01	1
CR02	1
CR03	1
CR04	1

Copy Paste

Unit Impulse Response

Maximum number of UIR exponentials: 2
Model Selection: Akaike
Weighting: Uniform Weighting
☐ Include Time Lag ☐ Strip Ka
Generate UIR Deconvolve

In Vivo Data Options

☐ Generate mean profiles (validate against mean profiles) Averaging: Mean
☒ Do not generate mean profiles (validate against individuals' profiles)
☐ Data already deconvolved and averaged

Project Information

Project Name: ivivc3
Load Project... Save Project...

Project Status

- ☒ Specify in-vitro data View Data
- ☒ Fit in-vitro models View Group
- ☒ Specify in-vivo data View Data
- ☒ Generate UIR View Group
- ☒ Deconvolve View Group
- ☒ Build correlation View Group
- ☒ Validate correlation View Group
- ☒ Prediction View Group

View All

Close

PK data selection

Dataset: Select the workbook containing your time-concentration or time-absorption data from among those open in WinNonlin.

Sheet: Select the worksheet containing the data.

Variables: Drag column names from the Variables list to the appropriate field on the right to identify those identifying individual profiles (Sort variables), time (Independent), concentration (Values) or fraction absorbed (Absorption), and drug formulation identifiers (Formulation).

Formulation information

Reference formulation: Select the identifier for the formulation to be used in computing the UIR (and also in building IVIVC correlation models). The reference formulation must use all oral, IV bolus or IV infusion input. The IVIVC Wizard does not support combinations of inputs for UIR calculation.

Reference data type: Identify whether the reference input was oral, IV bolus or IV infusion.

Dose units: Optionally, enter the dose units. Enter the dosage amount for each formulation in the table below, in the units identified above. For an infusion, enter the infusion time for each formulation.

Unit impulse response

Based on the reference formulation you select, the Wizard can compute UIR's for all subjects. It will fit a polyexponential function to each profile and choose the one with the best fit based on Akaike Information Criterion or Weighted SSR, as chosen under Model Selection. (See "AIC" on page 487 and "WRSS" on page 504, respectively.)

Include time lag: For an oral reference formulation, check this box to include a lag time parameter in the UIR model.

Strip Ka: For an oral reference formulation, check this box to generate the polyexponential describing the IV bolus PK. This assumes that absorption is truly first order and fits a model that is n -compartment polyexponential, convolved with first-order absorption. The absorption is mathematically separated from the decay. If this option is not checked, the UIR will simply be a polyexponential model of the reference data.

Click **Generate UIR** when all settings are ready to fit the unit impulse response function(s) and generate related charts and workbook output. Then, to compute fraction absorbed over time, click **Deconvolve** to perform numeric deconvolution. The IVIVC Wizard will use the patient and formulation sort keys specified to determine individual profiles to deconvolve, using the doses entered into the Wizard and the UIR's just generated. It will generate estimates for fraction absorbed at 200 time points.

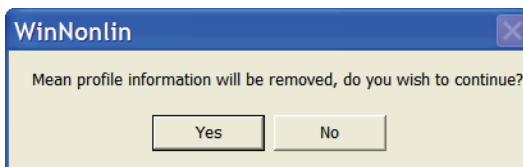
If you have already determined the UIR and (average) Fraction Absorbed, you must still follow the above procedure to generate the output workbooks in their appropriate format. Then check **Data already deconvolved and averaged** to make two of the output workbooks editable. In the Fa-avg.pwo workbook, the Time and Mean columns are editable. In UIR.pwo, the A* and alpha* columns on the Transposed Final Parameters sheet are editable. You can paste in the values that you prefer to use.

In Vivo Data Options

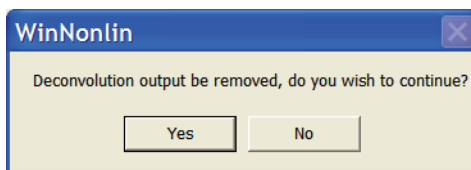
In the **In Vivo Data Options** area, users can select three averaging options for observed, or in-vivo, data: Generate mean profiles (validate against mean profiles), Do not generate mean profiles (validate against individuals' profiles),

and Data already deconvolved and averaged. Only one option can be selected in the In Vivo Data Options area.

If users change any of the In Vivo Data Options after deconvolution is performed, WinNonlin displays the following message to warn users that mean profile information will be removed from the project:



- **Generate mean profiles (validate against mean profiles):** the default option. Users must select an input data set in the Dataset menu. The data set is used to generate unit-impulse-response parameters, which are then deconvolved and averaged.
- In the **Averaging** menu, users can select **Mean** or **GeoMean**.
 - **Mean:** arithmetic means are generated.
 - **GeoMean:** geometric means are generated.
- **Do not generate mean profiles (validate against individuals' profiles):** if this option is selected, then %_{PE} AUC and C_{max} values are calculated during the validation and prediction phases using predicted vs. mean (or geomean)-observed time and concentration data. Means are generated directly from the input data set before any other IVIVC operations.
- **Data already deconvolved and averaged:** if the selected input data set has already been deconvolved and averaged, users can choose to have WinNonlin skip these steps. No UIR generation, deconvolution, or averaging is performed on the input data set when this option is selected. Data that is already deconvolved and averaged is validated against individual profiles.
- If Data already deconvolved and averaged is selected then the following three things happen:
 - WinNonlin displays a message warning users that any current deconvolution output in the IVIVC project will be removed:



- All of the Unit Impulse Response options are made unavailable to the user.
- The Fa-Avg and Fa-Avg by Profile plots are not created. If they already exist in the IVIVC project, they are removed.

Note: If any IVIVC operations are dependent on deconvolved and averaged data, and the Averaging Options are changed, then the project phase that contains the dependent operation turns yellow in the Project Status area.

Correlation tab

The IVIVC Wizard, Correlation tab supports creation and validation of an IVIVC correlation model for dissolution data defined as “internal” on the In-Vitro tab, and absorption data set up or estimated via the In-Vivo tab. See “[In-vitro tab](#)” on page 308 and “[In-vivo tab](#)” on page 309. The correlation will be built using those formulations identified as “internal” (or “reference”) on the In-Vitro and In-Vivo tabs. It will fit the selected correlation model to absorption data using the dissolution profiles from the In-Vitro tab as inputs to the correlation function.

To validate the correlation, the Wizard can estimate AUC’s and maximum concentrations based on the model predictions, and compare to actual values from the *in-vivo* data. To do so, it will convolve predicted fractions absorbed and UIR’s from the other tabs, then run noncompartmental analysis on that predicted PK data and the observed *in-vivo* data.

- Click **Generate Plots** to create the plots.

Note: If a correlation is validated, the internal Validation worksheet is saved as VALIDATION ERRORS.PWO if the IVIVC project is saved.

Validation

AUC: Select which computation of area under the curve to use in the validation: from dose time through the last positive concentration value (AUClast), from dose time through the final observation time (AUCall), AUClast + Clast/Lambda Z, where Clast is the last predicted concentration (AUCINF), or AUClast + Clast/Lambda Z, where Clast is the last observed concentration (AUCINF_obs).

Calculation method: You can choose to calculate AUC using either the linear, linear/log, or linear up/log down trapezoidal rule. (See “[Calculation methods for AUC](#)” on page 179 and “[AUC calculation and interpolation formulas](#)” on page 185 for definitions and calculation methods.)

Average: Select whether to compute averages as the mean or geometric mean.

Click **Run Validation** to generate a table of prediction errors for AUC and Cmax:

$$\%_PE = 100 * (\text{Predicted} - \text{Observed}) / \text{Observed}.$$

For “Avg Internal,” the average is taken over the absolute values of %_PE for the formulations specified as Internal.

ASCII user model for correlations

A custom model for an IVIVC correlation (see “[Correlation tab](#)” on page 313) needs to include only the parameter definitions and model equations. No other statements are required.

Enter the number of model parameters in the topmost field near the left side of this dialog, and enter its initial value below. Edit the statements to the right to reflect your model. Use the syntax of the WinNonlin modeling language detailed under “[User Models](#)” on page 219, and the following keywords that are specific to IVIVC.

IVINTERP() - This function is used to interpolate dissolution data. A common usage is for X to be the in vivo time, and X is then scaled or shifted by model parameters to yield an in vitro time, which is called T.

IVINTERP is then used to interpolate the dissolution data to find the dissolution at time T. The first and last values in the dissolution data are used when T is respectively before or after the time range of the dissolution data.

DOSEVIVO - This variable contains the dose amount for each formulation. The wizard will translate this variable into the DTA array that is part of the WNL modeling language.

IVIVC Correlation User Model

Edit your custom model here. The wizard will generate most of the required ASCII modeling statements during runtime. The required components that need to be entered here include the number of parameters, the parameter names, initial values for those parameters and the actual model statements. The model statements should use the new DISS keyword and IVINTERP() function, and may optionally use the new DOSE variable.

Number of custom model parameters:

Custom parameter names and initial estimates (8-char limit)

Parameter	Initial
A1	1
A2	2
B2	3

Modeling statements

```
T = A2 * X - B2
IF T > 0 THEN
  DISS = IVINTERP(T)
ELSE
  DISS = 0
ENDIF
F = A1 * DISS
```

OK Cancel Help

Copy Paste

Check Model Syntax...

Prediction tab

The **IVIVC Wizard**, Prediction tab generates a table of prediction errors for AUC and Cmax by comparing predicted data for test formulations on the Prediction tab to the target formulation identified on the In-Vitro tab.

IVIVC Wizard

Dissolution Data Selection

Dataset: C:\Progra... \Phst1426_tabs_test.dat

Sheet: Sheet1

Variables

Time
time
Dissolution
dis
Formulation
form

Formulation Information

Dose Units: mg

Form	Dose
Test1	100

Copy Paste

Project Information

Project Name

Load Project... Save Project...

Project Status

- ☒ Specify in-vitro data View Data
- ☒ Fit in-vitro models View Group
- ☒ Specify in-vivo data View Data
- ☒ Generate UIR View Group
- ☒ Deconvolve View Group
- ☒ Build correlation View Group
- ☒ Validate correlation View Group
- ☒ Prediction View Group

View All

Prediction

*AUClast (Mean) calculated by Linear Trapezoidal (Linear Interpolation)

Formulation	Parameter	Predicted	Target	%_PE	Ratio
-------------	-----------	-----------	--------	------	-------

Copy Prediction To Clipboard

Predict PK

Dissolution Model

☐ Hill ☐ Double Weibull ☐ Weibull ☐ Makoid-Banakar

Weighting: Uniform Weighting

Model Parameters... Fit Dissolution Data

Close

Dissolution data selection

Dataset: Select the workbook containing your dissolution data, as fraction dissolved over time, from among those open in WinNonlin.

Sheet: Select the worksheet containing the percent dissolution data.

Variables: Drag column names from the Variables list to the appropriate field on the right to identify those containing time, fraction dissolved and formulation identifiers.

Formulation information

Dose units: Optionally, enter the dose units for the formulations.

Enter the dosage amount for each formulation in the table below, in the units identified above.

Dissolution model

Optionally, in order to provide smoothing you may fit the dissolution data to a sigmoidal dissolution model. Select one of the IVIVC models described under “Sigmoidal/dissolution models” on page 551, then set the following.

Model parameters: 1.3.2.1.1. Depending on the model, you may wish to fix certain parameters to known values. For example, the asymptotic fraction dissolved, *Finf*, can be fixed to 1.0 to prevent incomplete measurements from biasing other parameter estimates. See [“Model parameters” on page 199](#) for instructions.

Weighting: Select how to weight data during modeling, optionally based on observed *Y* or predicted *Y* (Yhat) values. See also [“Weight” on page 207](#).

Click **Fit Dissolution Data** to run the model or linear interpolation and generate output.

Alternately, to use linear interpolation on the raw data instead of modeling, check **Use raw data** above the models.

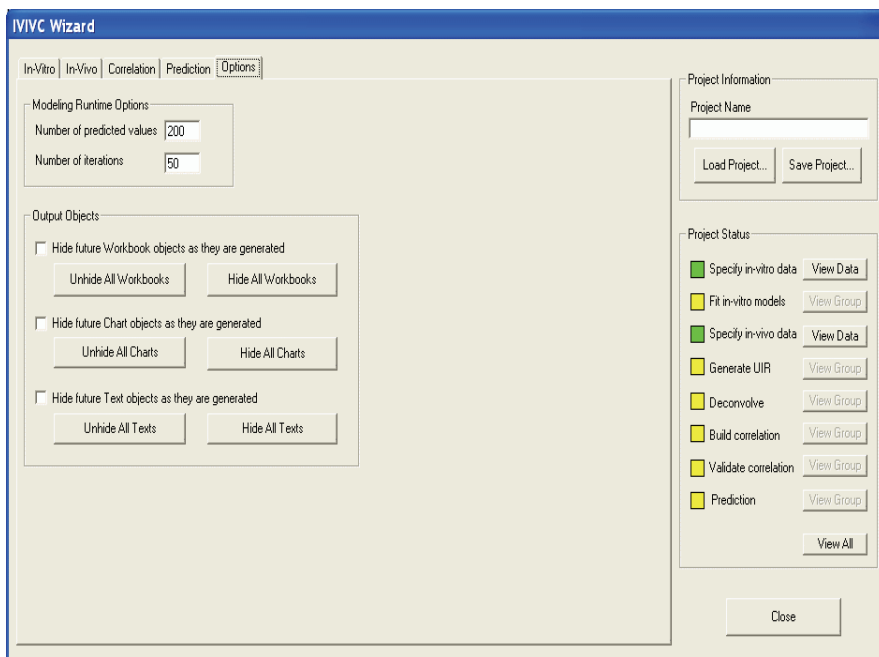
Prediction

Click **Predict PK** to generate the table of prediction errors for AUC and Cmax, comparing test formulations to the target formulation identified on the In-Vivo tab.

Note: If PK output is predicted, the internal Prediction worksheet is saved as PREDICTION ERRORS.PWO if the IVIVC project is saved.

Options

The [IVIVC Wizard](#), Options tab includes settings that control how operations on the other tabs will behave.



Modeling runtime options

Enter the number of predicted values to be generated by modeling operations initiated in the IVIVC Wizard and the maximum allowed number of iterations during model fitting.

Output objects

Select whether to hide specific output windows generated by the IVIVC Wizard in the WinNonlin window. See also [“Hidden windows” on page 18](#).

IVIVC Wizard Output Objects

File Name	Description	Process	Object Types
“Avg PE Stats Out”	Average absolute prediction error statistics for ‘internal’ formulations	Validate Correlation	Workbook

File Name	Description	Process	Object Types
"Corr Abs vs Diss"	Correlation chart of FAbs vs. FDis	Build Correlation	Chart
"Corr Model Out"	Correlation modeling output	Build Correlation	Workbook Text Chart
"Corr Model Sim Vitro"	Input vitro data to correlation model simulation for validation step	Validate Correlation	Workbook
"Corr Model Sim Vivo"	Input vivo data to correlation model simulation for validation step	Validate Correlation	Workbook
"Corr Model Vitro In"	Fraction dissolved data going into the correlation model fitting	Build Correlation	Workbook
"Corr Model Vivo In"	Fraction absorbed data going into the correlation model fitting	Build Correlation	Workbook
"Corr Plot Out"	Correlation plot using [Fa-avg] and modeled dissolution data	Generate Plots	Chart
"Deconvolution Out"	Deconvolution output – fraction absorbed vs. time for each profile	Deconvolve	Workbook Chart
"Dissolution Out"	Simulated dissolution data points, using fitted model selected on "In-Vitro" tab	Fit Dissolution Data	Workbook Text Chart

File Name	Description	Process	Object Types
"Fa-Avg"	Average fraction absorbed by formulation. For the chart, Fabs grouped by subject with Fabs_avg overlaid and sorted by formulation	Deconvolve	Workbook Chart (the chart is not created if the Data already deconvolved and averaged option is selected, or if no sort variables are selected in the In-Vivo tab)
"Fa-Avg by Profile"	Fabs grouped by formulation, vs. time, and sorted by profile	Deconvolve	Chart (the chart is not created if the Data already deconvolved and averaged option is selected, or if no sort variables are selected in the In-Vivo tab)
"Levy Plot Out"	Levy plot using [Fa-avg] and modeled dissolution data	Generate Plots	Chart
"Pred Conv Out"	Prediction convolution output – individual profiles of UIRs convolved with simulated correlation output	Predict PK	Workbook Chart
"Pred Corr Sim Out"	Simulated fraction absorbed output from correlation model	Predict PK	Workbook Chart
"Pred Corr Sim Vitro"	Input vitro data to correlation model simulation for prediction step	Predict PK	Workbook
"Pred Corr Sim Vivo"	Input vivo data to correlation model simulation for prediction step	Predict PK	Workbook

File Name	Description	Process	Object Types
"Pred Dissolution Out"	Simulated dissolution data points, using fitted model selected on "Prediction" tab	Predict PK	Workbook Text Chart
"Prediction Errors"	Internal error prediction worksheet, created if the IVIVC project is saved.	Predict PK	Workbook
"Pred PK NCA In"	Predicted concentration profiles, sampled at in vivo times, for input to NCA	Predict PK	Workbook
"Pred PK NCA Out"	NCA results for predicted profiles	Predict PK	Workbook
"Pred Stats Out"	NCA outputs averaged by formulation for target and predicted data, with PE computations.	Predict PK	Workbook
"Pred Target NCA"	NCA results from in vivo dataset for 'Target' formulation	Predict PK	Workbook
"UIR"	Unit Impulse Response (UIR) modeling output	Generate UIR	Workbook Text Chart
"Val Baseline NCA Out"	NCA results from in vivo dataset	Validate Correlation	Workbook
"Val Conv Out"	Validation convolution output – individual profiles of UIRs convolved with simulated correlation output	Validate Correlation	Workbook Chart

File Name	Description	Process	Object Types
"Val Corr Sim Out"	Simulated correlation output using "internal" dissolution profiles	Validate Correlation	Workbook Chart
"Validation Errors.pwo"	Internal validation errors worksheet, created if the IVIVC project is saved.	Validate Correlation	Workbook
"Val PK NCA In"	Validation convolution output sampled at in vivo times- goes into NCA	Validate Correlation	Workbook
"Val PK NCA Out"	NCA results from [Val PK NCA In.pwo]	Validate Correlation	Workbook
"Val Stats Out"	NCA outputs averaged by formulation for observed and predicted data, with PE computations.	Validate Correlation	Workbook

Using the IVIVC Wizard with PKS

Initiating a new IVIVC Project in an existing PKS scenario

Use these steps to use an existing PKS scenario in an IVIVC analysis.

1. Load a scenario.
2. Create workbooks for in-vivo, in-vitro, or test in-vitro data.
3. Use the IVIVC Wizard to perform the analysis.
4. Save the IVIVC project using the instructions under [Saving an IVIVC Project to PKS](#).

Loading a IVIVC Project from PKS

Use these steps to load an IVIVC project that has been saved to a PKS scenario.

1. In the **PKS** menu, select **Load**.
2. Log on to PKS and select the scenario.
3. Click **OK** to load the IVIVC project into WinNonlin.

Merging an Existing IVIVC Project into PKS

Use these steps when there is an existing IVIVC project that is saved to disk but needs to be merged into an existing PKS scenario.

Note: Before beginning the merging instructions, open the IVIVC project in the IVIVC Wizard. Then save the project to an empty directory. This allows users to use Windows File Explorer to drag and drop all files, except the .IVC file, into WinNonlin to open them.

1. Load a scenario from PKS.
2. Open all IVIVC project files including workbooks, text files, charts, and the .IVC file from their disk locations.
3. Open the IVIVC Wizard.
4. Click the **Load Project** button and select the .IVC file to load the IVIVC project.
5. Save the IVIVC project using the instructions under [Saving an IVIVC Project to PKS](#).

Saving an IVIVC Project to PKS

Use these steps to save an IVIVC project to a PKS Scenario.

- In the **PKS** menu, select **Save** or **Save As**.
 - Save saves the existing scenario, and Save As allows users to create a new scenario.

Users do not need to save any IVIVC project files to a disk before saving them in PKS.

Linear Mixed Effects Modeling

ANOVA, ANCOVA, and regression models

The Linear Mixed Effects function (LinMix) performs analyses using linear mixed effects models. It is a statistical analysis system for analysis of variance for crossover and parallel studies, including unbalanced designs. It can analyze regression and covariance models, and can calculate both sequential and partial tests. LinMix is discussed in the following sections.

- “[An example](#)”: Since linear mixed effects modeling is most easily introduced using an example, this chapter starts by creating a statistical model for a common experimental design that utilizes several error terms, as well as classical ANOVA modeling terms.
- “[The linear mixed effects model](#)” on page 330 discusses LinMix general theory in more detail.
- “[LinMix Wizard](#)” on page 354 provides stepped instructions.
- “[LinMix output](#)” on page 365 and “[Computational details](#)” on page 370 cover LinMix calculations and output parameters.
- “[Comparing LinMix to ANOVA and SAS’s PROC MIXED](#)” on page 376 explains similarities and differences among three analysis packages.

An example

The following example is used as a framework for discussing the functions and computations of the [Linear Mixed Effects Modeling](#) wizard.

An investigator wants to estimate the difference between two treatments for blood pressure, labeled A and B. Because responses are fairly heterogeneous among people, the investigator decided that each study subject will be his own control, which should increase the power of the study. Therefore, each subject

will be exposed to treatments A and B. Since the treatment order might affect the outcome, half the subjects will receive A first, and the other half will receive B first. For the analysis, note that this crossover study has two periods, 1 and 2, and two sequence groups AB and BA. The model will be of the form:

$$y_{ijkm} = \mu + \pi_i + \delta_j + \tau_q + S_{k(j)} + \varepsilon_{ijkm} \quad (1)$$

where:

i = period index (1 or 2)

j = sequence index (AB or BA)

k = subject within sequence group index (1 ... n_j)

m = observation within subject index (1, 2)

q = treatment index (1, 2)

μ = intercept

π_i = effect due to period

δ_j = effect due to sequence grouping

τ_q = effect due to treatment, the purpose of the study

$S_{k(j)}$ = effect due to subject

ε_{ijkm} = random variation

Typically, ε_{ijkm} is assumed to be normally distributed with mean 0 and variance $\sigma^2 > 0$. The purpose of the study is to estimate $\tau_1 - \tau_2$ in all people with similar inclusion criteria as the study subjects. Study subjects are assumed to be randomly selected from the population at large. Therefore, it is desirable to treat subject as a random effect. Hence, assume that subject effect is normally distributed with mean 0 and variance $\text{Var}(S_{k(j)}) \geq 0$.

This mixing of the regression model and the random effects is known as a mixed effect model. The mixed effect model allows one to model the mean of the response as well as its variance. Note that in model (1), the mean is:

$$E\{y_{ijkm}\} = \mu + \pi_i + \delta_j + \tau_q \quad (2)$$

and the variance is given by:

$$\text{Var}\{y_{ijkm}\} = \text{Var}\{S_{k(j)}\} + \sigma^2. \quad (3)$$

$E\{y_{ijkm}\}$ is known as the fixed effects of the model, while $S_{k(j)} + \epsilon_{ijkm}$ constitutes the random effects, since these terms represent random variables whose variances are to be estimated. A model with only the residual variance is known as a fixed effect model. A model without the fixed effects is known as a random effect model. A model with both is a mixed effect model.

The data are shown in Table 14-1. The LinMix wizard is used to analyze these data. The fixed and random effects of the model are entered on different pages of the wizard. This is appropriate since the fixed effects model the mean of the response, while the random effects model the variance of the response.

Table 14-1. Data for the example

Subject	Sequence	Period	Treatment	Response
1	AB	1	A	9.70
2	AB	1	A	10.24
3	AB	1	A	11.2
4	AB	1	A	7.82
5	AB	1	A	11.1
6	AB	1	A	9.31
7	BA	2	A	9.02
8	BA	2	A	7.88
9	BA	2	A	9.6
10	BA	2	A	9.63
11	BA	2	A	9.63
12	BA	2	A	9.91
7	BA	1	B	8.15
8	BA	1	B	9.23
9	BA	1	B	9.43
10	BA	1	B	10.13
11	BA	1	B	9.67

Table 14-1. Data for the example (continued)

Subject	Sequence	Period	Treatment	Response
12	BA	1	B	11.34
1	AB	2	B	8.72
2	AB	2	B	11.28
3	AB	2	B	11.73
4	AB	2	B	9.77
5	AB	2	B	8.91
6	AB	2	B	8.31

LinMix can fit linear models to Gaussian data. The mean and variance structures can be simple or complex. Independent variables can be continuous or discrete, and no distributional assumption is made about them.

The screenshot shows the 'LinMix Fixed Effects' dialog box. It contains several input fields and buttons. The 'Classification Variables' list includes 'Subject', 'Sequence', 'Treatment', and 'Period'. The 'Dependent Variables' list includes 'Response'. The 'Model Specification' text box contains 'Subject+Sequence+Period'. The 'Fixed Effects Confidence Level' is set to '95 %'. The 'Weight Variable' field is empty. The 'Dependent Variables Transformation' dropdown is set to 'None'. The 'No Intercept' checkbox is unchecked. The buttons include 'Clear Model', 'Load...', 'Save As...', 'Previous Model...', 'Help', '< Back', 'Next >', 'Calculate', and 'Cancel'.

This model provides a context for “[The linear mixed effects model.](#)”

The linear mixed effects model

WinNonlin’s [Linear Mixed Effects Modeling](#) feature is based on the general linear mixed effects model, which can be represented in matrix notation as:

$$y = X\beta + Z\gamma + \varepsilon$$

where:

X is a matrix of known constants and the design matrix for β ,

β is a vector of unknown (fixed effect) parameters,

γ is a random unobserved vector (the vector of random-effects parameters), normally distributed with mean 0 and variance G ,

Z is a matrix of known constants, the design matrix for γ ,

and ε is a random vector, normally distributed with mean 0 and variance R .

γ and ε are uncorrelated. In the LinMix wizard, the fixed effects dialog specifies the model terms for constructing X ; the random effects tabs in the Variance Structure dialog specifies the model terms for constructing Z and the structure of G ; and the repeated tab specifies the structure of R .

LinMix breaks the model into the following areas:

- Fixed effects
- Variance structure
- Least squares means
- Contrasts
- Estimates

Note: In the linear mixed effects model, if $R = \sigma^2 I$ and $Z = 0$, then the LinMix model reduces to the ANOVA model. Using the assumptions for the linear mixed effects model, one can show that the variance of y is:

$$V = ZGZ^T + R$$

If $Z \neq 0$, i.e., the model includes one or more random effects, the G matrix is determined by the variance structures specified for the random models. If the model includes a repeated specification, then the R matrix is determined by the variance structure specified for the repeated model. If the model doesn't include a repeated specification, then R is the residual variance—i.e., $R = \sigma^2 I$.

It is instructive to see how a specific example fits into this framework. The fixed effects model is shown in equation 2 under “An example” on page 327. Listing all the parameters in a vector, one obtains:

$$\beta^T = [\mu, \pi_1, \pi_2, \delta_1, \delta_2, \tau_1, \tau_2]$$

For each element of β , there is a corresponding column in the X matrix. In this model, the X matrix is composed entirely of 1's and 0's. The exact method of constructing it is discussed under “[Construction of the X matrix](#)”, below.

For each subject, there is one $S_{k(j)}$. The collection can be directly entered into γ . The Z matrix will consist of 1's and 0's. Element (i, j) will be 1 when observation i is from subject j , and 0 otherwise. The variance of γ is of the form

$$G \equiv \text{Var}(\gamma) = \text{Var}\{S_{k(j)}\} I_{n(1)+n(2)}$$

where I_b represents a $b \times b$ identity matrix. The parameter to be estimated for G is $\text{Var}\{S_{k(j)}\}$. The residual variance is

$$R = \text{Var}\{\epsilon_{ijkm}\} = \sigma^2 I_{2[n(1)+n(2)]}$$

The parameter to be estimated for R is σ^2 .

A summary of the wizard pages and the part of the model they generate is listed below. The details will be discussed later.

Wizard Page	Specifies	Tabs on page	Creates
Fixed Effects	Specifies X matrix		Generates β
Variance Structure	Specifies Z matrix	Random tabs (1 or more)	Generates $G = \text{var}(\gamma)$
		Repeated tab (exactly 1)	Generates $R = \text{var}(\epsilon)$

Construction of the X matrix

A term is a variable or a product of variables. A user specifies a model by giving a sum of terms. For example, if A and B are variables in the data set:

$$A + B + A*B.$$

LinMix takes this sum of terms and constructs the X matrix. The way this is done depends upon the nature of the variables A and B .

The No Intercept
option

By default, the first column of X contains all ones. The LinMix wizard references this column as the intercept. To exclude it, check **No Intercept** in the Fixed Effects dialog of the wizard. The rules for translating terms to columns of the X matrix depend on the variable types, regressor/covariate or classification. This is demonstrated using two classification variables, Drug and Form,

and two continuous variables, Age and Weight, in Table 14-2. A term containing one continuous variable produces a single column in the X matrix.

An integer value is associated with every level of a classification variable. The association between levels and codes is determined by the numerical order for converted numerical variables and by the character collating sequence for converted character variables. When a single classification variable is specified for a model, an indicator variable is placed in the X matrix for each level of the variable. Table 14-3 demonstrates this for the variable Drug.

The model specification allows products of variables. The product of two continuous variables is the ordinary product within rows. The product of a classification variable with a continuous variable is obtained by multiplying the continuous value by each column of the indicator variables in each row. For example, Table 14-3 shows the product of Drug and Age. Table 14-4 shows two classification and indicator variables. Table 14-5 gives their product.

The product operations described in the preceding paragraph can be succinctly defined using the Kronecker product^A. For example, consider the fifth row of Table 14-3. The product of Drug and Age is:

$$[0 \ 1 \ 0] \otimes [45] = [0 \ 45 \ 0].$$

Now consider the fifth row of Table 14-4. The product of Drug and Form is:

$$[0 \ 1 \ 0] \otimes [1 \ 0 \ 0 \ 0] = [0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]$$

The result is the fifth row of Table 14-5.

Table 14-2. Variables in sample data set

Drug		Form		Age	Weight
Name	Value	Name	Value		
DrugA	0	capsule	0	48	120
DrugA	0	drops	1	34	124
DrugA	0	syrup	2	27	123
DrugA	0	tablet	3	23	172
DrugB	1	capsule	0	45	200

A. Let $A = (a_{ij})$ be a $m \times n$ matrix and let $B = (b_{ij})$ be a $p \times q$ matrix. Then the Kronecker product $A \otimes B = (a_{ij} B)$ is an $mp \times nq$ matrix expressible as a partitioned matrix with $a_{ij} B$ as the (i, j) th partition, $i=1, \dots, m$ and $j=1, \dots, n$.

Table 14-2. Variables in sample data set (continued)

Drug		Form		Age	Weight
Name	Value	Name	Value		
DrugB	1	drops	1	26	150
DrugB	1	syrup	2	41	120
DrugB	1	tablet	3	32	130
DrugC	2	capsule	0	38	190
DrugC	2	drops	1	17	125
DrugC	2	syrup	2	32	175
DrugC	2	tablet	3	23	150

Table 14-3. Indicator variables and a product of a classification variable with a continuous variable

Drug		Indicator variables			Age	Drug*Age		
Name	Value	DrugA	DrugB	DrugC		DrugA	DrugB	DrugC
DrugA	0	1	0	0	48	48	0	0
DrugA	0	1	0	0	34	34	0	0
DrugA	0	1	0	0	27	27	0	0
DrugA	0	1	0	0	23	23	0	0
DrugB	1	0	1	0	45	0	45	0
DrugB	1	0	1	0	26	0	26	0
DrugB	1	0	1	0	41	0	41	0
DrugB	1	0	1	0	32	0	32	0
DrugC	2	0	0	1	38	0	0	38
DrugC	2	0	0	1	17	0	0	17
DrugC	2	0	0	1	32	0	0	32
DrugC	2	0	0	1	23	0	0	23

Table 14-4. Two classification variables and their indicator variables

Drug		Indicators			Form		Indicators			
Name	Val.	DrugA	DrugB	DrugC	Name	Val.	capsule	drops	syrup	tablet
DrugA	0	1	0	0	capsule	0	1	0	0	0
DrugA	0	1	0	0	drops	1	0	1	0	0
DrugA	0	1	0	0	syrup	2	0	0	1	0
DrugA	0	1	0	0	tablet	3	0	0	0	1
DrugB	1	0	1	0	capsule	0	1	0	0	0
DrugB	1	0	1	0	drops	1	0	1	0	0
DrugB	1	0	1	0	syrup	2	0	0	1	0
DrugB	1	0	1	0	tablet	3	0	0	0	1
DrugC	2	0	0	1	capsule	0	1	0	0	0
DrugC	2	0	0	1	drops	1	0	1	0	0
DrugC	2	0	0	1	syrup	2	0	0	1	0
DrugC	2	0	0	1	tablet	3	0	0	0	1

Table 14-5. Product of the two classification variables

Drug	Form	Indicator											
DrugA	capsule	1	0	0	0	0	0	0	0	0	0	0	0
DrugA	drops	0	1	0	0	0	0	0	0	0	0	0	0
DrugA	syrup	0	0	1	0	0	0	0	0	0	0	0	0
DrugA	tablet	0	0	0	1	0	0	0	0	0	0	0	0
DrugB	capsule	0	0	0	0	1	0	0	0	0	0	0	0
DrugB	drops	0	0	0	0	0	1	0	0	0	0	0	0
DrugB	syrup	0	0	0	0	0	0	1	0	0	0	0	0
DrugB	tablet	0	0	0	0	0	0	0	1	0	0	0	0
DrugC	capsule	0	0	0	0	0	0	0	0	1	0	0	0
DrugC	drops	0	0	0	0	0	0	0	0	0	1	0	0

Table 14-5. Product of the two classification variables (continued)

Drug	Form	Indicator											
DrugC	syrup	0	0	0	0	0	0	0	0	0	0	1	0
DrugC	tablet	0	0	0	0	0	0	0	0	0	0	0	1

Linear combinations of elements of β

Most of the inference done by LinMix is done with respect to linear combinations of elements of β . Consider, as an example, the model where the expected value of the response of the j^{th} subject on the i^{th} treatment is:

$$\mu + \tau_i \quad (4)$$

where μ is a common parameter and $\mu + \tau_i$ is the expected value of the i^{th} treatment. Suppose there are three treatments. The X matrix has a column of ones in the first position followed by three treatment indicators. The β vector is:

$$\begin{aligned} \beta &= [\beta_0 \ \beta_1 \ \beta_2 \ \beta_3] \\ &= [\mu \ \tau_1 \ \tau_2 \ \tau_3] \end{aligned}$$

The first form is for the general representation of a model. The second form is specific to model (4) above. The expression:

(5)

$$\sum_j l_j \beta_j$$

with the l_j constant is called a linear combination of the elements of β . The range of the subscript j starts with 0 if the intercept is in the model and starts with 1 otherwise. Most common functions of the parameters, such as $\mu + \tau_1$, $\tau_1 - \tau_3$, and τ_2 , can be generated by choosing appropriate values of l_j . Linear combinations of β are entered in LinMix through the wizard's Estimates and Contrasts pages. A linear combination of β is said to be estimable if it can be expressed as a linear combination of expected responses. In the context of

model 4, a linear combination of the elements of β is estimable if it can be generated by choosing c_1 , c_2 , and c_3 .

$$c_1(\mu + \tau_1) + c_2(\mu + \tau_2) + c_3(\mu + \tau_3). \quad (6)$$

Note that $c_1=1$, $c_2=0$, and $c_3=0$ generates $\mu + \tau_1$, so $\mu + \tau_1$ is estimable. A rearrangement of expression (6) is:

$$(c_1 + c_2 + c_3)\mu + c_1\tau_1 + c_2\tau_2 + c_3\tau_3. \quad (7)$$

The form in (7) makes some things more obvious. For example, $l_j = c_j$ for $j=1,2,3$. Also, any linear combination involving only τ s is estimable if, and only if, $l_1 + l_2 + l_3 = 0$. A linear combination of effects, where the sum of the coefficients is zero, is called a contrast. The linear combination $\tau_1 - \tau_3$ is a contrast of the τ 's and thus is estimable. The linear combination τ_2 is not a contrast of the τ 's and thus is not estimable.

Determining estimability numerically

The X matrix for the model in equation (4) has three distinct rows, and these are shown in Table 14-6. Also shown in Table 14-6 is a vector (call it z) that is a basis of the space orthogonal to the rows of X . Let l be a vector whose elements are the l_j of expression (5). The linear combination 5 in vector notation is $l^T b$ and is estimable if $l^T Z = 0$. Allow for some rounding error in the computation of z . The linear combination is estimable if:

$$\frac{l^T Z}{\|l\|_\infty \sqrt{Z^T Z}} < tol \quad (8)$$

where $\|\bullet\|_\infty$ is the largest absolute value of the vector, and tol is the Estimability Tolerance.

In general, the number of rows in the basis of the space orthogonal to the rows of X is equal the number of linear dependencies among the columns of X . If there is more than one z , then (8) must be satisfied for each z .

Table 14-6. The basis of the space orthogonal to the rows of X

Elements of β	μ	τ_1	τ_2	τ_3
Distinct rows of X	1	1	0	0
	1	0	1	0
	1	0	0	1
Basis of orthogonal space	-1	1	1	1

Substitution of estimable functions for non-estimable functions

It is not always easy for a user to determine if a linear combination is estimable or not. LinMix tries to be helpful, and if a user attempts to estimate a non-estimable function, LinMix will substitute a “nearby” function that is estimable. Notice is given in the text output when the substitution is made, and it is the user's responsibility to check the coefficients to see if they are reasonable. This section is to describe how the substitution is made.

The most common attempt to estimate a non-estimable function of the parameters is probably trying to estimate a contrast of main effects in a model with interaction, so this will be used as an example. Consider the model:

$$y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \varepsilon_{ijk} \quad (9)$$

where μ is the over-all mean; α_i is the effect of the i -th level of a factor A, $i = 1, 2$; β_j is the effect of the j -th level of a factor B, $j = 1, 2$; $(\alpha\beta)_{ij}$ is the effect of the interaction of the i -th level of A with the j -th level of B; and the ε_{ijk} are independently distributed $N(0, \sigma^2)$. The canonical form of the QR factorization of X is displayed in rows 0 through 8 of [Table 14-7](#). The coefficients to generate $\alpha_1 - \alpha_2$ are shown in row 9. The process is a sequence of regressions followed by residual computations. In the μ column, row 9 is already 0 so no operation is done. Within the α columns, regress row 9 on row 3. (Row 4 is all zeros and is ignored.) The regression coefficient is 1. Now calculate residuals: (row 9) - $1 \times$ (row 3). This operation goes across the entire matrix, *i.e.*, not just the α columns. The result is shown in row 10, and the next operation applies to row 10. The numbers in the β columns of row 10 are zero, so skip to the $\alpha\beta$ columns. Within the $\alpha\beta$ columns, regress row 10 on row 5. (Rows 6 through 8 are all zeros and are ignored.). The regression coefficient is 0, so there is no need to compute residuals. At this point, row 10 is a deviation of row 9 from

an estimable function. Subtract the deviation (row 10) from the original (row 9) to get the estimable function displayed in the last row. The result is the expected value of the difference between the marginal A means. This is likely what the user had in mind.

This section provides a general description of the process just demonstrated on a specific model and linear combination of elements of β . Partition the QR factorization of X vertically and horizontally corresponding to the terms in X . Put the coefficients of a linear combination in a work row. For each vertical partition, from left to right, do the following:

1. Regress (*i.e.*, do a least squares fit) the work row on the rows in the diagonal block of the QR factorization.
2. Calculate the residuals from this fit across the entire matrix. Overwrite the work row with these residuals.
3. Subtract the final values in the work row from the original coefficients to obtain coefficients of an estimable linear combination.

Table 14-7. The QR factorization of X scaled to canonical form

	μ	α_1	α_2	β_1	β_2	$(\alpha\beta)_{11}$	$(\alpha\beta)_{12}$	$(\alpha\beta)_{21}$	$(\alpha\beta)_{22}$
0	1	0.5	0.5	0.5	0.5	0.25	0.25	0.25	0.25
1		1	-1	0	0	0.5	0.5	-0.5	-0.5
2			0	0	0	0	0	0	0
3				1	-1	0.5	-0.5	0.5	-0.5
4					0	0	0	0	0
5						1	-1	-1	1
6							0	0	0
7								0	0
8									0
9	0	1	-1	0	0	0	0	0	0
10	0	0	0	0	0	-0.5	-0.5	0.5	0.5
Final result	0	1	-1	0	0	0.5	0.5	-0.5	-0.5

Fixed effects

The specification of the *fixed effects* (as defined in “[The linear mixed effects model](#)” on page 330) can contain both classification variables and continuous regressors. In addition, the model can include interaction and nested terms.

Output parameterization

Suppose X is $n \times p$, where n is the number of observations and p is the number of columns. If X has rank p , denoted $\text{rank}(X)$, then each parameter can be uniquely estimated. If $\text{rank}(X) < p$, then there are infinitely many solutions. To solve this issue, one must impose additional constraints on the estimator of β .

Suppose column j is in the span of columns $0, 1, \dots, j-1$. Then column j provides no additional information about the response y beyond that of the columns that come before. In this case, it seems sensible to not use the column in the model fitting. When this happens, its parameter estimate is set to 0.

As an example, consider a simple one-way ANOVA model, with three levels. The design matrix is

X_0	X_1	X_2	X_3
1	1	0	0
1	0	1	0
1	0	0	1
1	1	0	0
1	0	1	0
1	0	0	1

X_0 is the intercept column. X_1 , X_2 , and X_3 correspond to the three levels of the treatment effect. X_0 and X_1 are clearly linearly independent, as is X_2 linearly independent of X_0 and X_1 . But X_3 is not linearly independent of X_0 , X_1 , and X_2 , since $X_3 = X_0 - X_1 - X_2$. Hence β_3 would be set to zero. The degrees of freedom for an effect is the number of columns remaining after this process. In this example, treatment effect has 2 degrees of freedom, the number typically assigned in ANOVA.

See “[Predicted values and residuals](#)” on page 368 for computation details.

Singularity tolerance

If intercept is in the model, then center the data. Perform a Gram-Schmidt orthogonalization process. If the norm of the vector after GS divided by the norm of the original vector is less than δ , then the vector is called singular.

Transformation of the response

The transformation pull-down menu provides three options: No transformation, ln transformation (\log_e), or log transformation (\log_{10}). Transformations are applied to the response before model fitting. Note that all estimates using log will be the same estimates found using ln, only scaled by ln 10, since $\log(x) = \ln(x)/\ln(10)$.

If the standard deviation is proportional to the mean, taking the logarithm often stabilizes the variance, resulting in better fit. This implicitly assumes that the error structure is multiplicative with lognormal distribution.

Crossover example continued

The fixed effects are given in equation (5). Using that model, the design matrix X would be expanded into that presented in Table 14-8. The vector of fixed effects corresponding to that design matrix would be

$$\beta = (\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6).$$

Notice that $X_2 = X_0 - X_1$. Hence set $\beta_2 = 0$. Similarly, $X_4 = X_0 - X_3$ and $X_6 = X_0 - X_5$, and hence set $\beta_4 = 0$ and $\beta_6 = 0$.

Table 14-8. Design matrix expanded for fixed effects model
Sequence + Period + Treatment

Intercept	Sequence		Period		Treatment	
X0	AB [X1]	BA [X2]	1 [X3]	2 [X4]	A [X5]	B [X6]
1	1	0	1	0	1	0
1	1	0	1	0	1	0
1	1	0	1	0	1	0
1	1	0	1	0	1	0
1	1	0	1	0	1	0
1	1	0	1	0	1	0

Table 14-8. Design matrix expanded for fixed effects model
Sequence + Period + Treatment (continued)

Intercept	Sequence		Period		Treatment	
X0	AB [X1]	BA [X2]	1 [X3]	2 [X4]	A [X5]	B [X6]
1	0	1	0	1	1	0
1	0	1	0	1	1	0
1	0	1	0	1	1	0
1	0	1	0	1	1	0
1	0	1	0	1	1	0
1	0	1	0	1	1	0
1	0	1	1	0	0	1
1	0	1	1	0	0	1
1	0	1	1	0	0	1
1	0	1	1	0	0	1
1	0	1	1	0	0	1
1	1	0	0	1	0	1
1	1	0	0	1	0	1
1	1	0	0	1	0	1
1	1	0	0	1	0	1
1	1	0	0	1	0	1
1	1	0	0	1	0	1
1	1	0	0	1	0	1

Variance structure

The LinMix Variance Structure page specifies the Z , G , and R matrices defined in “[The linear mixed effects model](#)” on page 330. The user may have 0, 1, or more random effects specified. Only 1 repeated effect may be specified. The Repeated effect specifies the $R = \text{Var}(\epsilon)$. The Random effects specify Z and the corresponding elements of $G = \text{Var}(\gamma)$.

Repeated effect

The repeated effect is used to model a correlation structure on the residuals. Specifying the repeated effect is optional. If no repeated effect is specified, then $R = \sigma^2 I_N$ is used, where N denotes the number of observations, and I_N is the $N \times N$ identity matrix.

All variables used on this page of the LinMix wizard must be classification variables.

To specify a particular repeated effect, one must have a classification model term that uniquely identifies each individual observation. Put the model term in the Repeated Specification box. Note that a model term can be a variable name, an interaction term (*e.g.*, Time*Subject), or a nested term (*e.g.*, Time(Subject)).

The variance blocking model term creates a block diagonal R matrix. Suppose the variance blocking model term has b levels, and further suppose that the data are sorted according to the variance blocking variable. Then R would have the form

$$R = I_b \otimes \Sigma$$

where Σ is the variance structure specified.

The variance Σ is specified using the Type pull-down menu. Several variance structures are possible, including unstructured, autoregressive, heterogeneous compound symmetry, and no-diagonal factor analytic. See the Help file for the details of the variance structures. The autoregressive is a first-order autoregressive model.

To model heterogeneity of variances, use the group variable. Group will accept any model term. The effect is to create additional parameters of the same variance structure. If a group variable has levels $g = 1, 2, \dots, n_g$, then the variance for observations within group g will be Σ_g .

An example will make this easier to understand. Suppose 5 subjects are randomly assigned to each of 3 treatment groups; call the treatment groups T_1 , T_2 , and T_3 . Suppose further that each subject is measured in periods $t = 0, 1, 2, 3$, and that serial correlations among the measurements are likely. Suppose further that this correlation is affected by the treatment itself, so the correlation structure will differ among T_1 , T_2 , and T_3 . Without loss of generality, one may assume that the data are sorted by treatment group, then subject within treatment group.

First consider the effect of the group. It produces a variance structure that looks like

$$R = \begin{bmatrix} R_1 & 0 & 0 \\ 0 & R_2 & 0 \\ 0 & 0 & R_3 \end{bmatrix}$$

where each element of R is a 15×15 block. Each R_g has the same form. Because the variance blocking variable is specified, the form of each R_g is

$$I_5 \otimes \Sigma_g.$$

I_5 is used because there are five subjects within each treatment group. Within each subject, the variance structured specified is

$$\sigma_g^2 \begin{bmatrix} 1 & \rho_g & \rho_g^2 & \rho_g^3 \\ \rho_g & 1 & \rho_g & \rho_g^2 \\ \rho_g^2 & \rho_g & 1 & \rho_g \\ \rho_g^3 & \rho_g^2 & \rho_g & 1 \end{bmatrix}$$

This structure is the autoregressive variance type. Other variance types are also possible. Often compound symmetry will effectively mimic autoregressive in cases where autoregressive models fail to converge.

The output will consist of six parameters: σ^2 and ρ for each of the three treatment groups.

Random effects

Unlike the (single) repeated effect, it is possible to specify up to 10 *random effects*. Additional random effects can be added by clicking the **Add Random** button. To delete a previously specified random effect, click the **Delete Random** button. It is not possible to delete all random effects; however, if no entries are made in the random effect, then it will be ignored.

Model terms entered into the Random Effects Model produce columns in the Z matrix, constructed in the same way that columns are produced in the X matrix. It is possible to put more than one model term in the Random Effects Model, but each random page will correspond to a single variance type. The variance matrix appropriate for that type is sized according to the number of columns produced in that random effect.

The random intercept check box puts an intercept column (*i.e.*, all 1's) into the Z matrix.

The Variance Blocking Variables has the effect of inducing independence among the levels of the Variance Blocking Variables. Mathematically, it expands the Z matrix by taking the Kronecker product of the Variance Blocking Variables with the columns produced by the intercept check box and the Random Effects Model terms.

The Group model term is used to model heterogeneity of the variances among the levels of the Group model term, similarly to Repeated Effects.

Multiple random effects vs. multiple effects on one random effect

Suppose one has two random effect variables, *R1* and *R2*. Suppose further that *R1* has three levels, *A*, *B*, and *C*, while *R2* has two levels, *Y* and *Z*. The hypothetical data are shown as follows.

<i>R1</i>	<i>R2</i>
<i>A</i>	<i>Y</i>
<i>B</i>	<i>Y</i>
<i>C</i>	<i>Y</i>
<i>A</i>	<i>Z</i>
<i>B</i>	<i>Z</i>
<i>C</i>	<i>Z</i>

Specifying a random effect of *R1+R2* will produce different results than specifying *R1* on Random 1 and *R2* on Random 2 pages. This example models the variance using the compound symmetry variance type. To illustrate this, build the resulting *Z* and *G* matrices.

For *R1+R2* on Random 1 page, put the columns of each variable in the *Z* matrix to get the following.

<i>R1</i>			<i>R2</i>	
<i>A</i>	<i>B</i>	<i>C</i>	<i>Y</i>	<i>Z</i>
1	0	0	1	0
0	1	0	1	0
0	0	1	1	0
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1

The random effects can be placed in a vector $\gamma = (\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5)$. *R1* and *R2* share the same variance matrix *G* where:

$$G_1 = \begin{bmatrix} \sigma_1^2 & \sigma_2^2 & \sigma_3^2 \\ \sigma_2^2 & \sigma_1^2 & \sigma_2^2 \\ \sigma_3^2 & \sigma_2^2 & \sigma_1^2 \end{bmatrix}$$

Now, consider the case where $R1$ is placed on Random 1 page, and $R2$ is placed on Random 2 page. For $R1$, the Z matrix columns are:

$R1$		
A	B	C
1	0	0
0	1	0
0	0	1
1	0	0

$R1$		
A	B	C
1	0	0
0	1	0
0	0	1
1	0	0
0	1	0
0	0	1

$$G_1 = \begin{bmatrix} \sigma_1^2 & \sigma_2^2 & \sigma_3^2 \\ \sigma_2^2 & \sigma_1^2 & \sigma_2^2 \\ \sigma_3^2 & \sigma_2^2 & \sigma_1^2 \end{bmatrix}$$

For $R2$, the columns in the Z matrix and the corresponding G is:

$R2$	
Y	Z
1	0
1	0
1	0
0	1
0	1
0	1

$$G_2 = \begin{bmatrix} \sigma_4^2 & \sigma_5^2 \\ \sigma_5^2 & \sigma_4^2 \end{bmatrix}$$

In this case:

$$G = \begin{bmatrix} G_1 & \mathbf{0} \\ \mathbf{0} & G_2 \end{bmatrix}$$

Least squares means

Sometimes one would like to see the mean response for each level of a classification variable or for each combination of levels for two or more classification variables. If there are covariables with unequal means for the different levels, or if there are unbalanced data, the subsample means are not estimates that can be validly compared. This section describes least squares means, statistics that make proper adjustments for unequal covariable means and unbalanced data in the linear mixed effects model.

Consider a completely randomized design with a covariable. The model is:

$$y_{ij} = \alpha_i + x_{ij} \beta + \epsilon_{ij}$$

where y_{ij} is the observed response on the j^{th} individual on the i^{th} treatment; α_i is the intercept for the i^{th} treatment; x_{ij} is the value of the covariable for the j^{th} individual in the i^{th} treatment; β is the slope with respect to x ; and ϵ_{ij} is a random error with zero expected value. Suppose there are two treatments; the average of the x_{1j} is 5; and the average of the x_{2j} is 15. The respective expected values of the sample means are $\alpha_1 + 5\beta$ and $\alpha_2 + 15\beta$. These are not comparable because of the different coefficients of β . Instead, one can estimate $\alpha_1 + 10\beta$ and $\alpha_2 + 10\beta$ where the overall mean of the covariable is used in each linear combination.

Now consider a 2×2 factorial design. The model is:

$$y_{ijk} = \mu + \alpha_i + \beta_j + \epsilon_{ijk}$$

where y_{ijk} is the observed response on the k^{th} individual on the i^{th} level of factor A and the j^{th} level of factor B; μ is the over-all mean; α_i is the effect of the i^{th} level of factor A; β_j is the effect of the j^{th} level of factor B; and ϵ_{ijk} is a random error with zero expected value. Suppose there are six observations for the combinations where $i = j$ and four observations for the combinations where $i \neq j$. The respective expected values of the averages of all values on level 1 of A and the averages of all values on level 2 of A are $\mu + (0.6\beta_1 + 0.4\beta_2) + \alpha_1$ and $\mu + (0.4\beta_1 + 0.6\beta_2) + \alpha_2$. Thus, sample means cannot be used to compare levels of A because they contain different functions of β_1 and β_2 . Instead, one compares the linear combinations $\mu + (\beta_1 + \beta_2)/2 + \alpha_1$ and $\mu + (\beta_1 + \beta_2)/2 + \alpha_2$.

The preceding examples constructed linear combinations of parameters, in the presence of unbalanced data, that represent the expected values of sample means in balanced data. This is the idea behind least squares means. Least squares means are given in the context of a defining term, though the process can be repeated for different defining terms for the same model. The defining

term must contain only classification variables and it must be one of the terms in the model. Treatment is the defining term in the first example, and factor A is the defining term in the second example. When the user requests least squares means, LinMix automatically generates the coefficients l_j of expression (5) and processes them almost as it would process the coefficients specified in an estimate statement. This chapter describes generation of linear combinations of elements of β that represent least squares means. A set of coefficients are created for each of all combinations of levels of the classification variables in the defining term. For all variables in the model, but not in the defining term, average values of the variables are the coefficients. The average value of a numeric variable (covariable) is the average for all cases used in the model fitting. For a classification variable with k levels, assume the average of each indicator variable is $1/k$. The value $1/k$ would be the actual average if the data were balanced. The values of all variables in the model have now been defined. If some terms in the model are products, the products are formed using the same rules used for constructing rows of the X matrix as described in the section on Fixed Effects, above. It is possible that some least squares means will not be estimable.

For example, suppose the fixed portion of the model is:

$$\text{Drug} + \text{Form} + \text{Age} + \text{Drug} * \text{Form}.$$

To get means for each level of Drug, the defining term is Drug. Since Drug has three levels, three sets of coefficients are created. Build the coefficients associated the first level of Drug, DrugA. The first coefficient is 1 for the implied intercept. The next three coefficients are 1, 0, and 0, the indicator variables associated with DrugA. Form is not in the defining term, so average values are used. The next four coefficients are all 0.25, the average of a four factor indicator variable with balanced data. The next coefficient is 32.17, the average of Age. The next twelve elements are:

$$[1 \ 0 \ 0] \otimes [0.25 \ 0.25 \ 0.25 \ 0.25] = [0.25 \ 0.25 \ 0.25 \ 0.25 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]$$

The final result is shown in the DrugA column of Table 14-9. The results for DrugB and DrugC are also shown in Table 14-9. No new principles would be illustrated by finding the coefficients for the Form least squares means. The coefficients for the Drug*Form least squares means would be like representative rows of X except that Age would be replaced by average of Age.

Table 14-9. Coefficients of least squares means

Column of X		β_j	I_j		
			DrugA	DrugB	DrugC
Intercept		β_0	1	1	1
Drug	DrugA	β_1	1	0	0
	DrugB	β_2	0	1	0
	DrugC	β_3	0	0	1
Form	capsule	β_4	0.25	0.25	0.25
	drops	β_5	0.25	0.25	0.25
	syrup	β_6	0.25	0.25	0.25
	tablet	β_7	0.25	0.25	0.25
Age		β_8	32.17	32.17	32.17
Drug*Form	DrugA*capsule	β_9	0.25	0	0
	DrugA*drops	β_{10}	0.25	0	0
	DrugA*syrup	β_{11}	0.25	0	0
	DrugA*tablet	β_{12}	0.25	0	0
	DrugB*capsule	β_{13}	0	0.25	0
	DrugB*drops	β_{14}	0	0.25	0
	DrugB*syrup	β_{15}	0	0.25	0
	DrugB*tablet	β_{16}	0	0.25	0
	DrugC*capsule	β_{17}	0	0	0.25
	DrugC*drops	β_{18}	0	0	0.25
	DrugC*syrup	β_{19}	0	0	0.25
	DrugC*tablet	β_{20}	0	0	0.25

Contrasts

A contrast is a linear combination of the parameters associated with a term where the coefficients sum to 0. Contrasts facilitate the testing of linear com-

binations of parameters. For example, consider a completely randomized design with four treatment groups: Placebo, Low Dose, Medium Dose, and High Dose. The user may wish to test the hypothesis that the average of the dosed groups is the same as the average of the placebo group. One could write this hypothesis as:

$$H_0: \mu_{\text{Placebo}} - (\mu_{\text{Low}} + \mu_{\text{Medium}} + \mu_{\text{High}})/3 = 0$$

or equivalently:

$$H_0: 3\mu_{\text{Placebo}} - (\mu_{\text{Low}} + \mu_{\text{Medium}} + \mu_{\text{High}}) = 0.$$

Both of these are contrasts since the sum of the coefficients is 0. Note that both contrasts make the same statement. This is true generally: contrasts are invariant under changes in scaling.

Contrasts are tested using the Wald (1943) test statistic:

$$(\hat{L}\hat{\beta})^T (L(X^T \hat{V} X)^{-1} L^T)^{-1} (\hat{L}\hat{\beta})$$

where $\hat{\beta}$ and \hat{V} are estimators of β and V . L must be a matrix such that each row is in the row space of X . This requirement is enforced in the wizard. The Wald statistic is compared to an F distribution with $\text{rank}(L)$ numerator degrees of freedom and denominator degrees of freedom estimated by the program or supplied by the user.

Joint versus single contrasts

Joint contrasts are constructed when the number of columns for contrast is changed from the default value of 1 to a number larger than 1. Note that this number must be no more than one less than the number of distinct levels for the model term of interest. This tests both hypotheses jointly, *i.e.*, in a single test with a predetermined level of the test.

A second approach would be to put the same term in the neighboring contrast, both of which have one column. This approach will produce two independent tests, each of which is at the specified level.

To see the difference, compare Placebo to Low Dose and Placebo to Medium Dose. Using the first approach, enter the coefficients as follows.

	Contrast #1		
Title	Joint tests		

Effect	Treatment		
Contrast	Treatment		
#1	Placebo	-1	-1
#2	Low Dose	1	0
#3	Medium Dose	0	1
#4	High Dose	0	0

This will produce one test, testing the following hypothesis.

$$H_0: \begin{bmatrix} -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \end{bmatrix} \beta = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Now compare this with putting the second contrast in its own contrast.

	Contrast #1		Contrast #2	
Title	Low vs. Placebo		Medium vs. Placebo	
Effect	Treatment		Treatment	
Contrast	Treatment		Treatment	
#1	Placebo	-1	Placebo	-1
#2	Low Dose	1	Low Dose	0
#3	Medium Dose	0	Medium Dose	1
#4	High Dose	0	High Dose	0

This produces two tests.

$$H_{01}: [-1 \ 1 \ 0 \ 0] \beta = [0] \quad H_{02}: [-1 \ 0 \ 1 \ 0] \beta = [0]$$

Note that this method inflates the overall Type I error rate to approximately 2α , whereas the joint test maintains the overall Type I error rate to α , at the possible expense of some power.

Nonestimability of contrasts

If a contrast is not estimable, then an estimable function will be substituted for the nonestimable function. See [“Substitution of estimable functions for non-estimable functions” on page 338](#) for the details.

Degrees of freedom

The degrees of freedom check box allows the user to control the denominator degrees of freedom in the F approximation. The numerator degrees of freedom is always calculated as $\text{rank}(L)$. If the check box is not checked, then the default denominator degrees of freedom will be used, i.e., the Satterthwaite degrees of freedom. Check the box for residual degrees of freedom.

Note: For purely fixed effects model, Satterthwaite and residual degrees of freedom yield the same number. For details of the Satterthwaite approximation, see the section “Satterthwaite’s Approximation for Degrees of Freedom”.

To override the default degrees of freedom, check the box, and enter an appropriate number. The degrees of freedom must be greater than 0.

Other options

To control the estimability tolerance, enter an appropriate tolerance in the Estimability Tolerance text box (see [“Substitution of estimable functions for non-estimable functions”](#) on page 338).

The Show Coefficients check box will produce extra output, including the coefficients used to construct the tolerance. If the contrast entered is estimable, then it will repeat the contrasts entered. If the contrasts are not estimable, it will enter the estimable function used instead of the nonestimable function.

Estimates

The estimates page produces estimates output instead of contrasts. As such, the coefficients need not sum to zero. Additionally, multiple model terms may be included in a single estimate.

Unlike contrasts, estimates do not support joint estimates. The confidence intervals are marginal. The confidence level must be between 0 and 100. Note that it is entered as a percent: Entering 0.95 will yield a very narrow confidence interval with coverage just less than 1%.

All other options act similarly to the corresponding option in the Contrasts.

LinMix Wizard

The general steps to create a linear mixed effects model are outlined below. Some steps are optional, depending which calculations are included. Refer to [“An example” on page 327](#) for an in-depth discussion of linear mixed effects modeling in the context of an example study. See the *Examples Guide* for further linear mixed effects modeling and bioequivalence examples.

Note: To reuse the model last run in LinMix (during the same WinNonlin session), click the **Previous Model** button. To reload a saved model, click the **Load...** button and open the LinMix Command file (*.LML). ANOVA command files (*.ANV) can also be loaded into LinMix so long as bioequivalence is not included.

To set up a linear mixed effects model:

1. Open the worksheet to be analyzed.
2. Choose **Tools>Linear Mixed Effects Wizard** from the WinNonlin menus or click the **Linear Mixed Effects Wizard** tool bar button.
3. Define the model in the Fixed Effects dialog, as described under [“Fixed effects”](#) below.
4. Set up any other optional calculations and options as described under:
 - [“Variance structure” on page 357](#)
 - [“Contrasts” on page 360](#)
 - [“Estimates” on page 362](#)
 - [“Least squares means” on page 363](#)
 - [“LinMix general options” on page 364](#)
5. (Optional) At any point in the LinMix wizard, save the model to a LinMix Command file (*.LML) for future re-use, by clicking the **Save Model** button.
6. Click **Calculate** to start the calculations and close the LinMix wizard.

LinMix limits and constraints

Cell data may not include question marks (?) or either single (') or double (") quotation marks.

Variable names must begin with a letter. After the first character, valid characters include numbers and underscore (my_file_2). They cannot include spaces or the following operational symbols: +, -, *, /, =. They also cannot include parentheses (), question marks (?) or semicolons (;). Finally, variable names may not have single or double quotes in them.

Titles (in Contrasts, Estimates, or ASCII) can have single or double quotation marks—but not both.

The following are maximum counts for LinMix variables and other objects.

Table 14-10. LinMix limits

Item	Maximum
model terms	30
factors in a model term	10
sort keys	16
dependent variables	128
covariate/regressor variables	255
variables in the data set	256
random and repeated statements	10
variables	256
contrast statements	100
estimate statements	100
combined length of all variance parameter names (total characters)	10,000
combined length of all level names (total characters)	10,000
levels per variable	1,000
path and file name (total characters)	256
number of records passed from WinNonlin to other programs	65,535
characters per data cell	128
column name length (characters)	128
titles (e.g., output title or chart title)	5 lines

Fixed effects

To define the model:

1. Drag and drop variables to categorize those needed in the LinMix model:

- *Dependent variables* contain the values to which the model will be fit, e.g., drug concentration.
- *Classification variables* or factors are categorical independent variables, e.g., formulation, treatment, and gender.
- *Regressor variables* or *covariates* are continuous independent variables, e.g., temperature or body weight.
- *Sort variables* define subsets of data to be processed separately, that is, a separate analysis will be done for each level of the *sort variables*. If the sort variable has missing values, the analysis will be performed for the missing level and MISSING will be printed as the current value of the sort variable.
- A *weight variable* can be included if weights for each record are included in a separate data column. The weights are used to compensate for observations having different variances. When a weight variable is specified, each row of data is multiplied by the square root of the corresponding weight. Weight variable values should be proportional to the reciprocals of the variances. In the most common situation, the data are averages and weights are sample sizes associated with the averages. See also “[Weighting](#)” on page 243.

The Weight Variable cannot be a classification variable. It must have been declared as a regressor/covariate before it can be used as a weight variable. It can also be used in the model.

2. In the box labeled Model Specification, enter the fixed effects model:

- type the model, or
- drag variable names and click on the operators to be used in the model.

For discussion of the model, using an example to illustrate, see “[The linear mixed effects model](#)” on page 330.

Note: Parentheses in the model definition represent nesting of model terms. Hence:

Seq+Subject (Seq) +Period+Form

is a valid use of parentheses and indicates that Subject is nested within Seq.

Drug + Disease + (Drug*Disease)

is not a valid use of parentheses in the model definition.

3. Accept the default setting for **Fixed Effects Confidence Level** or enter the confidence interval for calculations of the fixed effects model.
4. Check the **Dependent Variables Transformation** check box if the dependent variable values should be log-transformed before analysis. Select the transformation from the pull-down list for **Dependent Variables Transformation**.

CAUTION: If the dependent variable data in the workbook are already log transformed, do not select a log transformation from the pull-down list, as this will transform the data a second time.

5. Select the **No Intercept** option to eliminate the intercept in the model. Note that this will change the parameterization of the classification variables. See “Construction of the X matrix” on page 332 for discussion.
6. At any time in the LinMix wizard, it is possible to save the model to a LinMix Command file (*.LML) for future re-use, by clicking the **Save Model** button.
7. Click **Next** to move to the **Variance structure** or **Calculate** to run the analysis.

Variance structure

The Variance Structure dialog follows the **Fixed effects** dialog in the LinMix wizard. (See “LinMix Wizard” on page 354.) It provides a variety of covariance structures. The variances of the random-effects parameters become the covariance parameters for this model. Mixed linear models contain both *fixed effects* and *random effect* parameters. See “Variance structure” on page 342 for discussion of variance in linear mixed effects models.

The choices for variance structures are shown in Table 14-11. The table uses the following notation:

n = number of groups, (n = 1 if the group option is not used)

t = number of time points in the repeated context

b = dimension parameter: for some variance structures, the num-

ber of bands; for others, the number of factors

$1(\text{expression}) = 1$ if expression is true, 0 otherwise

Table 14-11. Variance structure types

Description	Number of Parameters	$i^{\text{th}}, j^{\text{th}}$ element
Variance Components	n	$\sigma_k^2 1(i = j)$ and i corresponds to k^{th} effect
Unstructured	$n \times t \times (t + 1)/2$	σ_{ij}
Banded Unstructured (b)	$n \times b/2(2t - b + 1)$ if $b < t$; otherwise, same as Unstructured.	$\sigma_{ij}(i - j < n)$
Toeplitz	$n \times t$	$\sigma_{ i - j + 1}$
Banded Toeplitz	$n \times b$ if $b < t$; otherwise, same as Toeplitz	$\sigma_{ i - j + 1} 1(i - j < n)$
Compound Symmetry	$n \times 2$	$\sigma_1^2 + \sigma^2 1(i = j)$
Heterogeneous Compound Symmetry	$n \times (t + 1)$	$\sigma_i \sigma_j [\rho 1(i \neq j) + 1(i = j)]$
No-Diagonal Factor Analytic	$n \times t(t + 1)/2$	$\sum_{k=1}^{\min(i, j, n)} \lambda_{ik} \lambda_{jk}$
Banded No-Diagonal Factor Analytic (b)	$n \times b/2(2t - b + 1)$ if $b < t$; otherwise, same as No-Diagonal Factor Analytic	
Autoregressive (1)	$n \times 2$	$\sigma^2 \rho^{ i - j }$
Heterogeneous Autoregressive (1)	$n \times (t + 1)$	$\sigma_i \sigma_j \rho^{ i - j }$

See “[Random coefficients](#)” below to specify traditional variance components and/or to specify random coefficients; or “[Repeated measurements](#)” on [page 360](#) to set covariance structures for repeated measurements on subjects.

Random coefficients

The Random tab of the [Variance structure](#) dialog provides a means to incorporate random effects in a model. Other random models (tabs) can be added by clicking the Add Random button. It can be used to specify traditional variance components and/or to specify random coefficients. The random effects can be either classification or continuous. The Repeated tab is used to specify covariance structures for repeated measurements on subjects.

To specify random effects:

1. *Random Effects Model:* Enter the model by typing it or by dragging the variable names and operators to be used in the model from the appropriate boxes to the Random Effects Model box using the mouse. Use parentheses to indicate nesting. The model can be built from classification variables and/or regressors and the operators at the top of the dialog.

Note: The same variable cannot be used in both the fixed-effects specification and the random effects specification unless it is used differently—as part of a product, for instance. The same term (single variables, products, or nested variables) should not appear in both.

2. *Variance Blocking Variables (Subject):* (Optional) Drag the appropriate variable to this field. The variable must be a classification model term. This Subject field identifies the subjects in a data set. Complete independence is assumed among subjects. Thus, Subject produces a block diagonal structure with identical blocks.
3. *Group:* (Optional) This variable defines an effect specifying heterogeneity in the covariance structure. All observations having the same level of the group effect have the same covariance parameters. Each new level of the group effect produces a new set of covariance parameters with the same structure as the original group. This variable must be a classification model term.
4. *Type:* Select the type of covariance structure. The options are explained in [Table 14-11, “Variance structure types,”](#) on page 358.
5. *Random Intercept:* Check this box to indicate that the intercept is random. This is most commonly employed when a subject is specified in the Variance Blocking field. The default is off-no random intercept.

Repeated measurements

The Repeated tab of the [Variance structure](#) dialog provides the means to specify the R matrix in the mixed model. If no Repeated statement is specified, R is assumed to be equal to $\sigma^2 I$. The repeated effect must contain only classification variables.

To specify the repeated variance structure:

1. *Repeated Specification*: enter the model by typing or by dragging the classification variable names and operators from the appropriate boxes to this field. Syntax rules for the repeated specification are the same as those for the fixed-effects model.
2. *Variance Blocking Variables (Subject)*: (Optional) This must be a classification variable. This Subject field identifies the subjects in a data set. Complete independence is assumed among subjects; Subject produces a block diagonal structure with identical blocks.
3. *Group*: (Optional) This must be a classification variable with no regressors or operators. It defines an effect specifying heterogeneity in the covariance structure. All observations having the same level of the group effect have the same covariance parameters. Each new level of the group effect produces a new set of covariance parameters with the same structure as the original group.
4. *Type*: Select the type of covariance structure from the options explained in [Table 14-11, “Variance structure types,”](#) on page 358.
5. Click **Next** to proceed to the [Contrasts](#) or **Calculate** to run the analysis.

Contrasts

The Contrasts dialog is next in the LinMix wizard after the [Variance structure](#). Contrasts provide a mechanism for creating custom hypothesis tests. Contrasts can be computed only for classification variables. (See [“Fixed effects” on page 356](#) to select classification variables.) Further discussion of contrasts is provided in [“Contrasts” on page 350](#).

Note: Interactions can be used as contrasts only if the interaction is a model term; the same is true for nested terms.

To specify contrasts:

1. Enter a descriptive label for the contrast in the Title cell.
2. Drag the appropriate Model term to the cell labeled Effect.
3. Enter the appropriate contrast coefficients in the empty cells associated with Contrasts.

Note: The coefficients for single contrasts must sum to zero.

Once contrast #1 has been specified, the settings for the contrast can be specified. They are specific to that contrast.

Once Contrast #1 is entered, the grid automatically expands to allow for Contrast #2. This grid will allow additional contrasts, up to a maximum of 100.

Note: If all values of the dependent variable are missing or excluded for a given level of the effect variable, then this level of the effect variable will not appear in the Contrast vector.

Effect variables may be either numeric or alphabetic, but they may not be both. If a column contains both numeric and alphabetic data, the column cannot be used as an effect variable when building contrasts.

4. Specify the Settings for the contrast entered. These settings are specific to the active contrast.
 - a. **Number of columns for contrast:** check this box to test multiple linearly independent combinations simultaneously. These columns are tested jointly.
 - b. If the algorithm doesn't seem to be using appropriate choices for the degrees of freedom, check **User-specified** [denominator] **degrees of freedom** and enter an integer for df greater than 1.
 - c. The **Show Coefficients of Contrasts** check box ensures that the contrast is estimable. This shows the actual coefficients used in the output. If the contrast is not estimable, a nearby estimable function will be used instead.
 - d. The **Estimability tolerance** indicates the closeness of the zero vector before invoking the algorithm to make it estimable.

Each contrast is tested independently of other contrasts.

5. Click **Next** to proceed to the [Estimates](#), or **Calculate** to run the analysis.

Estimates

The LinMix Estimates dialog is similar to the [Contrasts](#) dialog—except that multiple effects can be used in each estimate statement. Interaction terms and nested terms can be used if they appear in the model. If the fixed-effects model includes an intercept (the default), then the term Intercept is included with the model terms. If the Intercept term is used as an effect for the estimate, it works like a regressor; only one number will be entered in the grid.

Note: The marginal confidence interval is generated for each estimate.

See also “[Estimates](#)” on page 353 and “[The linear mixed effects model](#)” on page 330.

To enter estimates:

1. Enter a descriptive label for the estimate in the cell labeled Title.
2. Drag the appropriate Model term to the cell labeled Effect. The grid expands to display every unique level of that term.
3. Enter the appropriate coefficients in the empty cells associated with Effects.

Note: Coefficients for estimates are not bound by the same limits as are those for [Contrasts](#).

4. (Optional) To create compound estimates, drag another model term below the existing estimate. The grid expands to include the new term and its levels.
5. Enter the appropriate coefficients in the cells associated with the new term.
6. Repeat as necessary for other terms within any one estimate.
7. Repeat as necessary for other estimates. The grid automatically expands up to the maximum number of estimates.
8. Specify the Settings for the Estimate:
 - a. Set the Confidence Level for confidence interval calculations.

- b. If the algorithm doesn't seem to be using appropriate choices for the degrees of freedom, select the **User-specified** [denominator] **degrees of freedom** check box and enter an integer for *df* greater than 1.
 - c. Checking **Show Coefficients of Estimates** displays in the output the actual coefficients used and, if the estimate is not estimable, applies a nearby, estimable function.
 - d. The **Estimability tolerance** indicates the closeness of the zero vector before invoking the algorithm to make it estimable.
9. Click **Next** to proceed to the **Least squares means**, or **Calculate** to run the LinMix model.

Least squares means

Least squares means are generalized least-squares means of the fixed effects. They are estimates of what the mean values would have been had the data been balanced. That is, these are the means predicted by the ANOVA model. If a data set is balanced, the least squares means will be identical to the raw (observed) means. See also “[Least squares means](#)” on page 348.

Least Squares Means can be computed for any classification model term. See “[Fixed effects](#)” on page 340 to set classification variables.

To specify Least Squares Means:

1. The Least Squares Means dialog follows the **Estimates** dialog in the LinMix wizard. (See “[LinMix Wizard](#)” on page 354.)
2. The **linear mixed effects model** terms are provided in the Model Classifiable Terms field. Drag each model term for which the least square means are to be calculated up to the Least Squares Means box.
3. (Optional) Select the Settings for the Least Squares Means. The settings selected are applied to all terms.
 - a. Set the Confidence Level for confidence interval calculations.
 - b. If the algorithm doesn't seem to be using appropriate choices for the degrees of freedom, check **User-specified** [denominator] **degrees of freedom** and enter an integer for *df* greater than 1.
 - c. Checking **Show Coefficients of LSM's** ensures that the contrast is estimable. If the hypothesis is estimable, no change is made. If the test is not estimable, this option will apply a nearby, estimable function.

- d. The **Estimability tolerance** indicates the closeness of the zero vector before invoking the algorithm to make it estimable.
 - e. The **Compute Pairwise differences of LSM's** check box provides additional output with the differences of confidence intervals and LSM's. This function tests $\eta_1 = \eta_2$.
4. Click **Next** to proceed to the **LinMix general options** or **Calculate** to run the LinMix analysis.

LinMix general options

To specify LinMix general options:

1. Check **ASCII output** to include text output in addition to workbook output. See "**LinMix output**" on page 365.
2. *Title*: Enter a title for the ASCII output (optional). The title can include up to 5 lines. When entering a multi-line title, press ENTER to move to the next line.
3. *Degrees of freedom calculation form*: (Optional) **Residual** is the same as in a purely fixed effects model. **Satterthwaite** computes the df base on χ^2 approximation to distribution of variance.
4. *maximum iterations*: (Optional) Enter the number of maximum iterations. This is the number of iterations in the Newton fitting algorithm.
5. (Optional) If the model has random effects, select the **Initial Variance Parameters** button to edit a grid of the parameters. If these are not specified, the application uses the method of moments estimates. Click **OK**.
6. *Singularity Tolerance*: (Optional) Enter the level. The columns in X and Z are eliminated if their norm \leq this number.
7. *Convergence Criterion*: (Optional) Specify the desired option, discussed under "**Convergence criterion**" on page 373.
8. (Optional) Select **Intermediate Calculations** to include the design matrix, reduced data matrix, asymptotic covariance matrix of variance parameters, Hessian, and final variance matrix in the ASCII output.
9. To save the model to a LinMix command file (*.LML) for later re-use, use the **Save Model** button.

10. Click **Calculate** to run the LinMix analysis, or **Back** to return to the [Least squares means](#) and other previous settings.

For details on ASCII and workbook output, see “[LinMix output](#)” on page 365.

LinMix output

LinMix produces the following text and/or workbook output. Specific output depends on the options set in the LinMix Wizard (see “[LinMix Wizard](#)” on page 354). See also “[Computational details](#)” on page 370.

Linear mixed effects text

This is the optional ASCII output selected in the [LinMix general options](#). It contains a complete summary of the analysis, including all output as well as analysis settings and any errors that occur.

Linear mixed effects workbook

The Linear Mixed Effects Workbook contains summary tables of the output. Depending which model options have been set up, workbook may contain the items listed in [Table 14-12](#). Details for specific output follow below.

Table 14-12. Linear mixed effects worksheets

Item	Contents
Diagnostics	Model-fitting information
Sequential Tests	Tests of fixed effect for significant improvement in fit, in sequence. Depends upon the order that model terms are written.
Partial Tests	Tests of fixed effects, but each term is tested last
Final Fixed Parameters	Estimates, standard error, t-statistic, p-value, and confidence intervals
Final Variance Parameters	Estimates and units of the final variance parameters
Parameter Key	Information about the variance structure
Contrasts	Hypothesis, df, F-statistic, and p-value
Estimates	Estimate, standard error, t-statistic, pvalue, and confidence interval

Table 14-12. Linear mixed effects worksheets (continued)

Item	Contents
Least Squares Means	Least squares means, standard error, df, t-statistic, p-value, and confidence interval
Residuals	Dependent variable(s), units, observed and predicted values, standard error and residuals
User Settings	Information about model settings
Initial Variance Parameters	Initial values for variance parameters
Iteration History	Data on the variable estimation in each iteration

Tests of hypotheses

The tests of hypotheses are the mixed model analog to the analysis of variance in the fixed model case. The tests are performed by finding appropriate L matrices and testing the hypotheses $H_0: L\beta = 0$ for each model term.

Sequential tests

The sequential tests test each model term sequentially. First it is tested whether the first model term should enter the model. Then it is tested whether the second model term should enter the model, given that the first term is in the model. Then it is tested whether the third model term should enter, given that the first two terms, etc., until all model terms are exhausted.

This is computed using a QR factorization of the XY matrix. The QR factorization is segmented to match the number of columns that each model term contributes to the X matrix.

Partial tests

Partial tests test each model term given every other model term. Unlike sequential tests, partial tests are invariant under the order in which model terms are listed in the Fixed Effects dialog. It does this by factoring out of each model term the contribution attributable to the remaining model terms.

This is computed by modifying the basis created by the QR factorization to yield a basis that more closely resembles that found in balanced data.

Discussion

For fixed effects models, certain properties can be stated for the two types of ANOVA. For the sequential ANOVA, the sums of squares are statistically independent. Also, the sum of the individual sums of squares is equal to the model sum of squares; *i.e.*, the ANOVA represents a partitioning of the model sum of squares. However, some terms in ANOVA may be contaminated with undesired effects. The partial ANOVA is designed to eliminate the contamination problem, but the sums of squares are correlated and do not add up to the model sums of squares. The mixed effects tests have similar properties.

Akaike's Information Criterion (AIC)

The Linear Mixed Effects module uses the smaller-is-better form of Akaike's Information Criterion:

$$AIC = -2L_R + 2s$$

where L_R is the restricted log-likelihood function evaluated at final fixed parameter estimates β and the final variance parameter estimates θ , and s is the rank of the fixed effects design matrix X plus the number of parameters in θ (*i.e.*, $s = \text{rank}(X) + \text{dim}(\theta)$).

Schwarz's Bayesian Criterion (SBC)

The Linear Mixed Effects module uses the smaller-is-better form of Schwarz's Bayesian Criterion:

$$SBC = -2L_R + s \log(N - r)$$

where L_R is the restricted log-likelihood function evaluated at the final estimates β and θ ,

N is the number of observations used,

r is the rank of the fixed effects design matrix X , and

s is the rank of the fixed effects design matrix X plus the number of parameters in θ

(*i.e.*, $s = \text{rank}(X) + \text{dim}(\theta)$).

Hessian eigenvalues

The eigenvalues are formed from the Hessian of the restricted log likelihood. There is one eigenvalue per variance parameter. Positive eigenvalues indicate that the final parameter estimates are found at a maximum.

Predicted values and residuals

In the Linear Mixed Effects module, predicted values for the dependent variables are computed at all of the input data points. To obtain predicted values for data points other than the observed values, the user should supply the desired points in the input data set with missing dependent variables. Predicted values are estimated by using the rows of the fixed effects design matrix X as the L matrices for estimation, and therefore the predicted values for the input data are equal to $\hat{y} = X\beta$, where β are the final fixed parameters estimates. Note that if the user requested a log-transform, then the predicted values are also transformed. The residuals are then the observed values minus the predicted values, or in the case of transforms, the residuals are the transformed values minus the predicted values. The variance of the prediction is as follows.

$$Var(\hat{y}) = X(X^T \hat{V} X)^{-1} X^T$$

The standard errors are the square root of the diagonal of the matrix above. T -type confidence intervals for the predicted values are also computed.

The Lower_CI and Upper_CI are the confidence bounds on the predicted response.

$$\hat{y}_1$$

The confidence level is given in Conf_Level. This is controlled on the Fixed Effects page of the wizard.

Fixed effects parameter estimates

In the Linear Mixed Effects module, the final fixed effects parameter estimates β are computed by solving

$$T_{00} \hat{\beta} = y_0$$

where T_{00} and y_0 are elements of the reduced data matrix as defined in the section on Restricted Maximum Likelihood Estimation, and β are the final fixed parameters estimates.

When there are multiple solutions, a convention is followed such that if any column of T_{00} is a linear combination of preceding columns, then the corresponding element of β is set to zero.

Coefficients

If Display Coefficients is checked on any of the page of the LinMix wizard, the coefficients of the parameters used to construct the linear combination will be displayed. If the linear combination is estimable, then the coefficients will be the same as those entered. If the linear combination is not estimable, then the estimable replacement will be displayed. This provides the opportunity to inspect what combination is actually used to see if it represents the intentions.

Iteration history

This worksheet holds the fitting algorithm results. If the initial estimates are the REML estimates, then zero iterations will be displayed.

Parameter key

If the model included random or repeated effects, the Parameter column of this grid contains the variance parameter names assigned by LinMix. If the Group option is not used in the model or if there is only one random or repeated model, then indices are not used in the parameter names.

Otherwise, two indices are appended to the variance parameter names in order to clarify which random or repeated model, or group, the variance parameter addresses. The first index indicates the random model or repeated model to which the parameter applies. An index of 1 indicates the model on the Random 1 tab in the user interface, etc. The highest index indicates the repeated model if there was one. The second index is the group index indicating that the parameters with the same group index are for the same group level.

Source	Random if the parameter is from a random model Repeated if the parameter is from a repeated specification Assumed if the variance parameter is the residual
Type	Variance type specified by the user, or Identity, if the residual variance

Bands	Number of bands or factors if appropriate for the variance type
Subject_Term	Subject term for this variance structure, if there was a subject
Group_Term	Group term for this variance structure, if there was a group
Group_Level	Level of the group that this parameter goes with, if there was a group

Computational details

Restricted maximum likelihood

For linear mixed effects models that contain random effects, the first step in model analysis is determining the maximum likelihood estimates of the variance parameters associated with the random effects. This is accomplished in LinMix using Restricted Maximum Likelihood (REML) estimation.

The linear mixed effects model described under “[The linear mixed effects model](#)” on page 330 is:

$$y = X\beta + Z\gamma + \varepsilon,$$

$$V = \text{Variance}(y) = ZGZ^T + R.$$

Let θ be a vector consisting of the variance parameters in G and R . The full maximum likelihood procedure (ML) would simultaneously estimate both the fixed effects parameters β and the variance parameters θ by maximizing the likelihood of the observations y with respect to these parameters. In contrast, restricted maximum likelihood estimation (REML) maximizes a likelihood that is only a function of the variance parameters θ and the observations y , and not a function of the fixed effects parameters. Hence for models that do not contain any fixed effects, REML would be the same as ML.

To obtain the restricted likelihood function in a form that is computationally efficient to solve, the data matrix $[X | Z | y]$ is reduced by a QR factorization to an upper triangular matrix. Let $[R_0 | R1 | R2]$ be an orthogonal matrix such that the multiplication with $[X | Z | y]$ results in an upper triangular matrix:

$$\begin{bmatrix} R_0^T \\ R_1^T \\ R_2^T \end{bmatrix} [X | Z | y] = \begin{bmatrix} T_{00} & T_{01} & y_0 \\ 0 & T_{11} & y_r \\ 0 & 0 & y_e \end{bmatrix}$$

REML estimation maximizes the likelihood of θ based on y_r and y_e of the reduced data matrix, and ignores y_0 . Since y_e has a distribution that depends only on the residual variance, and since y_r and y_e are independent, the negative of the log of the restricted likelihood function is the sum of the separate negative log-likelihood functions:

$$\log(L(\theta; y_r, y_e)) = \log(L(\theta; y_r)) + \log(L(\theta; y_e)) - (N - \text{rank}(X))/2 \log(2\pi)$$

where:

$$\log(L(\theta; y_r)) = -\frac{1}{2} \log(|V_r|) - \frac{1}{2} y_r^T V_r^{-1} y_r$$

$$\log(L(\theta; y_e)) = -n_e/2 \log(\theta_e) - \frac{1}{2} (y_e^T y_e / \theta_e)$$

$$V_r = \text{Variance}(y_r)$$

$$\theta_e = \text{residual variance}$$

$$n_e = \text{residual degrees of freedom, and}$$

$$N = \text{number of observations used.}$$

For some balanced problems, y_e is treated as a multivariate residual, $[y_{e1}|y_{e2}|\dots|y_{en}]$, in order to increase the speed of the program and then the log-likelihood can be computed by the equivalent form:

$$\log(L(\theta; y_e)) = \frac{n}{2} \log(|V^{-1}|) - \frac{1}{2} \sum_{i=1}^n y_{ei}^T V^{-1} y_{ei}$$

where V is the dispersion matrix for the multivariate normal distribution.

For more information on REML, see Corbeil and Searle (1976).

Newton's Algorithm for maximization of restricted likelihood

As a general review, Newton's algorithm is an iterative algorithm used for finding the minimum of an objective function. The algorithm needs a starting point, and then at each iteration, it finds the next point using:

$$\theta_{i+1} = \theta_i - H^{-1}(\theta_i) g(\theta_i),$$

where H^{-1} is the inverse of the Hessian of the objective function, and g is the gradient of the objective function. The algorithm stops when a convergence criterion is met, or when it is unable to continue.

In the Linear Mixed Effects module, a modification of Newton's algorithm is used to minimize the negative restricted log-likelihood function with respect to the variance parameters θ , hence yielding the variance parameters that maximize the restricted likelihood. Since a constant term does not affect the minimization, the objective function used in the algorithm is the negative of the restricted log-likelihood without the constant term:

$$\text{objective function} = -\log(L(\theta; y_r)) - \log(L(\theta; y_e))$$

Newton's algorithm is modified by adding a diagonal matrix to the Hessian when determining the direction vector for Newton's, so that the diagonal matrix plus the Hessian is positive definite, ensuring a direction of descent:

$$\theta_{i+1} = \theta_i - (H_i + D_i)^{-1} g_i$$

The appropriate D_i matrix is found by a modified Cholesky factorization as described in Gill, Murray, and Wright. Another modification to Newton's algorithm is that, once the direction vector

$$d = -(H_i + D_i)^{-1} g_i$$

has been determined, a line search is done as described in Fletcher (1980) to find an approximate minimum along the direction vector.

Note from the discussion on REML estimation that the restricted log-likelihood is a function of the variance matrix V_r and the residual variance, which are in turn functions of the parameters θ . The gradient and Hessian of the objective function with respect to θ are therefore functions of first and second derivatives of the variances with respect to θ . Since the variances can be determined through the types specified for G and R (e.g., Variance Components, Unstructured, etc.), the first and second derivatives of the variance can

be evaluated in closed-form, and hence the gradient and Hessian are evaluated in closed-form in the optimization algorithm.

Starting values

Newton's algorithm requires initial values for the variance parameter θ . The user can supply these in the LinMix Wizard, General Options page, via the Initial Variance Parameters... button. More often, LinMix will determine initial values. For variance structures that are linear in θ (Variance Components, Unstructured, Compound Symmetry, and Toeplitz), the initial values are determined by the method of moments. If the variance structure is nonlinear, then LinMix will calculate a "sample" variance by solving for random effects and taking their sums of squares and cross-products. The sample variance will then be equated with the parametric variance, and the system of equations will be solved to obtain initial parameter estimates. If either method fails, LinMix will use values that should yield a reasonable variance matrix, e.g., 1.0 for variance components and standard deviations, 0.01 for correlations.

Sometimes the method of moments will yield estimates of θ that are REML estimates. In this case the program will not need to iterate.

Convergence criterion

The convergence criterion used in the Linear Mixed Effects module is that the algorithm has converged if:

$$g^T(\theta) H^{-1}(\theta) g(\theta) < \tau,$$

where τ is the convergence criterion specified on the General Options page of the wizard. The default value is 1×10^{-10} .

The possible convergence outcomes from Newton's algorithm are:

- Newton's algorithm converged with a final Hessian that is positive definite, indicating successful convergence at a local—and hopefully global—minimum.
- Newton's algorithm converged with a modified Hessian, indicating that the model may be over-specified. The output is suspect. The algorithm converged, but not at a local minimum. It may be in a trough, at a saddle point, or on a flat surface.

- Initial variance matrix is not positive definite. This indicates an invalid starting point for the algorithm.
- Intermediate variance matrix is not positive definite. The algorithm cannot continue.
- Failed to converge in allocated number of iterations.

Satterthwaite's approximation for degrees of freedom

The linear model used in linear mixed effects modeling is:

$$Y \sim N(Xb, V) \quad (10)$$

where X is a matrix of fixed known constants, b is an unknown vector of constants, and V is the variance matrix dependent on other unknown parameters. Wald (1943) developed a method for testing hypotheses in a rather general class of models. To test the hypothesis $H_0: Lb = \mathbf{0}$ in (10), the Wald statistic is:

$$(L\hat{\beta})^T [L(X^T \hat{V}^{-1} X)^{-1} L^T]^{-1} (L\hat{\beta}) \quad (11)$$

where $\hat{\beta}$ and \hat{V} are estimators of β and V . L must be a matrix such that each row is in the row space of X . The Wald statistic is asymptotically distributed under the null hypothesis as a chi-squared random variable with $q = \text{rank}(L)$ degrees of freedom. In common variance component models with balanced data, the Wald statistic has an exact F -distribution. Using the chi-squared approximation results in a test procedure that is too liberal. LinMix uses a method due to Allen (2001) for finding an F distribution to approximate the distribution of the Wald statistic.

Giesbrecht and Burns (1985) suggested a procedure to determine denominator degrees of freedom when there is one numerator degree of freedom. Their methodology applies to variance component models with unbalanced data. Allen derived an extension of their technique to the general mixed model.

A one degree of freedom (df) test is synonymous with L having one row in (11). When L has a single row, the Wald statistic can be re-expressed as:

$$\frac{(L\hat{\beta})^2}{L(X^T\hat{V}^{-1}X)^{-1}L^T} = \frac{(L\hat{\beta})^2}{L(X^TV^{-1}X)^{-1}L^T} \cdot \frac{L(X^T\hat{V}^{-1}X)^{-1}L^T}{L(X^TV^{-1}X)^{-1}L^T} \quad (12)$$

Consider the right side of (12). The distribution of the numerator is approximated by a chi-squared random variable with 1 df. The objective is to find v such that the denominator is approximately distributed as a chi-squared variable with v df. If:

$$\frac{L(X^T\hat{V}^{-1}X)^{-1}L^T}{L(X^TV^{-1}X)^{-1}L^T} \sim \frac{\chi^2_v}{v}$$

is true, then:

$$Var\left[\frac{L(X^T\hat{V}^{-1}X)^{-1}L^T}{L(X^TV^{-1}X)^{-1}L^T}\right] = \frac{2}{v} \quad (13)$$

The approach is to use (13) to solve for v .

Find an orthogonal matrix R and a diagonal matrix Λ such that:

$$L(X^T\hat{V}^{-1}X)^{-1}L^T = R\Lambda R^T$$

The Wald statistic (11) can be re-expressed as:

$$W = (R^TL\hat{\beta})^T\Lambda^{-1}(R^TL\hat{\beta}) = \sum_{u=1}^q \frac{(r_uL\hat{\beta})^2}{\lambda_u} \quad (14)$$

where r_u is the u -th column of R , and λ_u is the u -th diagonal of Λ . Each term in (14) is similar in form to (12). Assuming that the u -th term is distributed $F(1, v_u)$, the expected value of W is:

$$E(W) = \sum_{u=1}^q \frac{v_u}{v_u - 2}$$

Assuming W is distributed $F(q, v)$, its expected value is $qv/(v-2)$. Equating expected values and solving for v , one obtains:

$$v = 2 E\{W\} / (E\{W\} - q)$$

Comparing LinMix to ANOVA and SAS's PROC MIXED

This section outlines differences between LinMix and the previous WinNonlin ANOVA module, and between LinMix and PROC MIXED in SAS.

LinMix and ANOVA

Table 14-13. LinMix compared to WinNonlin ANOVA

Feature	LinMix	ANOVA
Degrees of freedom	Uses Satterthwaite approximation for all degrees of freedom, by default. User can also select Residual.	Uses only Residual.
Partial tests of model effects	The partial tests in LinMix are not equivalent to the Type III method in SAS though they coincide in most situations. See "Partial tests" on page 366 .	As SAS's PROC GLM Type III
Not estimable parameters	If a variable is not estimable, LinMix outputs 'Not estimable.' LinMix provides the option to write out 0 instead.	Prints "Not estimable"

LinMix and PROC MIXED

Table 14-14. LinMix compared to SAS PROC MIXED

Feature	LinMix	PROC MIXED
Degrees of freedom	Uses Satterthwaite approximation for all degrees of freedom, by default. User can also select Residual.	PROC MIXED chooses different defaults based on the model, random, and repeated statements. To make LinMix and SAS® consistent, set the SAS 'ddfm=satterth' option.
Multiple factors on the same random command	In LinMix, effects associated with the matrix columns from a single random statement are assumed to be correlated per the variance specified. Regardless of variance structure: If random effects are correlated, put them on the same random tab. If independent, on separate tabs.	SAS behaves similarly, except that it does not follow this rule in the case of Variance Components, so LinMix and SAS will differ for models with multiple terms on a random statement using the VC structure.
REML	REML (Restricted Maximum Likelihood) estimation, as defined in Corbeil and Searle (1976), does maximum likelihood estimation on the residual vector. This is how it is done in LinMix.	By default, SAS modifies the REML by 'profiling out' residual variance. To be consistent with LinMix, set the 'noprofile' option on the PROC MIXED statement. In particular, LinMix $-\log(\text{likelihood})$ may not equal twice SAS -2 Res Log Like if noprofile is not set.
Akaike and Schwarz's criteria	LinMix uses the smaller-is-better form of AIC and SBC in order to match WinNonlin and WinNonMix.	SAS uses the bigger-is-better form. In addition, AIC and SBC are functions of the likelihood, and the likelihood calculation method in LinMix also differs from SAS.
Saturated variance structure	If the random statement saturates the variance, the residual is not estimable. LinMix sets the residual variance to zero. A user-specified variance structure takes priority over that supplied by LinMix.	SAS sets the residual to some other number. The SAS residual needs to be added to the other SAS variance parameter estimates in order to match the LinMix output.
Convergence criteria	The convergence criterion is $g^T H^{-1} g < \text{tolerance}$ with H positive definite, where g is the gradient and H is the Hessian of the negative log-likelihood.	The default convergence criterion options in SAS are 'convh' and 'relative'. The LinMix criterion is equivalent to SAS with the 'convh' and 'absolute' options set.
Modified likelihood	LinMix always tries to keep the final Hessian (and hence the variance matrix) positive definite. It will determine when the variance is over-specified, and set some parameters to zero.	SAS allows the final Hessian be non positive definite, and hence will produce different final parameter estimates. This often occurs when the variance is over-specified (i.e., too many parameters in the model).

Table 14-14. LinMix compared to SAS PROC MIXED (continued)

Feature	LinMix	PROC MIXED
Keeping VC positive	For VC structure, LinMix keeps the final variance matrix positive definite, but will not force individual variances > 0.	If SAS estimates a negative variance component, it outputs 0.
Partial tests of model effects	The partial tests in LinMix are not equivalent to the Type III method in SAS though they coincide in most situations. See "Partial tests" on page 366.	The Type III results can change depending on how one specifies the model. For details, consult the SAS manual or contact support@pharsight.com .
Not estimable parameters	If a variable is not estimable, LinMix outputs 'Not estimable.' LinMix provides the option to write out 0 instead.	SAS writes the value of 0.
ARH, CSH variance structures	LinMix uses the standard deviations for the parameters, and prints the standard deviations in the output.	SAS prints the variances in the output for final parameters, though the model is expressed in terms of standard deviations.
Repeated without any time variable	LinMix requires an explicit repeated model. To run some SAS examples, a time variable would need to be added.	SAS doesn't require an explicit repeated model. It instead creates an implicit time field and assumes data are sorted appropriately.
Group as classification variable	LinMix requires variables used for the subject and group options to be declared as classification variables.	SAS allows regressors, but requires data to be sorted and considers each new regressor value to indicate a new subject or group. Classification variables are treated similarly.
Singularity tolerance	Define d_{ii} as the i -th diagonal element of the QR factorization of X , and D_{ii} the sum of squares of the i -th column of the QR factorization. The i -th column of X is deemed linearly dependent on the proceeding columns if $d_{ii}^2 < (\text{singularity tolerance}) \times D_{ii}$. If intercept is present, the first row of D_{ii} is omitted.	Define d_{ij} as the diagonal element of the $X^T X$ matrix during the sweeping process. If the absolute diagonal element $ d_{ij} < (\text{singularity tolerance}) \times D_{ij}$, where D_{ij} is the original diagonal of $X^T X$, then the parameter associated with the column is set to 0, and the parameter estimates are flagged as biased.

Bioequivalence

Calculating average, individual, and population bioequivalence

WinNonlin includes a Bioequivalence Wizard that calculates average or population/individual bioequivalence. Defined as relative bioavailability, *bioequivalence* involves comparison between test and reference drug products, where the test and reference products can vary, depending upon the comparison to be performed. Although bioavailability and bioequivalence are closely related, bioequivalence comparisons rely on a criterion, a predetermined bioequivalence limit, and calculation of a confidence interval for that criterion.

Further details about WinNonlin bioequivalence are provided under the following headings.

- [“Introduction to bioequivalence” on page 379](#)
- [“The model and data” on page 382](#)
- [“Average bioequivalence” on page 383](#)
- [“Individual and population bioequivalence” on page 391](#)
- [“Bioequivalence Wizard” on page 399](#)
- [“Bioequivalence output” on page 407](#)

Introduction to bioequivalence

Basic terms

Bioequivalence is said to exist when a test formulation has a bioavailability that is similar to that of the reference. There are three types of bioequivalence: Average, Individual, and Population.

The July 1992 FDA guidance on *Statistical Procedures for Bioequivalence Studies Using a Standard Two-Treatment Crossover Design* recommended that a standard *in vivo* bioequivalence study design be based on the administration of either single or multiple doses of the test and reference products to healthy subjects on separate occasions, with random assignment to the two possible sequences of drug product administration. Further, the 1992 guidance recommended that statistical analysis for pharmacokinetic parameters, such as area under the curve (AUC) and peak concentration (C_{max}) be based on a test procedure termed the two one-sided tests procedure to determine whether the average values for pharmacokinetic parameters were comparable for test and reference formulations. This approach is termed average bioequivalence and involves the calculation of a 90% confidence interval for the ratio of the averages of the test and reference products. To establish bioequivalence, the calculated confidence interval should fall within a bioequivalence limit, usually 80-125% for the ratio of the product averages. In addition to specifying this general approach, the 1992 guidance also provided specific recommendations for (1) logarithmic transformations of pharmacokinetic data, (2) methods to evaluate sequence effects, and (3) methods to evaluate outlier data.

The [US FDA Guidance for Industry \(January 2001\)](#) *Statistical Approaches to Establishing Bioequivalence* recommends that average bioequivalence be supplemented by two new approaches, population and individual bioequivalence. Population and individual bioequivalence include comparisons of both the averages and variances of the study measure. Population bioequivalence assesses the total variability of the measure in the population. Individual bioequivalence assesses within-subject variability for the test and reference products as well as the subject-by-formulation interaction.

Bioequivalence

Bioequivalence between two formulations of a drug product, sometimes referred to as relative bioavailability, indicates that the two formulations are therapeutically equivalent and will provide the same therapeutic effect. The objective of a bioequivalence study is to determine whether bioequivalence exists between a test formulation and reference formulation and to hence identify whether the two formulations are pharmaceutical alternatives that can be used interchangeably to achieve the same effect.

Bioequivalence studies are important because establishing bioequivalence with an already approved drug product is a cost-efficient way of obtaining drug approval. For example, to get approval to market a generic drug product, the Food and Drug Administration (FDA) usually does not require a new drug

application submission if the generic drug company can provide evidence of bioequivalence between the generic drug product and the innovator product. Bioequivalence studies are also useful for testing new formulations of a drug product, new routes of administration of a drug product, and for a drug product that has changed after approval.

Three different types of bioequivalence have been established by the FDA. These are average bioequivalence, population bioequivalence, and individual bioequivalence. All types of bioequivalence comparisons are based on the calculation of a criterion (or point estimate), on the calculation of a confidence interval for the criterion, and on a predetermined acceptability limit for the confidence interval.

A procedure for establishing average bioequivalence was recommended by the FDA in the 1992 guidance on “Statistical Procedures for Bioequivalence Studies Using a Standard Two-Treatment Crossover Design.” The FDA recommended administration of either single or multiple doses of the test and reference formulations to subjects, with random assignment to the possible sequences of drug administration. The guidance then recommended that a statistical analysis of pharmacokinetic parameters such as AUC and C_{max} be done on a log-transformed scale to determine the difference in the average values of these parameters between the test and reference data, and to determine the confidence interval for the difference. To establish bioequivalence, the confidence interval, usually calculated at the 90% confidence level, should fall within a predetermined acceptability limit, usually within 20% of the reference average. The FDA also recommended an equivalent approach using two, one-sided *t*-tests. Both the confidence interval approach and the two one-sided *t*-tests are described further below.

The January 2001 guidance “Statistical Approaches to Establishing Bioequivalence” recommends that average bioequivalence be supplemented by two new approaches, population and individual bioequivalence. These added approaches are needed because average bioequivalence uses only a comparison of averages, and not the variances, of the bioequivalence measures. In contrast, population and individual bioequivalence approaches include comparisons of both the averages and variances of the study measure. The population bioequivalence approach assesses the total variability of the study measure in the population. The individual bioequivalence approach assesses within-subject variability for the test and reference products as well as the subject-by-formulation interaction.

The concepts of population and individual bioequivalence are related to two types of drug interchangeability—prescribability and switchability. Drug pre-

scribability refers to when a physician prescribes a drug product to a patient for the first time, and must choose from a number of bioequivalent drug products. Drug prescribability is usually assessed by population bioequivalence. Drug switchability refers to when a physician switches a patient from one drug product to a bioequivalent product, such as when switching from an innovator drug product to a generic substitute within the same patient. Drug switchability is usually assessed by individual bioequivalence.

The model and data

Linear model

Like the Linear Mixed Effects Modeling wizard, the Bioequivalence Wizard is based on a mixed effects model. See [“The linear mixed effects model” on page 330](#). For more details on the models required for different kinds of bioequivalence, see the [US FDA Guidance for Industry \(August 1999\)](#).

Variance structures

The variance structures available in the Bioequivalence Wizard are the same as those available in the Linear Mixed Effects Modeling wizard. See [“Variance structure” on page 357](#).

Missing data

For population and individual bioequivalence, the program assumes complete data for each subject. If a subject has a missing observation, that subject is not included in the analysis. If the data have many missing observations, consider imputing estimated values to produce complete records per subject. This program does not impute missing values.

Variable name limits

for use in WinNonlin bioequivalence analyses, variable (column) names must begin with a letter. After the first character, valid characters include numbers and underscore (my_file_2). They cannot include spaces or the following operational symbols: +, -, *, /, =. They also cannot include parentheses (), question marks (?) or semicolons (;). Finally, variable names may not have single or double quotes in them.

Average bioequivalence

Study designs

The most common designs for bioequivalence studies are replicated cross-over, nonreplicated crossover, and parallel. In a parallel design, each subject receives only one formulation in randomized fashion, whereas in a crossover design each subject receives different formulations in different time periods. Crossover designs are further broken down into replicated and nonreplicated designs. In nonreplicated designs, subjects receive only one dose of the test formulation and only one dose of the reference formulation. Replicated designs involve multiple doses. The FDA recommends that a bioequivalence study should use a crossover design unless a parallel or other design can be demonstrated to be more appropriate for valid scientific reasons. Replicated crossover designs should be used for individual bioequivalence studies, and can be used for average or population bioequivalence analysis.

In the Bioequivalence Wizard, if all formulations have at least one subject given more than one dose of that formulation, then the study is considered a replicated design. For example, for the two formulations T and R, a 2×3 cross-over design as described in the table below is a replicated crossover design.

	Period 1	Period 2	Period 3
Sequence 1	T	R	T
Sequence 2	R	T	R

An example of a nonreplicated crossover design is the standard 2x2 crossover design described in the 1992 FDA guidance and in the table below.

	Period 1	Period 2
Sequence 1	T	R
Sequence 2	R	T

Recommended models for average bioequivalence

The recommended model to be used in average bioequivalence studies depends on the type of study design that was used – parallel, nonreplicated crossover, or replicated crossover.

Replicated crossover designs

For replicated crossover designs, the default model used in the Bioequivalence Wizard is the example model given for replicated designs in Appendix E of the FDA guidance:

Fixed effects model is:

- $\text{DependentVariable} = \text{Intercept} + \text{Sequence} + \text{Formulation} + \text{Period}$

Random effects model is:

- Treatment
- Variance Blocking Variable set to Subject
- Type set to Banded No-Diagonal Factor Analytic with 2 factors

Repeated specification is:

- Period
- Variance Blocking Variables set to Subject
- Group set to Treatment
- Type set to Variance Components

Rather than using the implicit repeated model as in Appendix E, the time variable is made explicit as Period.

Nonreplicated crossover designs

For nonreplicated crossover designs, the default model is as follows.

Fixed effects model is:

- $\text{DependentVariable} = \text{Intercept} + \text{Sequence} + \text{Formulation} + \text{Period}$

Random effects model is the nested term:

- $\text{Subject}(\text{Sequence})$

There is no repeated specification, so the default error model $\varepsilon \sim N(\mathbf{0}, \sigma^2 \mathbf{I})$ will be used. This is equivalent to the classical analysis method, but using maximum likelihood instead of method of moments to estimate inter-subject variance. Using Subject as a random effect this way, the correct standard errors will be computed for sequence means and tests of sequence effects.

Using a fixed effect model, one must construct pseudo- F tests by hand to accomplish the same task.

When this default model is used for a standard 2x2 crossover design, Win-Nonlin creates two additional tabs in the output called Sequential SS and Partial SS, which contain the degrees of freedom (DF), Sum of Squares (SS), Mean Squares (MS), F-statistic and p-value, for each of the model terms. These tables are also included in the text output. Note that the F-statistic and p-value for the Sequence term are using the correct error term since Subject(Sequence) is a random effect.

If, in addition to using the default model, the data is also ln-transformed, then the Final Variance Parameters tab will contain two additional parameters:

$$\text{Intersubject CV} = \sqrt{\exp(\text{Var}(\text{Subj}(\text{Seq})) - 1)}$$

$$\text{Intrasubject CV} = \sqrt{\exp(\text{Residual_Variance}) - 1}$$

Note that, for this default model, $\text{Var}(\text{Subj}(\text{Seq}))$ is the intersubject (between subject) variance, and Residual_Variance is the intrasubject (within subject) variance.

Parallel designs

For parallel designs, the default model is as follows.

Fixed effects model is:

- Formulation

There is no random model and no repeated specification, so the residual error term is included in the model.

Note: In each case, the user may supplement or modify the model to suit the analysis needs of the data set in consideration.

Least squares means

In determining the bioequivalence of a test formulation and a reference formulation, the first step is the computation of the least squares means (LSM) and standard errors of the test and reference formulations and the standard error of the difference of the test and reference least squares means. These

quantities are computed by the same process that is used for the Linear Mixed Effects module. See also “Least squares means” on page 348.

To simplify the notation for this section, let:

- RefLSM = reference least squares mean (computed by LinMix),
- TestLSM = test least squares mean (computed by LinMix),
- fractionToDetect = (user-specified percent of reference to detect) / 100,
- DiffSE = standard error of the difference in LSM (computed by LinMix),
- RatioSE = standard error of the ratio of the least squares means.

The geometric LSM are computed for transformed data. For ln-transform or data already ln-transformed,

$$\text{RefGeoLSM} = \exp(\text{RefLSM})$$

$$\text{TestGeoLSM} = \exp(\text{TestLSM})$$

For log10-transform or data already log10-transformed,

$$\text{RefGeoLSM} = 10^{\text{RefLSM}}$$

$$\text{TestGeoLSM} = 10^{\text{TestLSM}}$$

The difference is the test LSM minus the reference LSM,

$$\text{Difference} = \text{TestLSM} - \text{RefLSM}$$

The ratio calculation depends on data transformation. For non-transformed,

$$\text{Ratio}(\% \text{Ref}) = 100 (\text{TestLSM} / \text{RefLSM})$$

For ln-transform or data already ln-transformed, the ratio is obtained on arithmetic scale by exponentiating,

$$\text{Ratio}(\% \text{Ref}) = 100 \exp(\text{Difference})$$

Similarly for log10-transform or data already log10-transformed, the ratio is

$$\text{Ratio}(\% \text{Ref}) = 100 \cdot 10^{\text{Difference}}$$

Classical and Westlake confidence intervals

Output from the Bioequivalence module includes the classical confidence intervals and Westlake confidence intervals for confidence levels equal to 80, 90, 95, and for the confidence level that the user gave on the bioequivalence tab if that value is different than 80, 90, or 95. To compute the classical confidence intervals, first the following values are computed using the students- t distribution, where $\alpha = (100 - \text{confidenceLevel}) / 100$.

$$\text{lower} = \text{Difference} - t_{(1-\alpha/2), df} \cdot \text{DiffSE}$$

$$\text{upper} = \text{Difference} + t_{(1-\alpha/2), df} \cdot \text{DiffSE}$$

These values are then transformed if necessary to be on the arithmetic scale, and translated to percentages. For ln-transform or data already ln-transformed,

$$\text{CI_Lower} = 100 \cdot \exp(\text{lower})$$

$$\text{CI_Upper} = 100 \cdot \exp(\text{upper})$$

For log10-transform or data already log10-transformed,

$$\text{CI_Lower} = 100 \cdot 10^{\text{lower}}$$

$$\text{CI_Upper} = 100 \cdot 10^{\text{upper}}$$

For no transform,

$$\begin{aligned} \text{CI_Lower} &= 100 (1 + \text{lower} / \text{RefLSM}) \\ &= 100 \left(1 + \frac{(\text{TestLSM} - \text{RefLSM})}{\text{RefLSM} - t_{(1-\alpha/2), df}} \cdot \frac{\text{DiffSE}}{\text{RefLSM}} \right) \\ &= 100 \left(\frac{\text{TestLSM}}{\text{RefLSM} - t_{(1-\alpha/2), df}} \cdot \text{RatioSE} \right) \end{aligned}$$

where the approximation $\text{RatioSE} = \text{DiffSE} / \text{RefLSM}$ is used. Similarly,

$$\text{CI_Upper} = 100 (1 + \text{upper} / \text{RefLSM})$$

The procedure for computing the Westlake confidence intervals is given in many statistical references. See, for example, Chow and Liu (2000), page 86. Computing these intervals is an iterative procedure for which the Nelder-Mead simplex algorithm is used. It is possible that there will not be convergence, in which case the Bioequivalence module will print “Not Estimable.”

The bioequivalence conclusion that appears in the text output depends on the user-specified values for the confidence level and for the percent of reference to detect; these options are entered on the bioequivalence options tab of the Bioequivalence wizard. To conclude whether bioequivalence has been achieved, the CI_Lower and CI_Upper for the user-specified value of the confidence level are compared to the following lower and upper bounds. Note that the upper bound for log10 or ln-transforms or for data already transformed is adjusted so that the bounds are symmetric on a logarithmic scale.

$$\text{LowerBound} = 100 - (\text{percent of reference to detect})$$

$$\begin{aligned}\text{UpperBound (ln-transforms)} &= 100 \exp(-\ln(1 - \text{fractionToDetect})) \\ &= 100 (1 / (1 - \text{fractionToDetect}))\end{aligned}$$

$$\begin{aligned}\text{UpperBound (log10-transforms)} &= 100 \cdot 10^{(-\log_{10}(1.0 - \text{fractionToDetect}))} \\ &= 100 (1 / (1 - \text{fractionToDetect}))\end{aligned}$$

$$\text{UpperBound (no transform)} = 100 + (\text{percent of reference to detect})$$

If the interval (CI_Lower, CI_Upper) is contained within LowerBound and UpperBound, average bioequivalence has been shown. If the interval (CI_Lower, CI_Upper) is completely outside the interval (LowerBound, UpperBound), average bioinequivalence has been shown. Otherwise, the module has failed to show bioequivalence or bioinequivalence.

Two one-sided T-tests

For ln-transform or data already ln-transformed, the first t -test is a left-tail test of the hypotheses:

$$H_0: \text{Difference} < \ln(1 - \text{fractionToDetect}) \quad (\text{bioinequivalence for left test})$$

$$H_1: \text{Difference} \geq \ln(1 - \text{fractionToDetect}) \quad (\text{bioequivalence for left test})$$

The test statistic for performing this test is:

$$t_1 = ((\text{TestLSM} - \text{RefLSM}) - \ln(1 - \text{fractionToDetect})) / \text{DiffSE}$$

The p -value is determined using the t -distribution for this t -value and the degrees of freedom. If the p -value is < 0.05 , then the user can reject H_0 at the 5% level, i.e. less than a 5% chance of rejecting H_0 when it was actually true.

For log10-transform or data already log10-transformed, the first test is done similarly using log10 instead of ln.

For data with no transformation, the first test is (where Ratio here refers to $\text{Ratio}(\% \text{Ref}) / 100$):

$$H_0: \text{Ratio} < 1 - \text{fractionToDetect}$$

$$H_1: \text{Ratio} \geq 1 - \text{fractionToDetect}$$

The test statistic for performing this test is:

$$t_1 = [(\text{TestLSM} / \text{RefLSM}) - (1 - \text{fractionToDetect})] / \text{RatioSE}$$

where the approximation $\text{RatioSE} = \text{DiffSE} / \text{RefLSM}$ is used.

The second t -test is a right-tail test that is a symmetric test to the first. However for \log_{10} or \ln -transforms, the test will be symmetric on a logarithmic scale. For example, if the percent of reference to detect is 20%, then the left-tail test is $\Pr(<80\%)$, but for \ln -transformed data, the right-tail test is $\text{Prob}(>125\%)$, since $\ln(0.8) = -\ln(1.25)$.

For \ln -transform or data already \ln -transformed, the second test is a right-tail test of the hypotheses:

$$H_0: \text{Difference} > -\ln(1 - \text{fractionToDetect}) \text{ (bioinequivalence for right test)}$$

$$H_1: \text{Difference} \leq -\ln(1 - \text{fractionToDetect}) \text{ (bioequivalence for right test)}$$

The test statistic for performing this test is:

$$t_2 = ((\text{TestLSM} - \text{RefLSM}) + \ln(1 - \text{fractionToDetect})) / \text{DiffSE}$$

For \log_{10} -transform or data already \log_{10} -transformed, the second test is done similarly using \log_{10} instead of \ln .

For data with no transformation, the second test is:

$$H_0: \text{Ratio} > 1 + \text{fractionToDetect}$$

$$H_1: \text{Ratio} \leq 1 + \text{fractionToDetect}$$

The test statistic for performing this test is:

$$t_2 = ((\text{TestLSM} / \text{RefLSM}) - (1 + \text{fractionToDetect})) / \text{RatioSE}$$

where the approximation $\text{RatioSE} = \text{DiffSE} / \text{RefLSM}$ is used.

The output for the two one-sided t -tests includes the p -value for the first test described above, the p -value for the second test above, the maximum of these p -values, and total of these p -values. The two one-sided t -tests procedure is

operationally equivalent to the classical confidence interval approach. That is, if the classical $(1-2\times\alpha) \times 100\%$ confidence interval for difference or ratio is within LowerBound and UpperBound, then both the H_0 's given above for the two tests are also rejected at the alpha level by the two one-sided t -tests.

Anderson-Hauck test

See Anderson and Hauck (1983), or page 99 of Chow and Liu (2000). Briefly, the Anderson-Hauck test is based on the hypotheses:

$$H_{01}: \mu_T - \mu_R \leq \theta_L \quad \text{vs.} \quad H_{A1}: \mu_T - \mu_R > \theta_L$$

$$H_{02}: \mu_T - \mu_R \geq \theta_U \quad \text{vs.} \quad H_{A2}: \mu_T - \mu_R < \theta_U$$

where θ_L and θ_U are the lower and upper Anderson-Hauck limits; these limits are entered on the bioequivalence options tab. Rejection of both null hypotheses implies bioequivalence. The Anderson-Hauck test statistic is t_{AH} given by

$$t_{AH} = \frac{\bar{Y}_T - \bar{Y}_R - (\theta_L + \theta_U)/2}{\text{Diff } SE}$$

where $\text{Diff } SE$ is the standard error of the difference in means. Under the null hypothesis, this test statistic has a noncentral t -distribution.

Power

Power is the post-hoc probability of detecting a difference greater than or equal to a specified percentage of the reference least squares mean. In general,

$$\begin{aligned} \text{Power} &= 1 - (\text{probability of a type II error}) \\ &= \text{probability of rejecting } H_0 \text{ when } H_1 \text{ is true.} \end{aligned}$$

In the bioequivalence calculations, the hypotheses being tested are:

$$H_0: \text{RefLSM} = \text{TestLSM}$$

$$H_1: \text{RefLSM} \neq \text{TestLSM}$$

For the no-transform case, the power is the probability of rejecting H_0 given:

$$|\text{TestLSM} - \text{RefLSM}| \geq \text{fractionToDetect} \times \text{RefLSM}$$

For ln-transform, and data already ln-transformed, this changes to:

$$|\text{TestLSM} - \text{RefLSM}| \geq -\ln(1 - \text{fractionToDetect}),$$

and similarly for log10-transform and data already log10-transformed.

For the sake of illustration, assume $\text{fractionToDetect} = 0.2$, $\text{RefLSM} > 0$, and no transform was done on the data. Also the maximum probability of rejecting H_0 occurs in the case of equality, $|\text{TestLSM} - \text{RefLSM}| = 0.2 \times \text{RefLSM}$. So:

$$\begin{aligned} \text{Power} &= \Pr(\text{rejecting } H_0 \text{ at the alpha level given } |\text{Difference}| = 0.2 \times \text{RefLSM}) \\ &= \Pr(|\text{Difference}/\text{DiffSE}| > t_{(1-\alpha/2), \text{df}} \text{ given } |\text{Difference}| = 0.2 \times \text{RefLSM}) \\ &= \Pr(\text{Difference}/\text{DiffSE} > t_{(1-\alpha/2), \text{df}} \text{ given } |\text{Difference}| = 0.2 \times \text{RefLSM}) \\ &\quad + \Pr(\text{Difference}/\text{DiffSE} < -t_{(1-\alpha/2), \text{df}} \text{ given } |\text{Difference}| = 0.2 \times \text{RefLSM}) \end{aligned}$$

Let:

$$\begin{aligned} t_1 &= t_{(1-\alpha/2), \text{df}} - 0.2 \times \text{RefLSM} / \text{DiffSE} \\ t_2 &= -t_{(1-\alpha/2), \text{df}} - 0.2 \times \text{RefLSM} / \text{DiffSE} \\ t_{\text{stat}} &= (\text{Difference} - 0.2 \times \text{RefLSM}) / \text{DiffSE} \end{aligned}$$

Then:

$$\begin{aligned} \text{Power} &= \Pr(t_{\text{stat}} > t_1 \text{ given } |\text{Difference}| = 0.2 \times \text{RefLSM}) \\ &\quad + \Pr(t_{\text{stat}} < t_2 \text{ given } |\text{Difference}| = 0.2 \times \text{RefLSM}) \end{aligned}$$

Note that t_{stat} is T-distributed. Let p_1 be the p-value associated with t_1 (the area in the tail), and let p_2 be the p-value associated with t_2 (the area in the tail; note that p_2 may be negligible). Then:

$$\text{Power} = 1 - (p_1 - p_2)$$

Individual and population bioequivalence

Introduction

Individual and population bioequivalence are used to assess switchability and prescribability. Because individual bioequivalence relies on estimates of within-subject, within-formulation variation, replicated crossover designs are

required. This algorithm was developed by Francis Hsuan. Designs that can be appropriately analyzed include, but are not limited to,

Sequence	Period	Design
2	4	TRTR/RTRT
4	4	TRTR/RTRT/TRRT/RTTR
4	2	TT/RR/TR/RT
4	3	TRT/RTR/TRR/RTT
2	5	TRRTT/RTTRR
3	3	TRR/RTR/RRT
2	3	RTR/TRT
6	3	TRR/RTT/TRT/RTR/TTR/RRT
2	4	TRRR/RTTT
6	4	TTRR/RRTT/TRRT/RTTR/TRRR/RTTT

The algorithm works for balanced and unbalanced data with an equal number of periods in each sequence and one measurement per subject in each period.

Bioequivalence criterion

The FDA population bioequivalence (PBE) criteria are

$$\frac{E(Y_{jT} - Y_{jR})^2 - E(Y_{jR} - Y_{j'R})^2}{\frac{1}{2}E(Y_{jR} - Y_{j'R})^2} < \theta_P \quad \text{if } \sigma_R > \sigma_P$$

$$\frac{E(Y_{jT} - Y_{jR})^2 - E(Y_{jR} - Y_{j'R})^2}{\frac{1}{2}\sigma_P^2} < \theta_P \quad \text{if } \sigma_R \leq \sigma_P$$

where σ_P defaults to 0.2 and $\theta_P = (\ln(1 - \text{PercentReference}) + \varepsilon_P) / \sigma_P^2$. The default value for PercentReference is 0.20. In the wizard, σ_P is called the Total SD standard. σ_R^2 is computed by the program and is the total variance of the reference formulation, i.e., the sum of within- and between-subject variance. The criteria take the linearized form

$$\eta_{P1} = E(\mu_T - \mu_R)^2 + \sigma_T^2 - (1 + \theta_P)\sigma_R^2 < 0 \quad \text{if } \sigma_R > \sigma_P$$

$$\eta_{P2} = E(\mu_T - \mu_R)^2 + \sigma_T^2 - \sigma_R^2 - \theta_P \sigma_P^2 < 0 \quad \text{if } \sigma_R \leq \sigma_P$$

The FDA individual bioequivalence (IBE) criteria are

$$\frac{E(Y_{jT} - Y_{jR})^2 - E(Y_{jR} - Y_{j'R})^2}{\frac{1}{2}E(Y_{jR} - Y_{j'R})^2} < \theta_I \quad \text{if } \sigma_{WR} > \sigma_I$$

$$\frac{E(Y_{jT} - Y_{jR})^2 - E(Y_{jR} - Y_{j'R})^2}{\frac{1}{2}\sigma_I^2} < \theta_I \quad \text{if } \sigma_{WR} \leq \sigma_I$$

where σ_I defaults to 0.2 and $\theta_I = (\ln(1 - \text{PercentReference}) + \epsilon_I) / \sigma_I^2$. The default value for Percent Reference is 0.20, and the default value for ϵ_I is 0.05. In the wizard, σ_I is called the within-subject SD standard. σ_{WR}^2 is computed by the program and is the within-subject variance of the reference formulation. The IBE criteria take the linearized form

$$\eta_{I1} = E(\mu_T - \mu_R)^2 + \sigma_D^* + \sigma_{WT}^2 - (1 + \theta_I) \sigma_{WR}^2 < 0 \quad \text{if } \sigma_{WR} > \sigma_I$$

and

$$\eta_{I2} = E(\mu_T - \mu_R)^2 + \sigma_D^* + \sigma_{WT}^2 - \sigma_{WR}^2 - \theta_I \sigma_I^2 < 0 \quad \text{if } \sigma_{WR} \leq \sigma_I$$

where σ_D^* will be defined below.

For reference scaling, use η_{P1} or η_{I1} . For constant scaling, use η_{P2} or η_{I2} . For mixed scaling, use one of the following.

Population	If $\sigma_R > \sigma_P$	use η_{P1}
	If $\sigma_R \leq \sigma_P$	use η_{P2}
Individual	If $\sigma_{WR} > \sigma_I$	use η_{I1}
	If $\sigma_{WR} \leq \sigma_I$	use η_{I2}

If the upper confidence bound on the appropriate η is less than 0, then the product is bioequivalent in the chosen sense. The confidence interval is set on the Bioequivalence Options page. The method of calculating that upper confidence bound follows.

Details of computations

Let $Y_{ijk t}$ be the response of subject j in sequence i at time t with treatment k , and \tilde{Y}_{ij}^t be a vector of responses for subject j in sequence i . The components of Y are arranged in the ascending order of time. Let \tilde{Z}_{iT} and \tilde{Z}_{iR} be the design vector for treatment T and R respectively in sequence i , $n_i > 1$, be the number of subjects in sequence i .

Assume the mean responses of \tilde{Y}_{ij} , μ_i follow a linear model

$$\mu_i = \tilde{Z}_{iT}\mu_T + \tilde{Z}_{iR}\mu_R \quad (1a)$$

where μ_T, μ_R are the population mean responses of Y with treatments T and R respectively, and X_i are design/model matrices.

Let $p_{iT} := \text{sum}(\tilde{Z}_{iT})$ and $p_{iR} := \text{sum}(\tilde{Z}_{iR})$ be the number of occasions T and R respectively be assigned to the subjects in sequence i ,

$$\underline{u}_{iT} = p_{iT}^{-1} \tilde{Z}_{iT}^T,$$

$$\underline{u}_{iR} = p_{iR}^{-1} \tilde{Z}_{iR}^T,$$

$$\underline{u}_{iT} = (\underline{u}_{iT} - \underline{u}_{iR}),$$

$$\text{and } \underline{d}_i = s_T^{-1} \underline{u}_{iT} - s_R^{-1} \underline{u}_{iR}.$$

Assume that the covariances $(Y_{ijk t}, Y_{ijk' t'})$ follow the model

$$\begin{aligned} \Sigma_i = & \sigma_{WT}^2 \text{diag}(\tilde{Z}_{iT}) + \sigma_{WR}^2 \text{diag}(\tilde{Z}_{iR}) + \sigma_{TT} \tilde{Z}_{iT} \tilde{Z}_{iT}^T \\ & + \sigma_{RR} \tilde{Z}_{iR} \tilde{Z}_{iR}^T + \sigma_{TR} \{ \tilde{Z}_{iT} \tilde{Z}_{iR}^T + \tilde{Z}_{iR} \tilde{Z}_{iT}^T \} \end{aligned} \quad (1b)$$

where the parameters

$$\underline{\sigma} = \{ \sigma_{WT}^2, \sigma_{WR}^2, \sigma_{TT}, \sigma_{RR}, \sigma_{TR} \}$$

are defined as follows:

$$\sigma_T^2 = \sigma_{TT} + \sigma_{WT}^2 \text{ and } \sigma_R^2 = \sigma_{RR} + \sigma_{WR}^2 \text{ are var } (Y_{ijTl}) \text{ and var } (Y_{ijRl}).$$

$$\rho_{TT} = \sigma_{TT}/\sigma_T^2 = \text{intra-subject correlation coeff. corr}(Y_T, Y_T), \\ 1 > \rho_{TT} > 0.$$

$$\rho_{RR} = \sigma_{RR}/\sigma_R^2 = \text{intra-subject correlation coeff. corr } (Y_R, Y_R), \\ 1 > \rho_{RR} > 0.$$

$$\rho_{TR} = \sigma_{TR}/(\sigma_T\sigma_R) = \text{intra-subject coeff. corr } (Y_T, Y_R), \ 1 > \rho_{TR} > -1.$$

$$\sigma_{TT} = \rho_{TT}\sigma_T^2; \sigma_{RR} = \rho_{RR}\sigma_R^2 \text{ and } \sigma_{TR} = \rho_{TR}\sigma_T\sigma_R \text{ are intra-subject covariances.}$$

$$\sigma_{WT}^2 = \sigma_T^2 - \sigma_{TT} \text{ is the intra-subject variance } (Y_T - Y_T)/2.$$

$$\sigma_{WR}^2 = \sigma_R^2 - \sigma_{RR} \text{ is the intra-subject variance } (Y_R - Y_R)/2.$$

For PBE and IBE investigations, it is useful to define additional parameters:

$$\sigma_D^* = \sigma_{TT} + \sigma_{RR} - 2\sigma_{TR} \tag{2a}$$

$$\sigma_{iT}^{*2} = (\sigma_T^2 - (1 - p_{iT}^{-1})\sigma_{WT}^2) \tag{2b}$$

$$\sigma_{iR}^{*2} = (\sigma_R^2 - (1 - p_{iR}^{-1})\sigma_{WR}^2) \tag{2c}$$

$$\sigma_{iI}^{*2} = (\sigma_D^* + p_{iT}^{-1}\sigma_{WT}^2 + p_{iR}^{-1}\sigma_{WR}^2) \tag{2d}$$

Except for (2a), all the quantities above are non-negative when they exist. The quantity is similar, but not identical, to the “subject-by-formulation” variance component in the mixed model discussed in the FDA guidance Section III. It satisfies the equation

$$\text{var}(Y_{ijT} - Y_{ijR}) = \sigma_D^* + \sigma_{WT}^2 + \sigma_{WR}^2$$

In general, this σ_D^* may be negative, while FDA's σ_D^2 is always non-negative. The FDA mixed model is a special case of the multivariate model above, and the quantity σ_D^* is identical to σ_D^2 whenever the former is nonnegative. This method for PBE/IBE is based on the multivariate model. It yields identical results to those described in the FDA guidance Appendices F & H when the design is *TRTR/RTTR*. This method is applicable to a variety of higher-order two-treatment crossover designs including *TR/RT/TT/RR* (the Balaam Design), *TRT/RTR*, or *TxRR/xRTT/RTxx/TRxx/xTRR/RxTT* (Table 5.7 of Jones and Kenward, page 205).

Given the i th sequence, let

$$\hat{\Delta} = \sum_i d_i' \bar{Y}_i.$$

$$S_{iT} = u_{iT}' V_i u_{iT}, S_{iR} = u_{iR}' V_i u_{iR}, S_{iI} = u_{iI}' V_i u_{iI}$$

$$S_{iWT} = (1 - p_{iT}^{-1})^{-1} (\text{tr}\{\text{diag}(u_{iT}) V_i\} - u_{iT}' V_i u_{iT})$$

$$S_{iWR} = (1 - p_{iR}^{-1})^{-1} (\text{tr}\{\text{diag}(u_{iR}) V_i\} - u_{iR}' V_i u_{iR})$$

where \bar{Y}_i is the sample mean of the i th sequence, and V_i is the within-sequence sample covariance matrix. It can be shown that

$$\hat{\Delta} \sim N(\mu_T - \mu_R, \sum_i n_i^{-1} d_i' \sum_i d_i) \quad (4a)$$

$$S_{iT} \sim \sigma_{iT}^{*2} (n_i - 1)^{-1} \chi_{n_i - 1}^2 \quad (4b)$$

$$S_{iR} \sim \sigma_{iR}^{*2} (n_i - 1)^{-1} \chi_{n_i - 1}^2 \quad (4c)$$

$$S_{iI} \sim \sigma_{iI}^{*2} (n_i - 1)^{-1} \chi_{n_i - 1}^2 \quad (4d)$$

$$S_{iWT} \sim \sigma_{WT}^2 (p_{iT} - 1)^{-1} (n_i - 1)^{-1} \chi_{(p_{iT} - 1)(n_i - 1)}^2 \quad (4e)$$

$$S_{iWR} \sim \sigma_{WR}^2 (p_{iR} - 1)^{-1} (n_i - 1)^{-1} \chi^2_{(p_{iT} - 1)(n_i - 1)} \quad (4f)$$

Furthermore, it can be shown that $\{S_{iT}, S_{iR}\}$ (for PBE) are statistically independent from $\{\Delta\}$ and $\{S_{iWT}, S_{iWR}\}$, and that the four statistics $\Delta, S_{iI}, S_{iWT}, S_{iWR}$ (for IBE) are statistically independent.

Let α_i 's be sets of normalized weights, chosen to yield the method of moments estimates of the η 's. Then define the estimators of the components of the linearized criterion by

$$\begin{aligned} E_0 &= \Delta^2 \\ E_{P1} &= \sum_i \alpha_{iT} S_{iT} \\ E_{P2} &= \left(1 - \sum_i \alpha_{iT} p_{iT}^{-1} \right) \sum_i \alpha_{iWT} S_{iWT} \\ E_{P3} &= \sum_i \alpha_{iR} S_{iR} \\ E_{P4} &= \left(1 - \sum_i \alpha_{iR} p_{iR}^{-1} \right) \sum_i \alpha_{iWR} S_{iWR} \\ E_{I1} &= \sum_i \alpha_{iI} S_{iI} \\ E_{I2} &= \left(1 - \sum_i \alpha_{iI} p_{iT}^{-1} \right) \sum_i \alpha_{iWT} S_{iWT} \\ E_{I3a} &= \left(1 + \theta_I + \sum_i \alpha_{iI} p_{iR}^{-1} \right) \sum_i \alpha_{iWR} S_{iWR} \\ E_{I3b} &= \left(1 + \sum_i \alpha_{iI} p_{iR}^{-1} \right) \sum_i \alpha_{iWR} S_{iWR} \end{aligned}$$

Using the above notation, one may define unbiased moment estimators for the PBE criteria:

$$\hat{\eta}_{P1} = \hat{\Delta}^2 + E_{P1} + E_{P2} - (1 + \theta_P)E_{P3} - (1 + \theta_P)E_{P4}$$

$$\hat{\eta}_{P2} = \hat{\Delta}^2 + E_{P1} + E_{P2} - E_{P3} - E_{P4} - \theta_P \sigma_P^2$$

and for the IBE criteria:

$$\eta_{I1} = \hat{\Delta}^2 + E_{I1} + E_{I2} - E_{I3a}$$

$$\eta_{I2} = \hat{\Delta}^2 + E_{I1} + E_{I2} - E_{I3b} - \theta_I \sigma_I^2$$

The FDA Guidance specifies a procedure to construct a 95% upper bound for η based on the *TRTR/RTTR* design using Howe's approximation I and a modification proposed by [Hyslop, Hsuan and Holder \(2000\)](#). This can be generalized to compute the following sets of n_q , H and U statistics:

$$H_0 = \left(|\hat{\Delta}| + t_{.95, n_0} (\hat{\text{var}}(\hat{\Delta}))^{1/2} \right)^2$$

$$H_q = n_q E_q / \chi_{.05, n_q}^2 \quad \text{for } q = P1, P2, I1 \text{ and } I2.$$

$$H_q = n_q E_q / \chi_{.95, n_q}^2 \quad \text{for } q = P3, P4, I3a \text{ and } I3b.$$

$$U_q = (H_q - E_q)^2 \quad \text{for all } q$$

where the degrees of freedom n_q are computed using Satterthwaite's approximation. Then, the 95% upper bound for each η is

$$H_\eta = \hat{\eta} + \left(\sum U_q^* \right)^{1/2}$$

where $U_q^* = (1 + \theta_P)^2 U_q$ for $q = P3, P4$, and $U_q^* = U_q$ for all other q . If $H_q < 0$, that indicates bioequivalence; $H_q \geq 0$ fails to show bioequivalence.

Bioequivalence Wizard

To set up and run a bioequivalence analysis:



1. Open the data set to be analyzed.
2. Start the Bioequivalence Wizard by choosing **Tools>Bioequivalence Wizard** or by clicking on the **Bioequivalence Wizard** tool bar button.

3. The Bioequivalence Wizard provides options to use prior model settings (optional), as follows. To create a new analysis, proceed to step 4 below.
 - To load a previously-saved bioequivalence model, click the **Load** button and open a LinMix/bioequivalence command (*.LML) file containing bioequivalence settings.
 - To reload the most recently run model, click the **Previous Model** button.
 - Click **Calculate** to run the model.
4. *Type of Bioequivalence*: Select individual/population or average bioequivalence using the radio buttons at the upper right. For a discussion of the analysis types, see “Average bioequivalence” on page 383 and “Individual and population bioequivalence” on page 391.
5. For average bioequivalence, select the study type at the top left.

Based on the analysis and study types, WinNonlin will try to match the variable names from the active data set with the Subject, Sequence, Period, and Formulation fields.

6. If WinNonlin has not already selected the appropriate variables, select the variables that identify the Subject, Sequence, Period, and Formulation.
7. Select the reference formulation, against which to test for bioequivalence, from the pull down list labeled Reference Value.

Note: For **Average** bioequivalence with a parallel study type, only **Formulation** and its reference value need to be specified.

8. At any point in the wizard, the **Save** button can be used to save the analysis settings, including variable names, to a command (*.LML) file for later re-use.
9. Click **Next** to proceed with the analysis:
 - “Using average bioequivalence” on page 400
 - “Using population/individual bioequivalence” on page 405

Using average bioequivalence

See the *Examples Guide* for an example using average bioequivalence.

Average bioequivalence analysis settings fall under the following headings:

- “Fixed effects for average bioequivalence” on page 400
- “Random effects” on page 402
- “Repeated variance” on page 403
- “Bioequivalence options” on page 404
- “Bioequivalence general options” on page 405

Fixed effects for average bioequivalence

The dependent variable(s), regressor(s)/covariate(s), and sort variable(s) must be specified. For Average bioequivalence other classification variables can be added by dragging them from the Variable Collection to the Classification

Variable field. The [Bioequivalence Wizard](#) Fixed Effects dialog works very much like the LinMix Fixed Effects dialog. See “[Fixed effects](#)” on page 356.

To define the fixed effects model:

1. Drag and drop variables to categorize those needed in the LinMix model:
 - *Dependent variables* contain the values to which the model will be fit, e.g., drug concentration.
 - *Classification variables* or factors are categorical independent variables, e.g., formulation, treatment, and gender.
 - *Regressor variables* or *covariates* are continuous independent variables, e.g., temperature or body weight.
 - *Sort variables* define subsets of data to be processed separately, i.e., a separate analysis will be done for each level of the *sort variables*. If the sort variable has missing values, the analysis will be performed for the missing level and MISSING will be printed as the sort variable value.
 - Select the *weight variable* if weights for each record are included in a separate column. The weights are used to compensate for observations having different variances. When a weight variable is specified, each row of data is multiplied by the square root of the corresponding weight. Weight variable values should be proportional to the reciprocals of the variances. In the most common situation, the data are averages and weights are sample sizes associated with the averages. See also “[Weighting](#)” on page 243.

The Weight Variable cannot be a classification variable. It must have been declared as a regressor/covariate before it can be used as a weight variable. It can also be used in the model.

For Average bioequivalence the Model Specification field automatically displays an appropriate fixed effects model for study type. Edit the model as needed. For discussion of the model, see “[The model and data](#)” on page 382.

Note: To reuse a model in a command file (*.LML), click the **Load Model** button. To reuse the most recently run model, click the **Previous Model** button.

2. *Fixed Effects Confidence Level*: Accept the default setting [95] or enter the confidence level for the parameter estimates. The confidence level for the bioequivalence test is specified later.
3. To transform the dependent variable values before analysis, select a log transformation from the pull down list for Dependent Variables Transformation. If the input data are already transformed, select **Already Ln-transformed** or **Already Log10-transformed**.
4. Click **Calculate** to run the model with default settings, or **Next** to specify [Random effects](#) or [Repeated variance](#).

Random effects

Use this tab of the [Bioequivalence Wizard](#) to set traditional variance components and/or to random coefficients; else see [“Repeated variance”](#) on page 403.

Note: The [Bioequivalence Wizard](#) Variance Structure dialog operates exactly as it does in the LinMix wizard. (See also [“Variance structure”](#) on page 357.)

1. *Classification variables and regressors*: Drag and drop column names (or type them) to as needed for the random effects model(s).
2. *Variance Blocking Variables (Subject)*: (Optional) This variable must be a classification model term. It identifies the subjects in a data set. Complete

independence is assumed among subjects. Thus, Subject produces a block diagonal structure with identical blocks.

3. *Group variable*: (Optional) This variable must be a classification model term. It defines an effect specifying heterogeneity in the covariance structure. All observations having the same level of the group effect have the same covariance parameters. Each new level of the group effect produces a new set of covariance parameters with the same structure as the original group.

4. *Type*: If a random effects model is specified, select the type of covariance structure here. See “[Variance structure](#)” on page 357 for details on each type.
5. Check **Random Intercept** to indicate that the intercept is random. This is most commonly employed when a subject is specified in the Variance Blocking field. The default is off—no random intercept.
6. Click the **Add Random** button to create multiple variance models, if needed.
7. Click **Calculate** to run the analysis or **Next** to set [Bioequivalence options](#).

Repeated variance

Select the Repeated tab of the [Bioequivalence Wizard](#) to specify the R matrix in the mixed model. If no Repeated statement is specified, R is assumed to be equal to $\sigma^2 I$. The repeated effect must contain only classification variables. See also “[Repeated measurements](#)” on page 360.

To specify the repeated variance settings:

1. *Repeated Specification*: Enter the model by typing it or by dragging the classification variable names and clicking on the operators. Syntax rules for the repeated specification are the same as those for the fixed-effects model terms.
2. *Variance Blocking Variables (Subject)*: (Optional) This variable must be a classification model term. This Subject field identifies subjects in the data set. Complete independence is assumed across subjects. Thus, Subject produces a block diagonal structure with identical blocks.
3. *Group variable*: (Optional) This must be a classification variable with no regressors or operators. It defines an effect specifying heterogeneity in the covariance structure. All observations having the same level of the group effect have the same covariance parameters. Each new level of the group effect produces a new set of covariance parameters with the same structure as the original group.
4. *Type*: If a Repeated Specification has been included, select the type of covariance structure. “[Variance structure](#)” on page 357 for types.

Note: The default Repeated model depends upon whether the crossover study is replicated or not. The default model follows the FDA guidance.

5. Click **Calculate** to run the analysis or **Next** to set [Bioequivalence options](#).

Bioequivalence options

This dialog of the [Bioequivalence Wizard](#) sets the range for bioequivalence.

To set the bioequivalence options:

1. The following options are pre-set to meet FDA requirements. See “[Average bioequivalence](#)” on page 383.
 - Confidence Level (90%)
 - Percent of Reference to Detect (20%)
 - Anderson-Hauck Lower Limit (0.8)
 - Anderson-Hauck Upper Limit (1.25 for log-transformed data; 1.2 for not transformed)

2. Click **Calculate** to run the analysis or **Next** for [Bioequivalence general options](#).

Bioequivalence general options

This dialog of the [Bioequivalence Wizard](#) sets the following operations:

- ASCII output can be selected as an additional output form.
- A title can be assigned as a header for the ASCII output. Press **Ctrl+Enter** to move to the next line, up to a maximum of five lines.
- The kind of degrees of freedom can be stipulated.
- The maximum number of iterations can be set.
- How output that is not estimable is to be represented may be set.
- The advanced user options include:
 - Initial estimates for the Variance Parameters
 - Change defaults for Singularity Tolerance, Convergence Criterion, and Intermediate Calculations.

When all the appropriate settings have been made, click the **Calculate** button to run the model. The average bioequivalence calculations begin and the output appears in a new workbook (and, an optional ASCII text window).

Using population/individual bioequivalence

Following is an overview of the basic steps to perform population/individual bioequivalence in the [Bioequivalence Wizard](#). See “[Individual and population bioequivalence](#)” on page 391 for a discussion of theory. See the *Examples Guide* for an example.

To specify population/individual bioequivalence information:

1. Launch the Bioequivalence Wizard and set up the variables for the analysis as described under “[Bioequivalence Wizard](#)” on page 399.

Note: Population/Individual bioequivalence can only use replicated crossover study data. Each sequence must contain the same number of periods. For each period each subject must have one measurement.

2. Click **Calculate** to run the model with the default settings, or **Next** to proceed to the [Fixed effects for population/individual bioequivalence](#).

Note: Use the Load Model button to load a previously-saved model. Use the Previous Model... button to reload the last-run model.

Fixed effects for population/individual bioequivalence

The Fixed effects model is prebuilt following FDA guidelines. The [Bioequivalence Wizard Fixed Effects](#) dialog for population/individual bioequivalence works very much like that for LinMix. See “[Fixed effects](#)” on page 356.

To complete the model:

1. Drag the appropriate term(s), e.g., concentration, from the Variable Collection to the Dependent Variables field.

Appropriate terms for study type are specified automatically as classification variables. For Population bioequivalence, classification variables, regressors, and fixed effects model are disabled and preset to FDA guidelines.

2. (Optional) Select a log transformation from the pull down list for Dependent Variables Transformation. Select **Ln(x)** or **Log(x)** to transform the values before analysis. If the input data are already transformed, select **Already Ln-transformed** or **Already Log10-transformed**.
3. Click **Calculate** to run the model, or **Next** to proceed to the [Bioequivalence options for population/individual bioequivalence](#).

Bioequivalence options for population/individual bioequivalence

To set the bioequivalence options:

1. The following options are pre-set to meet FDA standards. See “[Individual and population bioequivalence](#)” on page 391.
 - Confidence Level (95%)
 - Percent of Reference to Detect (20%)
 - Population Limits: Total SD Standard (0.2) and Epsilon (0.02)

- Individual Limits: Within Subject SD Standard (0.2) and Epsilon (0.05)
2. To start calculating Population/Individual bioequivalence, click **Calculate**.

Bioequivalence output

The Bioequivalence Wizard generates a new bioequivalence workbook. Population/individual bioequivalence always generates text output as well. Average bioequivalence can produce text output as an option.

Average bioequivalence

Average bioequivalence generates a workbook containing following output.

Table 15-1. Average bioequivalence output worksheets

Worksheet	Contents
Average Bioequivalence	Output from the bioequivalence analysis
Ratio Tests	Ratios of reference to test mean computed for each subject
Diagnostics	Number of observations, number of observations used, residual sums of squares, residual degrees of freedom, residual variance, restricted log likelihood, AIC, SBC, and Hessian eigenvalues
Sequential Tests	Sort Variable(s), Dependent Variable(s), Units, Hypothesis, Hypothesis degrees of freedom, Denominator degrees of freedom, F statistic, and p-value
Sequential SS	Only for 2x2 crossover designs using the default model: ANOVA sums of squares, degrees of freedom, mean squares, F-tests and p-values for sequential test
Partial Tests	Sort Variable(s), Dependent Variable(s), Units, Hypothesis, Hypothesis degrees of freedom, Denominator degrees of freedom, F statistic, and p-value
Partial SS	Only for 2x2 crossover designs using the default model: ANOVA sums of squares, degrees of freedom, mean squares, F-tests and p-values for partial test
Final Fixed Parameters	Sort variables, dependent variable(s), units, effect level, parameter estimate, t-statistic, p-value, confidence intervals, etc.
Final Variance Parameters	Final estimates of the variance parameters
Parameter Key	Variance parameter names assigned by WinNonlin

Table 15-1. Average bioequivalence output worksheets (continued)

Worksheet	Contents
Least Squares Means	Sort Variable(s), Dependent Variable(s), Units, and the least squares means for the test formulation
LSM Differences	Difference in least squares means
Residuals	Sort Variable(s), Dependent Variable(s), Units, and information on residual effects vs. predicted and observed effects
User Settings	User-specified settings
Initial Variance Parameters	Parameter and estimates for variance parameters
Iteration History	Iteration, Obj_function, and column for each variance parameter

To determine average bioequivalence of a test and reference formulation, the first step is computation of the least squares means of the test and reference formulations and their standard errors, and computation of the standard error of the difference between test and reference least squares means, and degrees of freedom for the difference. The Linear Mixed Effects module, LinMix, computes these quantities. Therefore the output for average bioequivalence includes all the output obtained by running LinMix, as described in [“LinMix output” on page 365](#). The output also contains the Average Bioequivalence Statistics. These are described in detail in [“Introduction to bioequivalence” on page 379](#) and [“Average bioequivalence” on page 383](#). For crossover designs, the output also contains the ratios table, described under [“Ratio tables” on page 410](#).

Population/Individual bioequivalence

Table 15-2. Population/individual output worksheets

Worksheet	Contents
Population/Individual Bioequivalence	Output from the bioequivalence analysis
Ratio Tests	Ratios of reference to test mean
User Settings	User-specified settings

Depending which settings were specified in the wizard, the output may not include all of these worksheets.

Errors and warnings in the computations appear in the ASCII output as well as the application log or data history.

The population/individual bioequivalence statistics are described further in [“Introduction to bioequivalence” on page 379](#). They include:

Difference (Delta) = difference in sample means between the test and reference formulations.

Ratio(%Ref) = ratio of the test to reference means expressed as a percent. This is compared with the percent of reference to detect that was specified by the user, to determine if bioequivalence has been shown. For example, if the user specified a percent of reference to detect of 20%, and also specified a ln-transform, then Ratio(%Ref) needs to be in the interval (80%, 125%) to show bioequivalence for the ratio test.

Note: The FDA guidance suggests that both the ratio test and the tests listed below need to be passed to indicate bioequivalence.

SigmaR - value that is compared with sigmaP, the Total SD Standard, to determine whether mixed scaling for population bioequivalence will use the constant-scaling values or the reference-scaling values.

SigmaWR - value that is compared with sigmaI, the Within Subject SD Standard, to determine whether mixed scaling for individual bioequivalence will use the constant-scaling values or the reference-scaling values.

Each of the following is given, with its upper confidence bound and bioequivalence conclusion. An upper confidence bound < 0 indicates bioequivalence.

Table 15-3. Population/Individual Bioequivalence Calculations

Statistic	Indication
Const_Pop_eta	Test statistic for population bioequivalence with constant scaling
Ref_Pop_eta	Test statistic for population bioequivalence with reference scaling
Mixed_Pop_eta	Test statistic for population bioequivalence with mixed scaling
Const_Indiv_eta	Test statistic for individual bioequivalence with constant scaling
Ref_Indiv_eta	Test statistic for individual bioequivalence with reference scaling
Mixed_Indiv_eta	Test statistic for individual bioequivalence with mixed scaling

For crossover designs, the output also contains the ratio tables, below.

Ratio tables

For any bioequivalence study done on a crossover design, the output also includes for each test formulation a table of differences and ratios of the original data computed individually for each subject. This is in response to the FDA guidance “Measures for each individual should be displayed in parallel for the formulations tested. In particular, for each BE measure the ratio of the individual geometric mean of the T product to the individual geometric mean of the R product should be tabulated side by side for each subject.”

Let *test* be the value of the dependent variable measured after administration of the test formulation for one subject. Let *ref* be the corresponding value for the reference formulation, for the case in which only one dependent variable measurement is made. If multiple test or reference values exist for the same subject, then let *test* be the mean of the values of the dependent variable measured after administration of the test formulation, and similarly for *ref*, except for the cases of ‘ln-transform’ and ‘log10-transform’ where the geometric mean should be used:

$$\text{‘ln-transform’}: \quad \text{test} = \exp\left(\sum_i \ln(X_i)/N\right)$$

$$\text{‘log10-transform’}: \quad \text{test} = 10^{N^{-1} \sum_i \log_{10} X_i}$$

where X_i are the measurements after administration of the test formulation, and similarly for *ref*. For each subject, for the cases of no transform, ln-transform, or log10-transform, the ratios table contains:

$$\text{Difference} = \text{test} - \text{ref.}$$

$$\text{Ratio}(\% \text{Ref}) = 100 \times \text{test} / \text{ref.}$$

For data that was specified to be ‘already ln-transformed,’ these values are back-transformed to be in terms of the original data:

$$\text{Difference} = \exp(\text{test}) - \exp(\text{ref}).$$

$$\text{Ratio}(\% \text{Ref}) = 100 \times \exp(\text{test} - \text{ref}) = 100 \times \exp(\text{test}) / \exp(\text{ref})$$

Similarly, for data that was specified to be ‘already log10-transformed’:

$$\begin{aligned}\text{Difference} &= 10^{\text{test}} - 10^{\text{ref}} \\ \text{Ratio}(\% \text{Ref}) &= 100 \times 10^{\text{test} - \text{ref}} = 100 \times 10^{\text{test}} / 10^{\text{ref}}\end{aligned}$$

Note: For ‘already transformed’ input data, if the mean is used for *test* or *ref*, and the antilog of *test* or *ref* is taken above, then this is equal to the geometric mean.

Simulation

Generating response values based on a model, a set of time (X variable) values, and doses

WinNonlin simulations can use library models or user defined models to calculate response values at specified time points. This capability is especially useful, for example, in simulating steady state responses from single dose data, comparing drug concentrations for different dosing regimens or for different values of model parameters (such as absorption or elimination rates), or determining optimal sampling times. Simulations are also useful when writing custom models, to explore the model shape and behavior. Running a simulation of a model can quickly uncover errors in the model specification.

Simulating responses

Simulation of output values requires a set of time or other X-variable values, a model, parameter values for that model, and dosing constants (for some models). Simulations can use any compiled or ASCII model, including a user model. The model, data variables, and dosing are set up as for model fitting, as detailed under [“Modeling” on page 195](#), except that no Y variable is specified, and the Simulate check box, in the Data Variables dialog, must be checked. An example is provided below, to illustrate.

See the *Examples Guide* for an example using simulation to evaluate competing sampling schedules, and an example of simulation using a user-defined ASCII model.

Simulation of a compiled library model

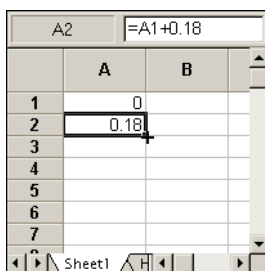
This example uses a compiled WinNonlin library model to illustrate the use of simulation to generate multiple-dose concentrations based on parameters from fitting single-dose data. Note that a compiled or ASCII model could be used.

Creating the input data

Simulation requires a data set with X values defining the time points over which to simulate Y variable values. This example uses a data set with times ranging from 0 to 36 hours, in increments of 0.18 hours.

To create the time value data:

1. Select **File>New** from the WinNonlin menus.
2. In the resulting dialog, select the **Workbook** radio button and click **OK**. A new workbook appears with two worksheets: the first, for data, the second, an automatic record of the data history.
3. Enter time values in the first column. To enter all of the times quickly; enter 0 in the first row, and $=A1+0.18$ in cell A2. Press the **Enter** key.
4. Next, highlight cell A2 and position the cursor over the lower right hand corner of the cell until the cursor changes into a small black cross.



5. Hold the left mouse button down, and drag down to Row 201. The formula will be copied, incrementing by 0.18 in each row.
6. Use **Data>Column Properties** in the menus, or double-click on the column header, to name Column A: Time.

Setting up the model

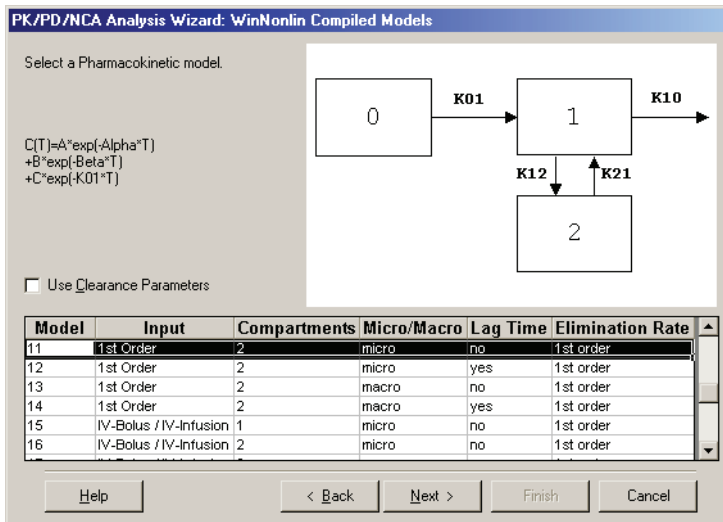
The kinetics will be modeled by a two-compartment open model with first-order input, Model 11 from WinNonlin's compiled pharmacokinetic library.

When simulating, the process of selecting the model (a compiled model or an ASCII model) is exactly the same as it would be when modeling.

To select the PK model:



1. Choose **Tools>PK/PD/NCA Analysis Wizard** from the menus or click the **PK/PD/NCA Analysis Wizard** tool bar button.
2. Select **Pharmacokinetic** in the Model Types dialog, and confirm that the **Compiled** check box is selected. Click **Next**.
3. Select **Model 11**.



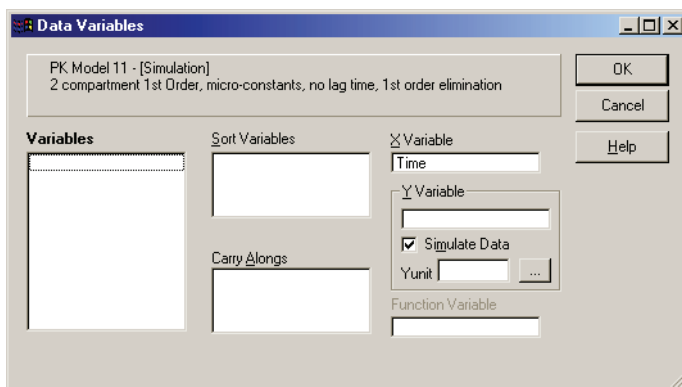
4. Click **Next**. In the final dialog, click **Finish** to load the model.

Data variables

The Simulate Data check box in the Data Variables dialog selects simulation rather than model fitting. The time variable must be identified.

To set model variables:

1. Choose **Model>Data Variables** from the menus.
2. Drag **Time** to the X Variable box.
3. Check the **Simulate Data** check box.



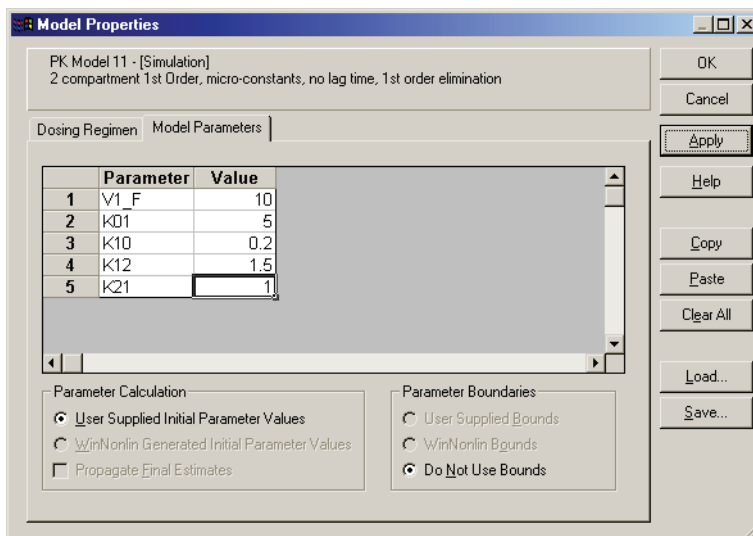
4. Click **OK**.

Model parameters

The simulation requires parameter values, as constants, to calculate output.

To enter parameter values:

1. Select **Model>Model Parameters** from the menus.
2. Enter the following parameter values into the grid as shown below: $V_F = 10$; $K_{10} = 0.2$; $K_{01} = 5$; $K_{12} = 1.5$; $K_{21} = 1$.



3. Click **Apply** (leave the dialog box open).

Dosing regimen

This example will dose 150 mg every 24 hours.

To enter the dosing information:

1. Open the **Dosing Regimen** tab.
2. Enter the values shown in the table below.

Dosing Constant	Value
Number of doses	2
Dose #1	150
Time of dose #1	0
Dose #2	150
Time of dose #2	24

3. Enter mg in the **Current Units** field.

The screenshot shows the 'Model Properties' dialog box with the 'Dosing Regimen' tab selected. The dialog box contains a table for dosing constants and values, and a section for dosing units.

PK Model 11 - [Simulation]
2 compartment 1st Order, micro-constants, no lag time, 1st order elimination

Dosing Regimen | Model Parameters

	Constant	Value
1	Number of Doses	2
2	Dose #1	150
3	Time of Dose #1	0
4	Dose #2	150
5	Time of Dose #2	24

Dosing Units

Current Units: mg [Restore] [Clear] [Add]

Mass Prefix: [] Mass Unit: []

Dose Normalization: none []

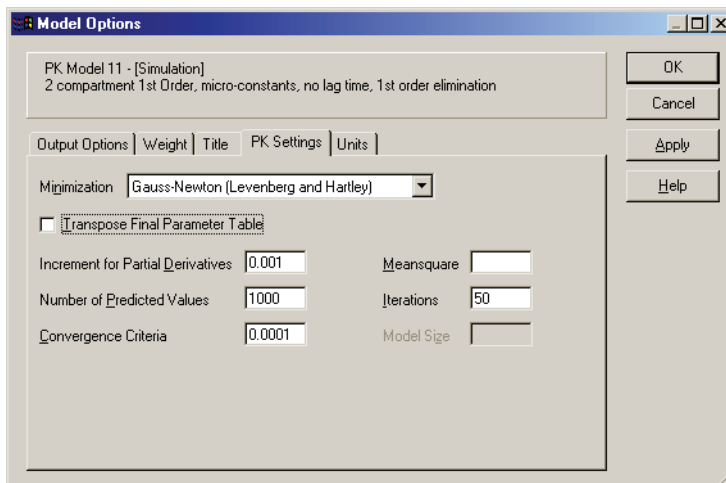
Buttons: OK, Cancel, Apply, Help, Copy, Paste, Clear All, Load..., Save...

4. Click **OK**.

Model options

To specify model options:

1. Click the **Model Options** tool bar button or choose **Model>Model Options** from the menus.
2. Select the PK Settings tab, and uncheck the **Transpose Final Parameter Table** check box. Click **OK** when finished.



Running the simulation

To start the simulation:

1. Click the **Start Modeling** button or choose **Model>Start Modeling** from the menus. The simulated response data appear in the Summary Table worksheet of the PK Workbook, and in the PK Chart output.

PK Workbook - [Untitled49] (Derived)

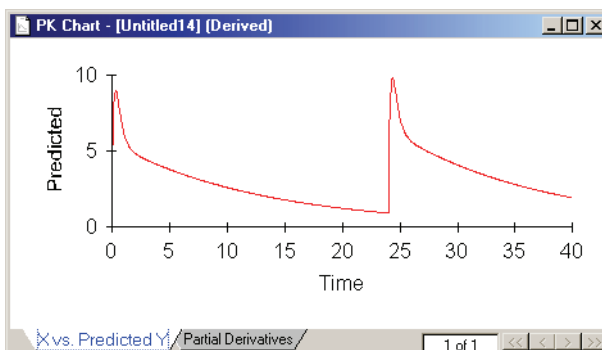
D26 0.0201591690882244

	Time_obs	Time	Predicted	VIF
1	0	0.0000	0.0000	0.0000
2	0.18	0.1800	7.5682	0.3715
3	0.36	0.3600	8.9582	0.2112
4	0.54	0.5400	8.4556	0.2001
5	0.72	0.7200	7.5697	0.1360
6	0.9	0.9000	6.7692	0.1067
7	1.08	1.0800	6.1539	0.1056
8	1.26	1.2600	5.7081	0.0992
9	1.44	1.4400	5.3900	0.0817
10	1.62	1.6200	5.1608	0.0618
11	1.8	1.8000	4.9909	0.0465

Summary Table Partial D

The Variance Inflation Factor (VIF) is useful in determining optimal sampling schedule for parameter estimation. Lower values indicate better estimates.

Note: The secondary parameter estimates in simulation output are based on the first dose.



Comparing dosing schemes

Simulations provide an easy means of modifying a scenario to see the impact on concentrations or effects. For example, re-run the prior simulation, keeping the total daily dose of 150, but this time administering 75 mg every 12 hours.

After the previous simulation (“[Simulation of a compiled library model](#)” on page 413), WinNonlin remembers the values for the Data Variables, Model Parameters, and Dosing Regimen. Only the Dosing Regimen need be edited.

Dosing regimen

For this new simulation, use doses of 75 mg at times 0, 12 and 24.

To enter dosing information:

1. Select **Model>Dosing Regimen** from the menus.
2. Enter the new doses and click **OK**.

Model Properties

PK Model 11 - [Simulation]
2 compartment 1st Order, micro-constants, no lag time, 1st order elimination

Dosing Regimen | Model Parameters

Constant	Value
1 Number of Doses	3
2 Dose #1	75
3 Time of Dose #1	0
4 Dose #2	75
5 Time of Dose #2	12
6 Dose #3	75
7 Time of Dose #3	24

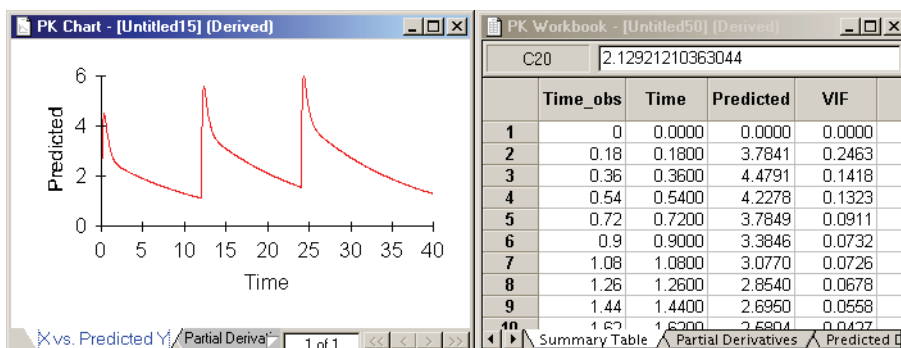
Dosing Units

Current Units: mg Restore Dose Normalization: none

Mass Prefix: Mass Unit: Clear Add

Buttons: OK, Cancel, Apply, Help, Copy, Paste, Clear All, Load..., Save...

3. Choose **Model>Start Modeling** to re-run the simulation.



Scripting

WinNonlin's object-oriented scripting language

The WinNonlin scripting language is an object-based programming language for automating WinNonlin tasks and analyses. Using the scripting language, users can devise script files, which are in essence computer programs, to customize WinNonlin and run a number of WinNonlin operations in batch mode.

This section includes:

- “Terminology” on page 421: object-based programming terminology
- “Writing and editing a script” on page 422
- “Executing a script” on page 424
- “Scripting object properties and methods” on page 425: a quick reference guide to properties and methods for each WinNonlin object and engine
- “Script file components” on page 430
- “Processing multiple profiles in script files” on page 432

See “Scripting Commands” on page 583 for details on each scripting method and property. See the *WinNonlin Examples Guide* for example scripts.

Note: E-mail sales@pharsight.com to learn about WinNonlin automation software and services.

Terminology

Object-oriented programming terms used in WinNonlin scripting follow.

Table 17-1. Scripting terminology

Objects	Object-oriented programming languages deal with objects. In WinNonlin, each type of “child window” that appears in the main WinNonlin parent window is an object. There are five types of objects in WinNonlin: Data objects, Text objects, Model objects, Graph objects, and WinNonlin itself.
Engines	Engines execute tasks in WinNonlin. The WinNonlin engines are: PK, DESC, LIN-MIX, BIOEQ, ANOVA ^a , PLOT, TABLE, TRANSFORM, MERGE and TOOLBOX.
Method	A method is a set of computer code that performs functionality on a specific object or engine. In short, it invokes an action. For example, the Method LOAD causes an associated data file (an Object) to be loaded. The Method RUN causes an associated PK Model (Engine) to run the model.
Property	Every object has properties that define how the object should appear and interact. Properties also set values for objects or engines. In short, properties are attributes or characteristics of an object. For example, a property of an output data object could contain a file name to identify that object. Another property could be the file type to be used when loading the object. A third property would define whether or not the named data file contains data headers (column names) as the first row.
Script File	A Pharsight script file (*.PSC) is an ASCII file that contains WinNonlin scripting code. Within a script, there could be many <i>Objects</i> of each type and several <i>Engines</i> . Only one Engine may be in use at any one time.

a. ANOVA is available for backward compatibility with older versions of WinNonlin. LinMix or Bioequivalence Wizards provide newer and more powerful options than ANOVA.

Writing and editing a script

This section discusses the steps in writing and editing a script. Script files are ASCII text files. They have the default extension of .PSC.

Note: Scripts written with early versions of WinNonlin have the extension *.WSC. They can be opened in this version of WinNonlin, but cannot be saved back to that file type. New script files use the .PSC file extension.

Older script files will run but may require minor modifications to file types. For example, output must be saved to an appropriate file type - a .PWO for a workbook rather than a .WDO.

To write a script:

1. From the **File** menu, select **New** (**Alt+F, N**). The New dialog appears.
2. Select the radio button for **Text**.
3. Click **OK**. A text window appears.
4. Type the content of the script.
5. When the script is complete, select **Save As** from the **File** menu (**Alt+F, A**). The File Save dialog appears.
6. Select the appropriate drive and directory.
7. Select **Script Files (*.PSC)** from the pull-down menu in the **Save as** type box.
8. In the File name box, type the desired file name, using the file extension **.PSC**.
9. Click **Save**.

Comments may be added to the Script file by adding lines to the script that begin with **REMA**, or **REMARK**.

Case does not matter for any of the Properties or Methods.

To edit a script:

1. Select **Open** from the **File** menu (**Alt+F, O**); or click the **Open** button on the tool bar. The File Open dialog appears.
2. Select the appropriate drive and directory if necessary.
3. Select **Script Files (*.PSC)** in the **List Files of Type** box. All script files in the selected directory will be listed.
4. Double click on the name of the script file to be edited. The script file will appear in a text window.
5. In the text window, click the text line where changes are needed, then type in the changes.
6. When done, click **Save**.

Executing a script

Once a script is created, the following steps are used to run it. Script files may be executed either from within WinNonlin or from a Windows command line. When running a script from within WinNonlin, no WinNonlin windows may be open. When executing from the Windows command line, WinNonlin must be closed.

Within WinNonlin

To execute a script while in WinNonlin:

1. Close all WinNonlin windows, including the script.
2. From the **File** menu, select **Run Script (Alt+F, R)**. The Run Script dialog appears.
3. Select the appropriate drive and directory and double click on the script file name or click on the script file name and click **Open**. The script file executes and the output appears in a WinNonlin output window.

From the command line

To execute a script from the command line in Windows XP, Vista, and 7:

1. Exit WinNonlin if it is already running.
2. Click the Windows **Start** button.
3. Select **Run**. A Windows dialog will appear.
4. In the Open box, type WINNONLN, or WINNONLN . EXE along with the appropriate path, followed by the script file name, again including the full file path. For example:

C:\WINNONLN\WINNONLN C:\WINNONLN\EXAMPLES\EXMPL_1.PSC

5. Click **OK**.

Note: When a script file cannot be successfully executed because of an error or errors in the file, a log file is created. A text window will appear showing the contents of that log file. If no errors occur during a script file execution, a log file will not be created. The Pharsight application log will contain appropriate entries for actions taken during the script file execution (exclusions, etc.).

From file manager or explorer

To execute scripts from Windows File Manager or from Explorer:

Script files can be run simply by double clicking on their names in Windows Explorer (Windows XP, Vista, and 7).

To do so, the file extension .PSC must be associated with WINNONLN.EXE.

Scripting object properties and methods

This section lists all properties and methods that are applicable to each WinNonlin object and engine. Refer to this Quick Reference Guide to see which properties and methods are used for each operation. Detail on each command is provided under “[Scripting Commands](#)” on page 583. See the *WinNonlin Examples Guide* for examples of their usage.

Objects

There are five WinNonlin objects: the WinNonlin, Data, Text, Model and Graph objects. A quick reference table of applicable properties and methods is given below for each Object.

Table 17-2. WinNonlin object properties and methods

Arrange	Load	Save
Cascade	MsgBox	TileHorizontal
CloseAll	PrintAll	TileVertical
DeleteFiles	PrintData	Unload
ExportAll	PrintGraph	WorkingDir
FileMask	PrintModel	WindowState

Table 17-2. WinNonlin object properties and methods (continued)

Halt	PrintText	
------	-----------	--

Table 17-3. Data object properties and methods

Append	FileType	RenameCol
AppendFilename	Find	Replace
AppendFileType	Font	Save
AppendSheet	FontSize	SearchArea
AutoFileOptions	Headers	Sort
CaseSensitive	Include	SortVar()
Column	Load	SortVars
ConseqDelim	Missing	StartCol
Copy	MultiGrid	StartRow
Cut	Operation	Tab
Delimiter	Paste	Tolerance
EndCol	Print	Unload
EndRow	Remove	Unit
Exclude	RemoveCol	WindowState
Filename	RemoveRow	With

Table 17-4. Text object properties and methods

Filename	Print	WindowState
FileType	Save	
Load	Unload	

Table 17-5. Model object properties and methods

DoseUnit	LinkFile	Print
Filename	Load	Save
FileType	ParmFile	Unload

Table 17-5. Model object properties and methods (continued)

LamzFile	PartFile	WindowState
----------	----------	-------------

Table 17-6. Graph object properties and methods

Filename	MultiGraph	Unload
FileType	Print	WindowState
FootnoteFont	PrintMulti	XLabelsFont
FootnoteFontSize	PrintMultiAcross	XLabelsFontSize
LegendFont	PrintMultiDown	XTitleFont
LegendFontSize	Save	XTitleFontSize
Load	SaveAll	XUnits
LoadTemplate	SaveAllPrefix	YLabelsFont
MoveFirst	SaveTemplate	YLabelsFontSize
MoveLast	SortLevel	YTitleFont
MoveNext	TitleFont	YTitleFontSize
MovePrevious	TitleFontSize	YUnits
	Uniform	

Engines

The tables below provide a quick reference to the properties and methods applicable to each WinNonlin engine. See “[Scripting Commands](#)” on [page 583](#) for details on individual properties and methods.

Table 17-7. PK engine properties and methods

OutData	OutText	RunWordExport
OutGraph	Run	WordExportFilename

Table 17-8. Descriptive statistics engine properties and methods

Confidence	OutData	SortVars
InData	Percentiles	SummaryVar()

Table 17-8. Descriptive statistics engine properties and methods (continued)

Interval	Run	SummaryVars
	SortVar()	Weight

Table 17-9. ANOVA engine properties and methods^a

Filename	OutData	Run
FileType	OutText	

- a. The former ANOVA functions are now handled by the Linear Mixed Effects Modeling engine. Scripts using the ANOVA engine will continue to work, but it is best that new scripts use the LinMix engine.

Table 17-10. Plot engine properties and methods

AutoSort	OutGraph	UpErr()
DnErr()	Run	XTitle
ErrType	SameErrCol()	XUnits
GroupVar(,)	ShowUnits	XVar()
GroupVars()	SortVar(,)	YTitle
InData()	SortVars()	YUnits
Legend()	Title	YVar
NumData		

Table 17-11. Table engine properties and methods

AggregateVar	IDVars	OutData
CrossVar()	InData()	PageFooter
CrossVars	JoinCrossVar(,)	PageHeader
DefinitionFile	JoinCrossVars	RegVar()
Filename	JoinGroupVar(,)	RegVars
GroupVar()	JoinGroupVars	Run
GroupVarBreak()	JoinIDVar(,)	Stats
GroupVars	JoinIDVars	TableNum

Table 17-11. Table engine properties and methods (continued)

IDVar()	NumData	
----------	---------	--

Table 17-12. Transform engine properties and methods

DestArea	Function	SourceCol
DestCol	InData	
Extra	Run	

Table 17-13. Merge engine properties and methods

CarryData	InData()	Run
IncludeVar(,)	NumData ^a	SortVar(,)
IncludeVars()	OutData	SortVars

a. Must be equal to 2.

Table 17-14. LinMix and Bioequivalence engine properties and methods

Filename	OutData	Run
FileType	OutText	

Table 17-15. Toolbox engine properties and methods^a

Toolbox.function assignments	Applicable properties and methods	
Toolbox.Function=Crossover	InData OutData Run Response SortVar()	SortVars ShowUnits Subject TreatmentA TreatmentB Stacked

Table 17-15. Toolbox engine properties and methods^a (continued)

Toolbox.function assignments	Applicable properties and methods	
Toolbox.Function=Semicompartmental	ConcCol Ecol InData Keo OutData	OutGraph Run SortVar() SortVars TimeCol
Toolbox.Function=Superposition	ConcCol DoseUnit InData InitialDose MaintenanceDose Npoints OutData OutGraph	Run SortVar() SortVars Tau TerminalLower TerminalPhase TerminalUpper TimeCol TimeRepeat
Toolbox.Function=Deconvolution	AlphaTerms() Aterms() Bolus ConcCol DoseUnit Delta InputLag	Npoints NumTerms Run Smoothing SortVar() SortVars TimeCol

- a. The set of properties available to the Toolbox engine depends on the value assigned to the Function property. Function can be assigned one of four values (above) to select the analysis tool. Valid properties are listed for each Function property assignment.

Note: Deconvolution commands from scripts written prior to WinNonlin version 5.1 will function properly in WinNonlin 5.3, but the deconvolution output may differ due to enhancements made in version 5.1.

Script file components

While some operations are specified in Script files completely through the use of the scripting language commands described in this chapter, other, more complex, operations are specified via files that are prepared in advance. This section describes how each engine is implemented in the scripting language.

Table 17-16. Script file operations and methods of implementation

Engine	Method of implementation
Bioequivalence	Bioequivalence analysis, including average, population, and individual bioequivalence models, can be specified in scripts using a bioequivalence command file (*.LML).
Charts	Charts are specified in scripts via scripting language commands.
Compartmental models	Compartmental models are specified in scripts via Pharsight Model Objects (*.PMO). Command files (*.CMD) from prior versions of WinNonlin can also be loaded. To use Command files with different model parameters, etc., for each sort level, see “Processing multiple profiles in script files” on page 432 . Note that although a workbook will be loaded and used for the modeling as a result of either the Command file or Pharsight Model Object, that data file will not be available for any other processing. To use that data set for any other purpose within the script, it will be necessary to load it separately.
Descriptive statistics	Descriptive Statistics are specified in Scripts via scripting language commands that address the descriptive statistics engine. The user specifies which columns are summary variables and a new output data grid is generated.
LinMix	Linear mixed effects modeling, including ANOVA, ANCOVA, and linear regression can be included in scripts using a linear mixed effects command file (*.LML). Note: ANOVA command files (*.ANV) from previous versions of WinNonlin can also be loaded. Note that although a data grid will be loaded and used for the modeling as a result of loading the ANOVA command file (*.ANV), that data file is not available for any other processing because it does not have an alias assigned to it. To use that data set for any other purpose within the script, it will be necessary to load it separately.
Merging data sets	The merging of data sets are specified in Scripts via scripting language commands that use the merge engine properties and methods. Any combination of data from two data sets can be merged and saved as one data set.
Noncompartmental analysis^a	Noncompartmental models are specified in scripts via Pharsight Model Objects (*.PMO). Command files (*.CMD) from prior versions of WinNonlin can also be loaded. To use Command files with different Lambda Z ranges, etc., for each sort level, see “Processing multiple profiles in script files” on page 432 . Note that although a workbook will be loaded and used for the modeling as a result of either the Command file or Pharsight Model Object, that data file will not be available for any other processing because it will not have an alias assigned to it. To use that data set for another purpose, it will be necessary to load it separately.
PKS	Access to data in the Pharsight Knowledgebase Server (PKS) is possible using scripting in WinNonlin. Scripting the PKS is done by adding an additional FileType property to the WinNonlin Data Object. The Load and Save operations on this Data Object then rely on a PKS file pointed to by the Data Object. The PKS file is the same format as the files generated by the Study Definition and Study Mappings dialogs used in the creation of a study in the PKS. The file format is XML and is defined by an embedded DTD.

Table 17-16. Script file operations and methods of implementation (continued)

Engine	Method of implementation
Tables	<p>Tables may be specified in Script files in two different ways: through the use of scripting language commands, or using a table definition file (*.TDF).</p> <p>Scripting language commands can specify:</p> <ul style="list-style-type: none">The Input data set(s)Table template numberVariable mappingsSummary Statistics (all or none)Page Break options <p>Using only the Scripting Language commands loses some flexibility. Summary statistics can be included, but it is not possible to select a subset. Also, default formatting is used.</p> <p>A table definition file contains all of the information needed to create the table except the input data set(s) to be used. A script can call any existing table definition file. For more information on table definition files, see “Chart templates” on page 129</p>
Transformations	<p>Data transformations are specified in Scripts via scripting language commands that access properties and methods of the transform engine.</p>

- a. When writing scripts that will use the NCA Data output, it is important to be aware of the form in which this output will appear. The NCA Data output can appear with the variables (columns) Parameter and Estimate (in addition to the sort and carry-along variables), or, with a separate variable for each of the Final Parameters in the transposed output.

The Final Parameters worksheet will be transposed if NCATRANS is included in the NCA specification or if the user has set up “transposed output” as the WinNonlin default. Both transposed a; this option simply determines which format will appear in the worksheet entitled Final Parameters. See [“Transpose final parameter table” on page 208](#).

Note on saving files

CAUTION: When saving files with the Scripting Language, WinNonlin does not check to see if that file name already exists. It is possible to overwrite existing files in this way.

Processing multiple profiles in script files

Two options are available to process compartmental models and noncompartmental analyses in the Scripting Language:

- Loading a Pharsight model object (*.PMO)

- Loading a Command file (*.CMD) from an earlier version of WinNonlin

When using a Pharsight model object, different values may be used for dosing, parameter values, partial areas or lambda_z range for each sort level automatically. Command files, however, store only one set of values to be used for all sort levels. It is possible, however, to load a command file, then load ASCII files that contain unique information for each sort level. These ASCII files can be created and saved within WinNonlin or a text editor.

The information contained in the files specified by these properties is expected to be in a very specific format. The next section discusses the ASCII file formats that are expected when using the following methods with Model Objects.

- .LAMZFILE for specifying lambda_z range information
- .DOSEFILE for specifying dosing information
- .PARMFILE for specifying parameter values
- .LINKFILE for specifying PK/PD models that require two sets of parameter values
- .PARTFILE for specifying partial area information

The following example script files shows how a command file could be loaded, and the methods used.

```
Set MYMODEL = Model
MYMODEL.Filetype= "ASCII"
MYMODEL.File Name= "Profiles.cmd"
MYMODEL.Load
MYMODEL.Dosefile= "Doseinfo.dat"
MYMODEL.PartFile= "Partinfo.dat"
Pk.OutData= "MYOUTDATA"
REMA Text and graphics output objects may also be named at this time
Pk.Run
```

Note: File references within scripts can use absolute or relative file paths.

Special file formats

This section discusses the required ASCII file formats when using the following methods with an ASCII user model or Model Object.

- LAMZFILE names a file containing lambda_z range information (start and end times for computation of Lambda Z in noncompartmental analysis).
- DOSEFILE for dosing information.
- PARMFILE for initial parameter values and bounds (optional).
- LINKFILE for PK/PD models that require two sets of parameter values.
- PARTFILE for partial area information (start and end times for computation of partial areas under the curve).
- UNITS for preferred parameter units in non-compartmental analysis.

The general structure for all of the ASCII files is given, followed by annotated examples for several.

Note: These files may be created using a text editor or from within WinNonlin, using the Save buttons on the appropriate tabs of the Model Properties or Model Options dialogs.

General structure

Order in file	Contents	Comments
1st	"Checksum or Identifier"	Must be included on the first line of file.
2nd	nType	nType is between 1 to 5 as follows: 1 lamdbaz range information 2 dosing information 3 model parameter values 4 link parameter values 5 partial areas
3rd	nProfiles	nProfiles is a value < 32767 that indicates how many profiles or subsets of values are included.
4th	nRows, nCols	nRows & nCols are integers that indicate how many rows and columns of data are in the file.

Order in file	Contents	Comments
5th- (nCols+4)th record	n, Char	For each column, these values indicate the number of decimal places and data type ("A" for strings or "N" for numeric) of the data, respectively.
Remaining records	Data values of each profile	Data values of column 1, 2, 3, etc. Set(s) of values for each nRow.

Note: When ntype = 1 or 5, nRow = nProfiles (no. of profiles). When ntype = 2 or 3 or 4, nRow = NCON (no. of constants) or NPAR (no. of parameters).

Lambda Z range information (nType=1)

Data Grid with Lambda Z Range Information for three profiles. Model used: NCA Model 200.

Subject	Start time	End time	Exclusions	
1	3	18	4	5
2	3	18	3	
3	3	18		

ASCII File with Lambda_z Range information of the above three profiles.

Table 17-17. Lambda Z file structure

Record	Contents	Comments
1 st	Checksum or Identifier	Required
2 nd	1	nType = 1, as 1 represents lambda_z range information.
3 rd	3	nProfiles = 3, as the number of profiles (Subjects) used is 3.
4 th	3, 4	When nType is 1, the number of rows equals nProfiles (the number of profiles), so nRows = 3. When nType is 1, the number of columns equals 2 + the maximum number of exclusions, so nCols = 4.
5 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 1st column.

Table 17-17. Lambda Z file structure (continued)

Record	Contents	Comments
6 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 2nd column.
7 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 3rd column.
8 th	3	Numeric data in the 1st column (Start Time), 1st profile.
9 th	18	Numeric data in the 2nd column (EndTime), 1st profile.
10 th	4	Numeric data in the 3rd column (Exclusions), 1st profile.
11 th	5	Numeric data in the 4th column (Exclusions), 1st profile.
12 th	3	Numeric data in the 1st column (Start Time), 2nd profile.
13 th	18	Numeric data in the 2nd column (EndTime), 2nd profile.
14 th	3	Numeric data in the 3rd column (Exclusions), 2nd profile.
15 th	0	Invalid exclusions in the 4th column, 2nd profile. (See Note below).
16 th	3	Numeric data in the 1st column (Start Time), 3rd profile.
17 th	18	Numeric data in the 2nd column (EndTime), 3rd profile.
18 th	-1e150	Invalid exclusions in the 3rd column, 3rd profile. (See Note below).
19 th	0	Invalid exclusions in the 4th column, 3rd profile. (See Note below).

Note: Values that are in the excluded part of the data grid but are not valid exclusions have a value of 0. This does place the restriction that Time=0 may not be an excluded data point.

Dosing information (nType = 2)

Data Grid with Dosing Information for three profiles. Model used: NCA Model 200.

Subject	Constant	Value
1	Dose	100
1	Time of Last Dose	0

Subject	Constant	Value
2	Dose	100
2	Time of Last Dose	0
3	Dose	100
3	Time of Last Dose	0

ASCII File with dosing information of the above three profiles.

Table 17-18. Dosing file structure

Record	Contents	Comments
1 st	Checksum or Identifier	Was not used before WinNonlin version 3.2. Can be used to save dosing units. See note below ^a .
2 nd	2	nType = 2, as 2 represents dosing information.
3 rd	3	nProfiles = 3, as the number of profiles (Subjects) used is 3.
4 th	3, 1	When nType is 2, the number of rows equals NCON (the number of constants), so nRows = 3. When nType is 2, the number of columns equals 1, so nCols = 1. (See Note below on nRows).
5 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the column.
6 th	100	Numeric data in the 1st column (Value), 1st profile.
7 th	0	Null data in the 2nd column, 1st profile.
8 th	0	Null data in the 3rd column, 1st profile.
9 th	100	Numeric data in the 1st column (Value), 2nd profile.
10 th	0	Null data in the 2nd column, 2nd profile.
11 th	0	Null data in the 3rd column, 2nd profile.
12 th	100	Numeric data in the 1st column (Value), 3rd profile.
13 th	0	Null data in the 2nd column, 3rd profile.
14 th	0	Null data in the 3rd column, 3rd profile.

a. It is possible to save dosing units and dose normalization information to ASCII *.DAT files. To do so, the format for the first line becomes V32 , sdoseunit , sdosenormalization. The commas must be included, even without the units [hence it could be V32 , ,].

Note: When using the Model.DoseFile method with NCA models, nRows is equal to NCON+1 in order to accommodate the Time of Last Dose values. When steady state is being calculated, nRows is equal to NCON+2 to accommodate the Tau values. If steady state is not being calculated, these values should be set to 0.

Model parameter values (nType = 3)

Data grid with parameter values on two profiles. Model used: PK Model 3

Subject	Parameter	Initial value	Lower	Upper
1	Volume_F	0.2	0	0
1	K01	20	0	0
1	K10	2	0	0
2	Volume_F	0.15	0	0
2	K01	33	0	0
2	K10	1.5	0	0

ASCII File with parameter values of the above two profiles.

Table 17-19. Model parameter file structure

Record	Contents	Comments
1 st	Checksum or Identifier	Unused
2 nd	3	nType = 3, as 3 represents parameter values.
3 rd	2	nProfiles = 2, as the number of profiles (Subjects) used is 2.
4 th	3, 3	When nType is 3, the number of rows equals NPAR (the number of parameters), so nRows = 3. When nType is 3, the number of columns equals 3, so nCols = 3. (See Note below on nCol).
5 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 1st column (Initial Value).
6 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 2nd column (Lower Limit).

Table 17-19. Model parameter file structure (continued)

Record	Contents	Comments
7 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 3rd column (Upper Limit).
8 th	0.2	Numeric data of the Volume/F parameter in the 1st column (Initial Value), 1st profile.
9 th	0	Null data of the Volume/F parameter in the 2nd column (Lower Limit), 1st profile.
10 th	0	Null data of the Volume/F parameter in the 3rd column (Upper Limit), 1st profile.
11 th	20	Numeric data of the K01 parameter in the 1st column, 1st profile.
12 th	0	Null data of the K01 parameter in the 2nd column, 1st profile.
13 th	0	Null data of the K01 parameter in the 3rd column, 1st profile.
14 th	2	Numeric data of the K10 parameter in the 1st column, 1st profile.
15 th	0	Null data of the K10 parameter in the 2nd column, 1st profile.
16 th	0	Null data of the K10 parameter in the 3rd column, 1st profile.
17 th	.15	Numeric data of the Volume/F parameter in the 1st column (Initial Value), 2nd profile.
18 th	0	Null data of the Volume/F parameter in the 2nd column (Lower Limit), 2nd profile.
19 th	0	Null data of the Volume/F parameter in the 3rd column (Upper Limit), 2nd profile.
20 th	33	Numeric data of the K01 parameter in the 1st column (Initial Value), 2nd profile.
21 st	0	Null data of the K01 parameter in the 2nd column (Lower Limit), 2nd profile.
22 nd	0	Null data of the K01 parameter in the 3rd column (Upper Limit), 2nd profile.
23 rd	1.5	Numeric data of the K10 parameter in the 1st column (Initial Value), 2nd profile.
24 th	0	Null data of the K10 parameter in the 2nd column (Lower Limit), 2nd profile.
25 th	0	Null data of the K10 parameter in the 3rd column (Upper Limit), 2nd profile.

Note: The three columns: nCol1 = Initial Value, nCol2 = Lower Limit, nCol3 = Upper Limit, are not always used, but must still be part of the file.

Link parameter values (nType = 4)

Since a PK/PD model actually requires two sets of parameter values, the Linkfile is included.

Data grids with link parameter values on two profiles. Models used: PK Model 3/PD Model 105

The following data grid is applied to the PK Model:

Subject	Parameter	Initial Value	Lower	Upper
1	Volume_F	0.2	0	0
1	K01	20	0	0
1	K10	2	0	0
2	Volume_F	0.15	0	0
2	K01	33	0	0
2	K10	1.5	0	0

ASCII File with parameter values of the above two profiles.

Table 17-20. Link PK parameter values

Record	Contents	Comments
1 st	Checksum or Identifier	Was not used before WinNonlin version 3.2. Can be used to save PK concentration units. See note below ^a .
2 nd	4	nType = 4, as 4 represents Link parameter values.
3 rd	2	nProfiles = 2, as the number of profiles (Subjects) used is 2.
4 th	3, 3	When nType is 4, the number of rows equals NPAR (the number of parameters), so nRows = 3. When nType is 4, the number of columns equals 3, so nCols = 3. (See Note below on nCol).
5 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 1st column (Initial Value).

Table 17-20. Link PK parameter values (continued)

Record	Contents	Comments
6 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 2nd column (Lower Limit).
7 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 3rd column (Upper Limit).
8 th	0.2	Numeric data of the Volume/F parameter in the 1st column (Initial Value), 1st profile.
9 th	0	Null data of the Volume/F parameter in the 2nd column (Lower Limit), 1st profile.
10 th	0	Null data of the Volume/F parameter in the 3rd column (Upper Limit), 1st profile.

- a. It is possible to save PK concentration units for the PK tab of PK/PD models to ASCII .DAT files. To do so, the format for the first line becomes V32 , PKconcentrationunit. The comma must be included, even without the units (hence it could be V32 ,).

The remainder of the file structure is the same as the file shown in Example 3.

The following data grid is applied to the PD Model:

Subject	Parameter	Initial value	Lower	Upper
1	E _{max}	5	0	0
1	E _{ce50}	6	0	0
1	Gamma	7	0	0
2	E _{max}	1	0	0
2	E _{ce50}	2	0	0
2	Gamma	3	0	0

ASCII File with parameter values of the above two profiles.

Table 17-21. PD parameter values

Record	Contents	Comments
1 st	Checksum or Identifier	Unused

Table 17-21. PD parameter values (continued)

Record	Contents	Comments
2 nd	4	nType = 4, as 4 represents Link parameter values.
3 rd	2	nProfiles = 2, as the number of profiles (Subjects) used is 2.
4 th	3, 3	When nType is 4, the number of rows equals NPAR (the number of parameters), so nRows = 3. When nType is 4, the number of columns equals 3, so nCols = 3. (See Note below on nCol.)
5 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 1st column (Initial Value).
6 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 2nd column (Lower Limit).
7 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 3rd column (Upper Limit).
8 th	5	Numeric data, Emax parameter in the 1st column (Initial Value), 1st profile.
9 th	0	Null data, Emax parameter in the 2nd column (Lower Limit), 1st profile.
10 th	0	Null data, Emax parameter in the 3rd column (Upper Limit), 1st profile.
11 th	6	Numeric data, Ece50 parameter in the 1st column, 1st profile.
12 th	0	Null data, Ece50 parameter in the 2nd column, 1st profile.
13 th	0	Null data, Ece50 parameter in the 3rd column, 1st profile.
14 th	7	Numeric data, Gamma parameter in the 1st column, 1st profile.
15 th	0	Null data, Gamma parameter in the 2nd column, 1st profile.
16 th	0	Null data, Gamma parameter in the 3rd column, 1st profile.
17 th	1	Numeric data, Emax parameter in the 1st column (Initial Value), 2nd profile.
18 th	0	Null data, Emax parameter in the 2nd column (Lower Limit), 2nd profile.
19 th	0	Null data, Emax parameter in the 3rd column (Upper Limit), 2nd profile.
20 th	2	Numeric data, Ece50 parameter in the 1st column, 2nd profile.
21 st	0	Null data, Ece50 parameter in the 2nd column, 2nd profile.

Table 17-21. PD parameter values (continued)

Record	Contents	Comments
22 nd	0	Null data, Ece50 parameter in the 3rd column, 2nd profile.
23 rd	3	Numeric data, Gamma parameter in the 1st column, 2nd profile.
24 th	0	Null data, Gamma parameter in the 2nd column, 2nd profile.
25 th	0	Null data, Gamma parameter in the 3rd column, 2nd profile.

Note: The three columns: nCol1 = Initial Value, nCol2 = Lower Limit, nCol3 = Upper Limit, are not always used, but must still be part of the file.

Partial areas (nType = 5)

Data grid with partial area information for three profiles. Model used: NCA Model 200

Subject	Min Time	Max Time	AUC1 Lower	AUC1 Upper
1	.25	48	5	6
2	.25	48	8	12
3	.25	48	2	3

ASCII File with partial areas information of the above three profiles.

Table 17-22. Partial areas file format

Record	Contents	Comments
1 st	Checksum or Identifier	Unused
2 nd	5	nType = 5, as 5 represents Partial Areas.
3 rd	3	nProfiles = 3, as the number of profiles (Subjects) used is 3.
4 th	3, 2	When nType is 5, the number of rows equals nProfiles, so nRows = 3. When nType is 5, the number of columns equals the maximum number of partial areas multiplied by 2, so nCols = 2.
5 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 1st column (AUC1 Lower).

Table 17-22. Partial areas file format (continued)

Record	Contents	Comments
6 th	4, N	n = 4, as the number of decimal places, if any, is 4. Char = N as the data type is numeric for the 2nd column (AUC1 Upper).
7 th	5	Numeric data of the column AUC1 Lower of the 1st profile.
8 th	6	Numeric data of the column AUC1 Upper of the 1st profile.
9 th	8	Numeric data of the column AUC1 Lower of the 2nd profile.
10 th	12	Numeric data of the column AUC1 Upper of the 2nd profile.
11 th	2	Numeric data of the column AUC1 Lower of the 3rd profile.
12 th	3	Numeric data of the column AUC1 Upper of the 3rd profile.

Note: Values that are in the partial area portion of the data grid, but are not valid partial areas, should be given lower and upper values of 0.

Scripting the Pharsight Knowledgebase Server

The Pharsight Knowledgebase Server (PKS), and the files stored as studies and scenarios in it, can be accessed by scripting from WinNonlin. Scripting the PKS is done by adding an additional FileType property to the Data Object. The Load and Save operations on this Data Object will rely on a PKS file pointed to by the Data Object. This PKS file is the same format as the files generated by the Study Definition and Study Mappings dialogs used in the creation of a study in the PKS. The file format is XML and is defined by an embedded DTD. See “[PKS file format](#)” on page 445 for details.

See the *WinNonlin Examples Guide* for examples of scripts to run WinNonlin/PKS operations.

The workflow rules associated with the manipulation of PKS data via Scripting can be encapsulated based on the method being invoked (i.e. Load or Save).

Load

Note: Rules follow the user interface.

1. If the PKS file pointed to by the Data Object has only a study name, then only a study will be loaded.
2. If the PKS file pointed to by the Data Object has a study name and a scenario name, then both the study and latest version of the scenario will be loaded. Scripting will not load previous versions of a scenario.

Save

Note: A more involved set of rules based on the state of the application.

1. If no PKS session has been created (i.e. no connection has been made to the PKS), then the Save Method will create a study in the PKS based on the information in the PKS file pointed to by the File Name Property of the Data Object.
2. If a PKS session has been created via loading a study only and no other Objects (i.e. charts, texts, workbooks) are present in WinNonlin, then the Save Method will update the Study only.
3. If a PKS session has been created via loading a study and scenario or a study only and other Objects (i.e. charts, texts, workbooks) are present in WinNonlin, then the Save Method will update the Study (if it changed) and Save the scenario to a new version or an entirely new scenario. A new version will be created if and only if the scenario name being saved is the same as the scenario that was loaded.

PKS file format

The PKS file contains XML that describes load and/or save operations to and/or from the PKS. It is saved as an XML file with the following format.

```
<?xml version="1.0"?>
<!DOCTYPE PWRImport
[
<!ELEMENT PWRImport (StudyInformation?, Scenario?
,Mappings?)>
<!ATTLIST PWRImport
  UserID CDATA #IMPLIED
  Password CDATA #IMPLIED
  Server CDATA #IMPLIED
```

```
Port CDATA #IMPLIED
Root CDATA #IMPLIED
```

```
>
```

```
<!--ELEMENT StudyInformation EMPTY-->
```

```
<!--ATTLIST StudyInformation
```

```
BlindingType CDATA #IMPLIED
```

```
Compound CDATA #IMPLIED
```

```
Description CDATA #IMPLIED
```

```
Design CDATA #IMPLIED
```

```
EndTime CDATA #IMPLIED
```

```
Indication CDATA #IMPLIED
```

```
Name CDATA #IMPLIED
```

```
Portfolio CDATA #IMPLIED
```

```
Project CDATA #IMPLIED
```

```
StartTime CDATA #IMPLIED
```

```
State CDATA #IMPLIED
```

```
Type CDATA #IMPLIED
```

```
Reason CDATA #IMPLIED
```

```
>
```

```
<!--ELEMENT Scenario EMPTY-->
```

```
<!--ATTLIST Scenario
```

```
Name CDATA #IMPLIED
```

```
Reason CDATA #IMPLIED
```

```
GetLatest (TRUE|FALSE) #IMPLIED
```

```
>
```

```
<!--ELEMENT Mappings (Sources?, Mapping*)-->
```

```
<!--ATTLIST Mappings
```

```
Type (SAMPLE|DOSE|BOTH) #IMPLIED
```

```
>
```

```
<!--ELEMENT Sources EMPTY-->
```

```
<!--ATTLIST Sources
```

```
Observation CDATA #IMPLIED
```

```
Dose CDATA #IMPLIED
```

```
Demographics CDATA #IMPLIED
```

```
>
```

```
<!--ELEMENT Mapping (FromFields*, FieldOption*)-->
<!--ATTLIST Mapping
  FromField    CDATA #IMPLIED
  ToField      CDATA #IMPLIED
  FieldType    (Subject_ID|Subject_Data|Observa-
tion|DCP_Description|Actual_Clock_Time|Relative_Nomina
l_Time|Sample_Number|Relative_Actual_Time|Relative_Ac
tual_End_Time|Treatment|Period|Phase|Undefined|Dose)
#IMPLIED
  TreatAsDCP   (TRUE|FALSE) #IMPLIED
-->

<!--ELEMENT FromFields EMPTY-->
<!--ATTLIST FromFields
  Name    CDATA #IMPLIED
-->

<!--ELEMENT FieldOption EMPTY-->
<!--ATTLIST FieldOption
  OptionName
(Unit|Sample_Description|Sample_Matrix|Analytical_Metho
d|Status|LLOQ|ULOQ) #IMPLIED
  IsColumn   (TRUE|FALSE) #IMPLIED
  Value      CDATA #IMPLIED
-->

]
-->
```


Using the Pharsight Knowledgebase Server

Data access in a secure environment

The Pharsight Knowledgebase Server (PKS) is a database system for secure storage of study data and analyses. The PKS provides a secure data management system with complete auditing capabilities. The combination of the PKS and WinNonlin Enterprise provides an effective means to share source data, models, scripts and results among the members of a drug development team. The combination also supports compliance with the U.S. Food and Drug Administrations's 21 CFR Part 11 regulation.

This chapter covers the use of WinNonlin as a PKS client: loading PKS data to WinNonlin and saving data from WinNonlin as PKS studies and scenarios. The PKS can also be accessed by way of a web interface, which provides additional operations, and an XML API. These functions and operations are covered in the Pharsight Knowledgebase Server's documentation. This chapter covers WinNonlin/PKS interactions under the following headings.

- [“The PKS information structure” on page 450](#)
- [“Dosing” on page 452](#)
- [“Accessing the PKS from WinNonlin” on page 453](#)
- [“Creating a study” on page 456](#)
- [“Loading a study or scenario” on page 467](#)
- [“Creating scenarios” on page 469](#)
- [“Optimizing PKS/WNL performance” on page 471](#)
- [“Change tracking and audit information” on page 474](#)
- [“Dosing information” on page 476](#)
- [“Appending or updating a study” on page 478](#)

- “PKS libraries and object shares” on page 479
- “PKS Information” on page 483
- “WinNonlin and PKS differences” on page 484

See also “Scripting the Pharsight Knowledgebase Server” on page 444.

The PKS information structure

WinNonlin is an object (text, chart, model, data) based application. Those objects fit into specific niches in the PKS data structure. PKS saves observations and related data as a PKS *study*. Analysis settings and output are added to the study as *scenarios*. Non-WinNonlin documents, such as analysis code files, reports and graphics that are related to the study, can be loaded into the PKS and associated with a scenario using WinNonlin’s “Other documents” feature.

Studies

The basic unit of information in the PKS is a *study*. A study starts with a collection of raw data that must include longitudinal observations, observation times and subject identifiers, and may include dosing and subject demographics. As analyses are performed on these data, the analyses, including model settings and results, can be added as scenario(s) within the study.

A PKS study requires specific fields. First, a study must include at least one column containing subject identifiers. A collection of fields taken together can be used for subject identification (e.g., first name, last name).

A study may include (but is not required to have) multiple columns of *subject data*—time-invariant, subject-specific information such as body weight or gender, i.e., baseline covariates.

A study may include time-dependent observation and dosing data. Observation data may include records of concentration, measurements of blood pressure, heart rate, etc. As the PKS manuals explain, each data collection point can have multiple observations. In this case, a sampling variable is used to differentiate the samples.

If the observation data include multiple, concurrent measurements for any subject (e.g., multiple assays performed on a given sample or multiple sam-

ples during a given visit), the data set must include a variable that differentiates the measurements (e.g., values 1, 2, 3 for the 1st, 2nd and 3rd samples).

Time fields

Because a study generally includes time-dependent observation data, a study must include *relative nominal time*, indicating the times at which observations (and doses, if included) were scheduled in the protocol. This is in contrast to *relative actual time*, representing the times that observations or doses really happened. (*Relative time* is time elapsed since the start of the study, period or phase.) If the data uses infusion dosing, the *relative nominal end time* must be identified as well. If only the Relative Nominal Time is defined in the data set, a duplicate column will be created for the actual times.

Note: To create a study that does include longitudinal observation data, include a dummy time column, with values for each row (as placeholders), in the study.

Study metadata

When a study is created, information about the study can be entered in the Study Definition dialog (see below). This dialog includes information about the study: Study Name, Portfolio, Project, Indication, Study Type, Study Design, Compound, etc. This metadata provides ways to search and filter the data within the database. The metadata are saved with the study. See [“Creating a study” on page 456](#) and [“Loading a study or scenario” on page 467](#).

Scenario

When WinNonlin analysis is performed on a PKS study, the combination of data, model, and (optional) results data can then be saved in the PKS as a *scenario* of the original study. A *scenario* can include a structural model, initial parameters, and/or statistical tests to be applied to a particular data set for fitting or analysis, as well as any derived output from those tests or analysis. Scenarios are roughly comparable to the WinNonlin concept of a workspace in that it is possible to save, in one place and under one name, related work—the data source, a model with model parameters, dosing, other model settings, and the analytical output. Scenarios are naturally associated with (i.e., are children of) a particular data set (study), although it is possible to create further scenarios from the study (more children) or from an existing scenario (grandchildren, etc.). See also [“Creating scenarios” on page 469](#).

Other documents

The PKS can store documents that are not related to WinNonlin, such as study reports, SAS or NONMEM code files, or a graphic created by saving a WinNonlin chart to a JPEG file. Sets of such documents can be associated with scenarios. See [“Adding non-WinNonlin documents to a scenario” on page 470](#) for instructions.

Dosing

Dosing information is a special case of study data. Dosing is generally saved as part of a PKS study, though it is not required. Dose data can be added to a study by one of three methods.

- [Dosing worksheet](#): load dosing during study creation, from a separate worksheet (not the worksheet containing observation data)
- [Append dosing](#) after study creation, from a dosing worksheet
- [Dosing dialog](#): enter dosing as part of a modeling scenario, using the WinNonlin Dosing Regimen dialog.

If multiple scenarios are included in a study, each may use different dose data. See [“Dosing information” on page 476](#) for details.

Dosing worksheet

It is possible to include dosing data when creating a study in the PKS. The dosing information must appear in a different worksheet from that containing the study observation data. That worksheet is selected as the dose source during study creation, and its variables are assigned as part of mapping the study variables (see [“Creating a study” on page 456](#)). The worksheet must contain, at minimum, subject identifiers, dose amounts, and relative nominal dose times. It must contain the sort variables required to distinguish profiles. The columns representing subject identifiers and nominal time must be named exactly as the corresponding columns are in the workbook containing the observation data. The PKS will use those columns to match observation data to dose data for given subject(s).

The dosing worksheet can contain columns for more than one analysis scenario. Users can select which dose columns to use for a given scenario, as detailed under [“Dosing information” on page 476](#).

Append dosing

To append dosing data to a PKS study, close all WinNonlin windows, then open a workbook containing the dose information in a single worksheet. The worksheet must contain, at minimum, subject identifiers, dose amounts, and relative nominal dose times. The PKS will use the subject and time columns to match dose data to records for specific subjects at specific times. Use the PKS Append/Update feature to map the new data columns to PKS study fields.

Dosing dialog

When a PKS study or scenario with no dosing data is loaded in WinNonlin, and a model is selected, then dosing data can be entered via the WinNonlin Dosing Regimen or Dosing Data Variable dialog. When the scenario is saved to the PKS with model, model parameters and dosing information, the dosing is transferred to a Dosing worksheet inside the study workbook. Thereafter, any changes to the dosing then must be done on the Dosing worksheet rather than through the WinNonlin Dosing Regimen. The WinNonlin dosing dialog can still be used for simulations and for user models. Dosing entered for a simulation, and subsequently saved to a scenario, resides with the scenario information, and does not become the study dosing data.

Accessing the PKS from WinNonlin

Access to the Pharsight Knowledgebase Server occurs via the PKS menu in WinNonlin Enterprise. The menu items include:

Table 18-1. PKS menu

Menu item	Used when	Function
Load	Always	Closes the current PKS session, if any, and loads a PKS study or scenario
Save	A PKS study is loaded	Saves changes to the study and creates a new version of the current scenario, if any
Save As	A PKS study and scenario are loaded	Saves changes to the study, if any, and creates a new scenario
Create Study	No PKS session is open and a workbook is loaded	Opens the mapping wizard to set up a new PKS study from the current workbook

Table 18-1. PKS menu (continued)

Menu item	Used when	Function
Append/Update Study	No PKS session is open and a workbook is loaded	Opens the mapping wizard, so that data columns from the current workbook can be added to an existing PKS study
Dosing	A PKS study or scenario is loaded	Sets dosing column(s) to be used in the current model or analysis (scenario)
Add columns	A PKS study is loaded	Loads additional study data columns from PKS into WinNonlin
Other documents	Always	Opens the WinNonlin Document Manager, to save or load non-WinNonlin documents to or from a PKS scenario
Share	Always	Loads a non-WinNonlin document ("other document") from any PKS scenario for use in the current analysis; share link can be saved with a scenario
Libraries	Always	Loads or creates a PKS system object, storing a binary file containing code, parameter names, and other items to be used as standards across many studies and analyses
Status	A library or share item is loaded	Checks whether any currently-loaded library or share items are up to date with the most recent version in PKS
Information	A PKS study is open	Displays the PKS study identifier and scenario identifier, if one is loaded

PKS sessions

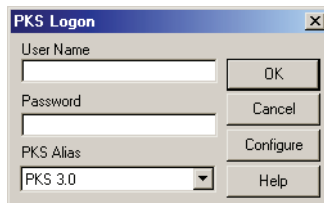
The Enterprise edition of WinNonlin has a concept of a *PKS session*. A PKS session begins when WinNonlin retrieves and uses data from the PKS. While the link between WinNonlin and the PKS is not constant. WinNonlin knows whether data came from the PKS, and tracks all changes. If data are then saved back to PKS (whether to an existing study or scenario or to a new one), any changes are documented for audit trail purposes.

Connecting to the PKS

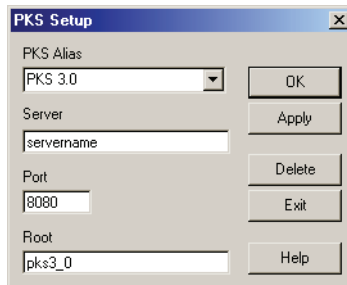
The PKS configuration must be set up before WinNonlin can contact the PKS. The configuration settings are accessed from the PKS Logon dialog. This information need be entered once only; WinNonlin will save it.

To configure the PKS connection:

1. Select any available item in the PKS menu to open the PKS Logon dialog. For example, choose **PKS>Load...**



2. Click the **Configure** button to edit the PKS setup.



3. Enter a name describing the PKS database under PKS Alias. This will be used to select a specific PKS database instance when logging in, as more than one PKS database may be available on the local network. For example, many sites have one production database and one example database, used to run examples from *PKS Examples Guide*.
4. Enter the PKS server machine name under Server. Similarly, enter the port number under Port, and root directory for the PKS database under Root.

Note: The system administrator who set up the PKS database should provide the values for the server, port and root.

5. Click **OK**.

Logging in

User name and password

Each time data are loaded to or from the PKS, the user must log in and provide a password. The log in user name and password are provided by your PKS system administrator, who must set up an account for each user to access the PKS (see the *PKS User's Guide* for details).

Creating a study

Studies are the fundamental unit of PKS organization. A PKS study stores raw data to be modeled and analyzed in one or more scenarios. See [“Studies” on page 450](#) for further discussion.

Note: See [“Dosing” on page 452](#) for methods to add dosing information to a study.

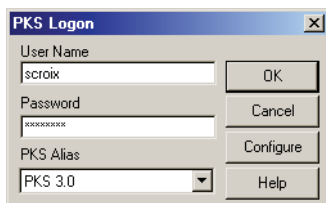
To create a study:

1. Open sample data in a WinNonlin workbook. If dosing and/or demographics data are to be loaded, open those as well.

Data requirements: Observation data must contain, at minimum, columns for subject identifiers, sample values and nominal (scheduled) times. Dosing must be in a separate worksheet from observations, and must contain the same data collection point identifiers, i.e., subject identifiers, time fields, and indicators of period, phase, etc. Demographic data must be time-invariant and subject-specific, and can be loaded from the same worksheet as the observations.

2. Choose **PKS>Create Study** from the WinNonlin menus.

If logging in for the first time, set up the PKS configuration. See [“Connecting to the PKS” on page 455](#).



3. Enter the user name and password supplied by your PKS administrator. These are not necessarily the same as those used to log onto your Windows network. You must have WRITE system access in order to create studies.
4. Select the desired PKS database under PKS Alias and click **OK**.

WinNonlin will contact the PKS and open the [Study definition](#) dialog.

Study definition

[Creating a study](#) requires specification of the following information. (Required fields are marked with an asterisk in the Study Definition dialog.) These attributes are called *study metadata*. They are useful in database searches for studies on a given compound, indication, etc. See your database administrator for local rules for metadata values.

Table 18-2. Study metadata fields

Attribute	Description	Data entry	Format
Study name ^a	Used to identify and locate study in PKS; must be unique within the PKS database	Typed	50 character max
Study start time and end time	Dates of first and last days in the study	Typed	Month day, year, e.g.: May 5, 2002 or mm/dd/yyyy, e.g.: 05/05/2002

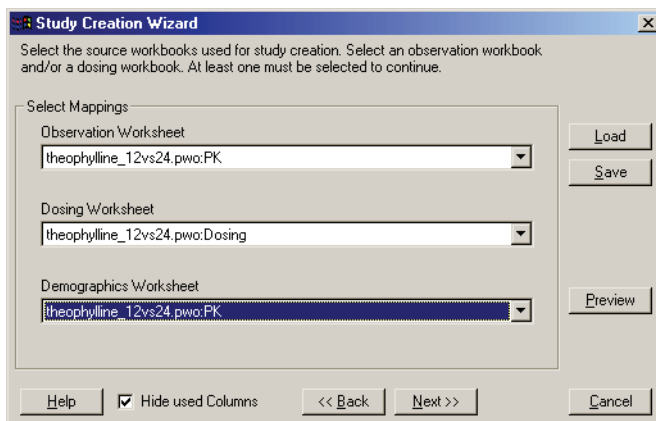
Table 18-2. Study metadata fields (continued)

Attribute	Description	Data entry	Format
Description	Text describing the study; use Ctrl+Enter for line breaks	Typed	2000 character max
Portfolio	Development portfolio to which the study belongs	Typed or selected from list of prior values	50 character max
Project	Development project to which the study belongs	Typed or selected from list of prior values	100 character max
Indication	Primary indication(s) for the study	Typed or selected from list of prior values	50 character max
Blinding type ^a	blinding level for the study	Typed or selected from list of prior values	30 character max
Study type ^a	type of study	Typed or selected from list of prior values	30 character max
Study design ^a	Experimental structure, e.g., parallel or crossover	Typed or selected from list of prior values	30 character max
Compound	Main drug compound(s) under development	Typed or selected from list of prior values	50 character max
Status	Whether the study will be locked against changes or not	Selected	Normal or locked

a. Required field.

1. Click **OK** to proceed to the **Study Creation Wizard**.

Study Creation Wizard

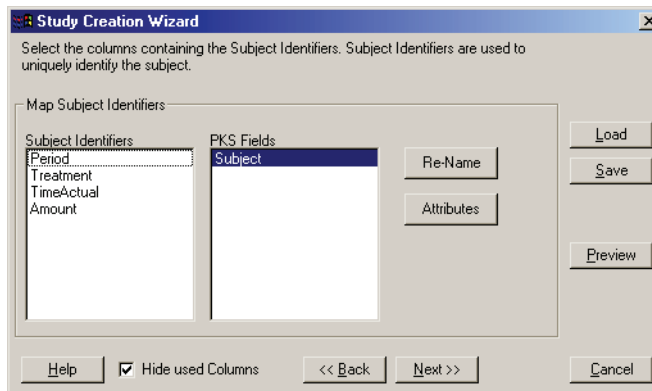


1. Select the worksheet for the observation data under Observation Worksheet.
2. Select dosing and demographics worksheets (optional).

Note: The Save button saves all study creation settings, including file paths/names and variable mappings, as an XML file with a *.PKS file extension. The Load button reloads the settings from that file.

3. Click **Next** to proceed to the [Subject identifiers](#).

Subject identifiers



1. Drag the variable(s) that identify individual subjects to the PK Fields box.

Note: If dosing and/or demographics worksheets are used, only columns that appear in all worksheets, named identically, will appear in the Subject Identifiers list.

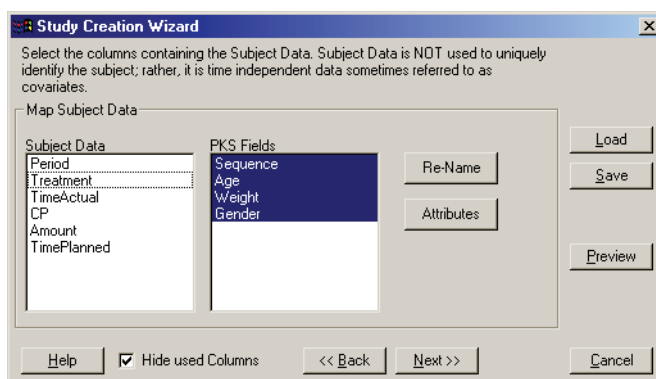
2. Select a variable under PKS Fields and click the **Attributes** button to set units for the subject ID variable(s), if needed, as detailed under “[Subject data attributes](#)” below.
3. Click **Next** to map [Subject data](#).

Subject ID attributes

If units already exist for a given column in the WinNonlin worksheet, those units are loaded to the PKS. To view them, or to assign units to a column:

1. To load units for each record from a column in the data set, drag that column to the Unit field.
2. To set one unit for an entire column, check **Static** and enter or select the unit under Unit. Time units must come from a fixed, controlled list. Other units can be any string, but if a unit is not defined in the PKS, unit conversions will not be available for it.
3. Click the **Hide** button to return to the [Subject identifiers](#).

Subject data



1. Drag columns that contain subject demographics (baseline covariates) and other time-invariant, subject-specific columns to the PKS Fields box.

CAUTION: Improper variable mappings can slow PKS operations substantially. In particular, mapping subject-specific data as observations slows performance. Observations include one record for every data collection point, rather than one record per subject. See also [“Optimizing PKS/WNL performance” on page 471](#) and [“Performance Tips” in the PKS System Administrator’s Guide](#)

2. To enter a new name for a column, to be used in the PKS study, select that column under PKS Fields and click the **Rename** button. The name may include up to 50 characters, and must follow WinNonlin column name limits.
3. Select the variable under PKS Fields and click the **Attributes** button to set units, as detailed under [“Subject data attributes”](#) below.
4. Click **Next** to specify variables related to [Time](#).

Subject data attributes

If units already exist for a given column in the WinNonlin worksheet, those units are loaded to the PKS. To view them, or to assign units to a column:

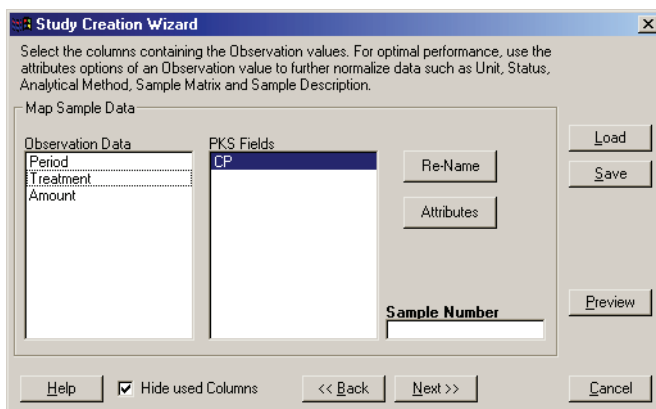
1. To load units for each record from a column in the data set, drag that column to the Unit field.
2. To set one unit for an entire column, check **Static** and enter or select the unit under Unit. Time units must come from a fixed, controlled list. Other units can be any string, but if a unit is not defined in the PKS, unit conversions will not be available for it.
3. Click the **Hide** button to exit the Attributes dialog and return to the [Subject data](#) dialog.

Time

The screenshot shows the 'Study Creation Wizard' dialog box, specifically the 'Map Time Data' step. The dialog has a title bar with a close button. Below the title bar, there is a text area with instructions: 'Select the columns containing the Time data. Minimally, the Relative Nominal Time must be defined, along with the unit of time. The unit of time can be obtained from a column also.' Below this, there is a section titled 'Map Time Data' which contains a list of 'Worksheet Columns' on the left and several input fields on the right. The 'Worksheet Columns' list includes: Period, Treatment, TimeActual, Amount, Tau, G, H, and TimePlanned. The 'TimePlanned' column is currently selected. The input fields on the right are: 'Relative Actual Time' (with 'TimeActual' entered), 'Relative Nominal Time' (with 'TimePlanned' entered), 'Relative Actual End Time' (empty), 'Time Unit' (a dropdown menu showing 'hr'), 'Clock Time' (empty), and 'Relative Nominal End Time' (empty). There is a checkbox labeled 'Static' which is checked. On the right side of the dialog, there are buttons for 'Load', 'Save', 'Preview', and 'Cancel'. At the bottom, there are buttons for 'Help', '<< Back', 'Next >>', and 'Cancel'. A checkbox labeled 'Hide used Columns' is also present at the bottom left.

1. *Relative Nominal Time*: Drag the variable containing scheduled times, expressed relative to the start of the study, to this field.
2. Set other time variables (optional). Note that infusion dosing requires Relative Actual End Time. See “[Time fields](#)” on page 451 for a discussion of the PKS time variables. All time variables must exist with identical names in both the observations worksheet and the dosing worksheet, if included.
3. *Time unit*: Either choose a column containing the units for each record, or check **Static** and enter the unit under Time Unit. See your system administrator for a list of allowed units.
4. Click **Next** to map [Sample data](#).

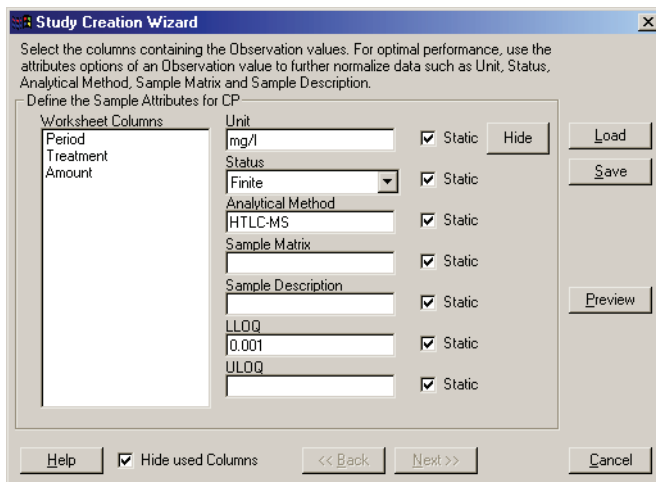
Sample data



1. Drag each column containing sample data to the PKS Fields box.
2. If a column includes multiple, concurrent measurements for any given subject, drag the variable that differentiates measurements to Sample Number.
3. Select a variable under PKS Fields and click the **Attributes** button to edit the [Sample attributes](#).
4. Click **Next** to set [Dosing](#).

Sample attributes

All attributes are optional.

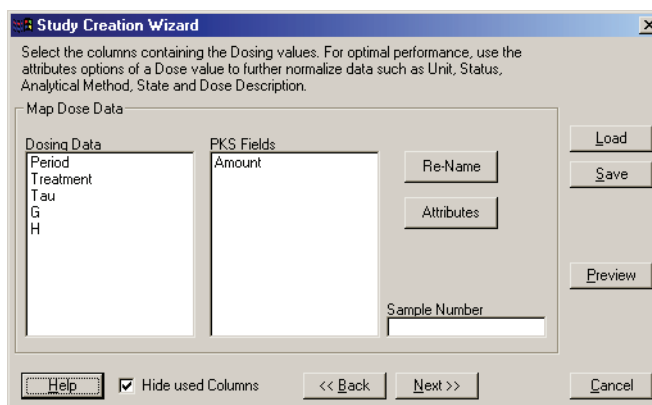


Each attribute can take one value for an entire column, or a unique value for each observation record (from a column in the data set).

- *Unit*: units of measurement.
 - *Status*: indicates data status, e.g., below a limit of quantification (BQL), or missing due to dropout or non-adherence. Valid data are “finite.” The list of statuses is defined by your PKS administrator.
 - *Analytical Method*: assay or technique used to obtain the sample data.
 - *Sample Description*: further information about the sample.
 - *LLOQ*: lower limit of quantification for the assay. A number value.
 - *ULOQ*: upper limit of quantification for the assay. A number value.
1. To load attribute values for each record from a column in the data set, drag that column to the appropriate field.
 2. To set one attribute value for an entire column, check **Static** and enter or select the value.
 3. Click the **Hide** button to exit the Attributes dialog and return to the [Sample data](#).

Dosing

1. If a dosing worksheet is used, the Map Dosing Data screen appears.



2. Drag the column(s) containing dose amounts to the PKS Fields box.
3. Select a dose variable under PKS Fields and click the **Attributes** button to set units and other optional attributes, as needed. (See “[Dosing attributes](#)” below.)

4. Click **Next** to continue to the [Data Collection Point \(DCP\)](#) screen.

Dosing attributes

Study Creation Wizard

The attributes of a Dose can be either static or selected from a column. For optimal performance, use the attributes options of a Dose value to further normalize data such as Unit, Status, Analytical Method, State and Dose Description.

Define the Dose Attributes for Form

Worksheet Columns	Unit	Status	Analytical Method	State	Dose Description
seq	mg				
per					

Buttons: Help, ☒ Hide used Columns, << Back, Next >>, Cancel

All attributes are optional. Each can take one value for an entire column, or a unique value for each observation record (from a column in the data set).

- *Unit*: units of measurement.
 - *Status*: indicates dose status, e.g., missing due to dropout or non-adherence. Valid data are “finite.” The list of statuses is defined by your PKS administrator.
 - *Analytical Method*: assay or technique used to obtain the sample data.
 - *State*: further text information about the sample.
 - *Dose Description*: further text information about the sample.
1. To load attribute values for each record from a column in the data set, drag that column to the appropriate field.
 2. To set one attribute value for an entire column, check **Static** and enter or select the value.
 3. Click the **Hide** button to exit the Attributes dialog and return to the [Dosing](#).

Data Collection Point

The screenshot shows the 'Study Creation Wizard' dialog box, specifically the 'Data Collection Point' step. The title bar reads 'Study Creation Wizard'. The main text area contains the instruction: 'To finish the study creation, additional profiling of the data can be done, but is not required. Select a Period, Phase, Matrix, Day, Visit, Analyte, Data Collection Point Description, and/or Treatment column.' Below this, the 'Map DCP Data' section is active. It features a 'Worksheet Columns' list on the left containing 'Treatment'. To the right of this list are several input fields: 'Period' (with 'Period' selected), 'Phase', 'Matrix', 'Day', 'Visit', 'Analyte', and 'DCP Description'. On the far right of this section are 'Load', 'Save', and 'Preview' buttons. At the bottom of the dialog are 'Help', 'Hide used Columns' (checked), '<< Back', 'Next >>', and 'Cancel' buttons.

1. Drag variables representing the study period (required for crossover designs only), phase (optional) and other study descriptors to the appropriate boxes.
2. *DCP Description*: (Data Collection Point description) use this field for variables that distinguish different measurements stored a single observation column, for example, an indicator variable noting whether a given record contains a measurement of plasma concentration or pain score.
3. Click **Next** to set **Treatment** variables.

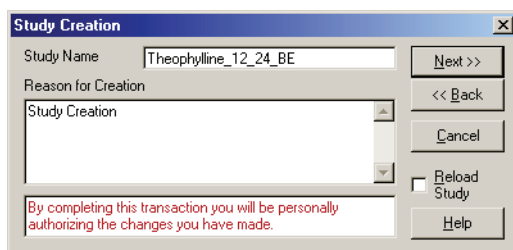
Treatment

The screenshot shows the 'Study Creation Wizard' dialog box, specifically the 'Treatment' step. The title bar reads 'Study Creation Wizard'. The main text area contains the instruction: 'Select a Treatment, Route, Regimen, Formulation, FastedFed, State, and/or Treatment Description column if required for this study.' Below this, the 'Map Treatment Data' section is active. It features a 'Worksheet Columns' list on the left. To the right of this list are several input fields: 'Treatment Code' (with 'Treatment' selected), 'Route', 'Regimen', 'Fasted Fed', 'Formulation', 'State', and 'Treatment Description'. On the far right of this section are 'Load', 'Save', and 'Preview' buttons. At the bottom of the dialog are 'Help', 'Hide used Columns' (checked), '<< Back', 'Finish', and 'Cancel' buttons.

1. Drag variables to the following (optional) to add treatment descriptors per your company's standards:
 - a. *Treatment Code*: variable identifying the treatment.
 - b. *Route*: variable identifying route (*oral, IV, IM*, etc.) for each dose.
 - c. *Regimen*: variable identifying dose regimen.
 - d. *Fasted/fed*: variable identifying whether subject ate before dosing.
 - e. *Formulation*: variable identifying drug formulation for each dose.
 - f. *State*: additional descriptive field.
 - g. *Treatment description*: additional descriptive field.
2. Use the **Preview** button to review the variable mappings, and the **Back** button to make any adjustments. Use **Save** to save mappings to a re-loadable file.
3. Click **Finish** when the variable mappings are complete. The **Study creation** dialog appears.

Study creation

This dialog is the final step in study creation, causing study data to be saved to PKS. (See “Creating a study” on page 456.)



Note: The study name cannot be changed in this dialog.

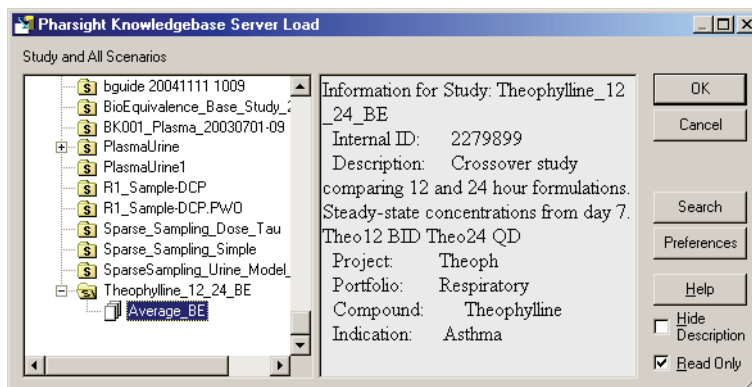
1. If the study does not need to remain open in WinNonlin for modeling or other further work, uncheck **Reload Study**.
2. When all necessary data has been entered, click **Next** to save the PKS study.
3. Once all data are successfully loaded to the PKS, a confirmation dialog will appear. Click **OK**.

Loading a study or scenario

To load a PKS study into WinNonlin:

1. Launch WinNonlin, if needed. Do not leave any windows open; PKS will close them before loading any study.
2. Choose **PKS>Load...** from the WinNonlin menus.
3. Log into PKS as detailed under “Connecting to the PKS” on page 455.
4. Click **OK**.

WinNonlin will contact the PKS database and load a tree view of available studies and scenarios. The list is determined by your user access rights, assigned by your PKS administrator.

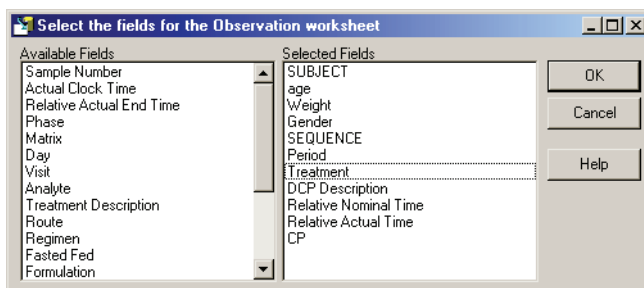


Note: A red scenario icon indicates that at least one derived output object is out of sync with its source. A green scenario icon means that the study data have changed since the scenario last loaded the study data.

5. Un-check the **Read Only** check box if any alterations or additions are to be made to the existing study or scenario (as opposed to creating a new scenario). Loading the study as Read Only improves performance significantly.
6. Use the **Search** button to search by study attributes. See “Scenario search”.
7. Use the **Preferences** button to organize the study tree by study name, indication, compound, and/or portfolio. In addition, the preferences select whether all scenarios will be displayed (“All”), or only the most-recent (“Latest”), that is, the final leaf on each scenario branch in the study tree.

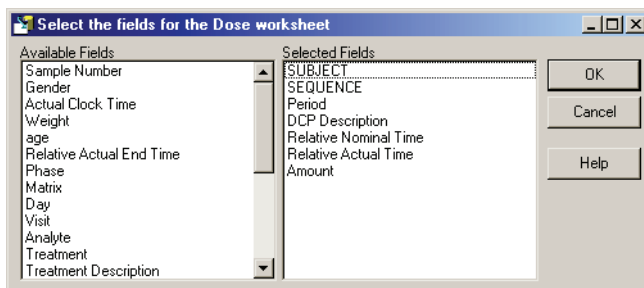
8. Select the study or scenario and click **OK** to load the scenario or, for studies, to select data columns to appear in observation and dose worksheets.

Select observations



Drag and drop columns to include in the observation worksheet when the study is loaded into WinNonlin, and click **OK**.

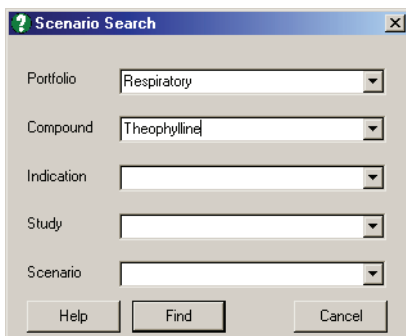
Select dose data



Select the columns to display in the dosing worksheet and click **OK** to load the data into WinNonlin.

Scenario search

The Search button available while [Loading a study or scenario](#) allows a search of available studies by study name, scenario name and/or value of study meta-data, including portfolio, compound and indication.

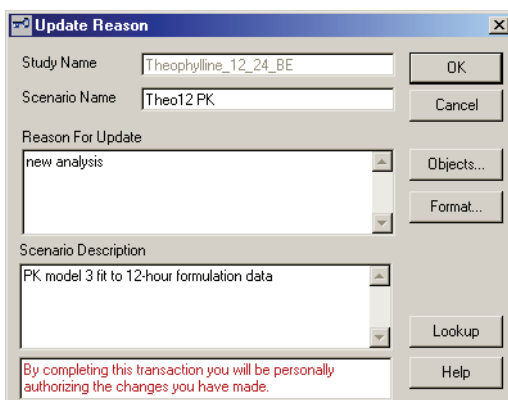
A dialog box titled "Scenario Search" with a question mark icon. It contains five dropdown menus: "Portfolio" (set to "Respiratory"), "Compound" (set to "Theophylline"), "Indication", "Study", and "Scenario". At the bottom are three buttons: "Help", "Find", and "Cancel".

Creating scenarios

Each scenario contains analysis settings and results for one analysis. A scenario can be derived from an existing study or from another scenario. Each scenario can be updated, creating a new *version* of the scenario.

To save analysis results to PKS:

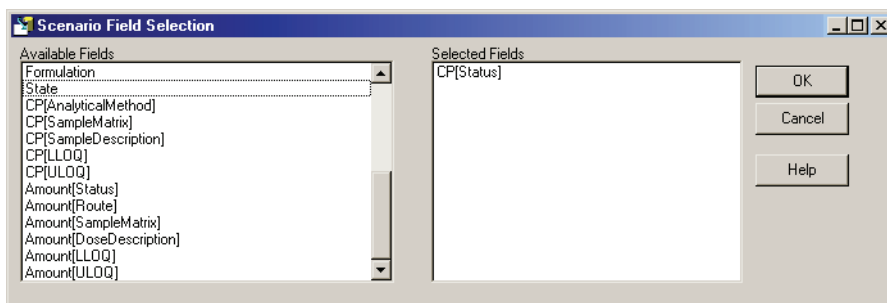
1. Load the PKS study to analyze in WinNonlin as detailed under “[Loading a study or scenario](#)” on page 467.
2. Load the model or analysis, including any related settings, in WinNonlin. If the PKS study does not include dose data, it can be added with the scenario.
3. If analysis results will be saved in the scenario, run the model or analysis.
4. Once the analysis is complete, choose **PKS>Save** from the menus.

A dialog box titled "Update Reason" with a folder icon. It contains two text input fields: "Study Name" (set to "Theophylline_12_24_BE") and "Scenario Name" (set to "Theo12 PK"). Below these are two list boxes: "Reason For Update" (containing "new analysis") and "Scenario Description" (containing "PK model 3 fit to 12-hour formulation data"). To the right of the list boxes are buttons for "Objects..." and "Format...". At the bottom right are buttons for "Lookup" and "Help". A red text box at the bottom left contains the message: "By completing this transaction you will be personally authorizing the changes you have made." Buttons for "OK" and "Cancel" are located to the right of the input fields.

5. Enter a scenario name and reason for the study (scenario) update. Both are required. Enter a scenario description (optional). Click **OK**.
6. If any output windows were not saved and named, a dialog will appear requesting names. Click **OK**.
7. Log in to PKS if required. The new scenario will be saved with the original study. A final “Save complete” dialog will confirm scenario creation.

Adding data columns to a scenario

After a scenario is created, you can add columns from the study to the scenario by loading the scenario and choosing **PKS>Add Columns...** from the WinNonlin menus. The Scenario Field Selection dialog appears.



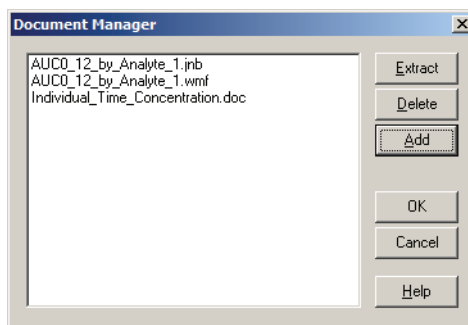
Drag study data columns from the left side to the Selected Fields box to add those data columns to the Observations worksheet for the scenario. To save the updated scenario, choose **PKS>Save** from the WinNonlin menus.

Adding non-WinNonlin documents to a scenario

Non-WinNonlin documents can be saved to (or loaded from) a PKS with a scenario. Such documents may include graphics files, data sets from other analysis programs, Microsoft Word reports created with the PKS Reporter, or graphics created from WinNonlin charts, among others. Any file can be attached to a scenario. PKS can display JPG files in the Web Administrator interface. The PKS Client for Word and the PKS Client for Excel can view XL and Word documents. Other documents can be attached, but may not be viewable until extracted from the PKS, saved to disk, and viewed in an appropriate software program. See also “[PKS share](#)” on page 479 and “[Libraries](#)” on page 480.

To save or load a non-WinNonlin document as part of a PKS scenario:

1. Load the scenario. See “Loading a study or scenario” on page 467.
2. Choose **PKS>Other Documents** from the WinNonlin menus to open the Document Manager.



3. Use the **Add** button to locate and save a file to the PKS, as part of the current scenario.
4. Select a document in the list and click the **Extract** button to save it to a hard drive or network location.
5. Select a document and click the **Delete** button to remove it from the scenario.
6. To delete multiple documents, highlight all the documents to be deleted and click the **Delete** button.
7. Click **OK** when finished.
8. To commit modifications of the document manager list to the scenario in the PKS, choose **PKS>Save** in the menus.

Optimizing PKS/WNL performance

A number of issues impact the performance of operations between WinNonlin and the PKS.

Read-only access

When loading data, WinNonlin and the PKS must do a significant amount of work to track any changes to the study data. If the study data are loaded as read-only—allowing the user to analyze them in WinNonlin but not change data values in the source data—the loading can be done much more quickly.

Note: The read-only check box on the PKS Load dialog applies to the study data and not the scenario. All scenarios are, by their very nature, read-only. Only new versions of scenarios can be created. Users must have system level rights and right to the original study in order to create a new scenario.

Appropriate field mappings

Study data fields must be mapped to an appropriate data type when initially loaded from the WinNonlin client into the PKS. The basic data types are (1) Subject identifier, (2) Subject data, (3) Time (actual, nominal, and combinations), or (4) Observation. Some data types (such as subject identifiers and subject data) are constant across time, while others (observation data) vary. When studies are loaded, the observation data are checked for any changes. Thus, the more that data are mapped as observation data, the more they must be checked when loaded and the slower the loading is.

Subject identifiers (up to five fields) uniquely identify a subject. While they can be changed (corrected), they are not normally modified in the course of a study and its analysis. Changes are captured in an audit.

Subject data (unlimited number of fields) are data related to a subject that does not change over time. This data may also be called baseline covariates or demographics.

Time data identify the nominal and actual time points for the observed data. *Nominal times* are the dose and observation times designated in the study protocol. *Actual time* differs from nominal time when subject adherence to the protocol schedule is imperfect.

Observation data (unlimited) include sample or dose data that change over time for a given subject.

In creating a study or scenario and loading/reloading, WinNonlin and the PKS track any changes to the data. Make sure data that varies over time are mapped as observation data while demographic data are mapped as subject data. If baseline covariates are mapped as observations rather than subject data, the loading/reloading process becomes much slower because those data cells must be checked for changes.

Note: Mapping fields in the creation of a study is particularly important. Selecting the correct data types can ensure performance is optimized. It is not possible to change the data types in a study once it has been created.

Creation of a base scenario for a study

Loading a scenario is faster than loading a study. Therefore, to save time on subsequent operations, create a *base scenario* that contains all the columns that will be used for a set of scenarios on a study. All subsequent scenarios should be derived from this base scenario. This allows WinNonlin to load the study data in a more condensed format if read-only study data access is selected.

Example:

1. Load the study from the PKS into WinNonlin. See “[Loading a study or scenario](#)” on page 467.
2. Once the study is loaded, create a text object describing the base scenario (**File>New** then **Text**).
3. Name the Text object by right clicking on it, and selecting **Object Name**. Highlight the Text object in the list box and press F2. Give the object a name (i.e., Scenario Description). Click **OK**.
4. Save the base scenario via **PKS>Save**.
5. An Update Reason dialog is displayed. Enter a scenario name for the base scenario (i.e., Base Scenario). Enter a reason (i.e., Base Scenario). Click **Save**.
6. The logon dialog appears. Enter the password and click **OK**.
7. **File>Close All** (it may take some time to close all since a bit of memory is being freed up).
8. Now all work can be done by loading the base scenario with read-only access to the study; this will be much faster than loading the source study. Modeling and other analysis can be done and saved as new scenarios using **PKS>Save As**.

Note: Make sure that all dosing data are loaded into the study. Dosing data are considered part of the study and cannot be added after studies are loaded in read-only mode.

Change tracking and audit information

If any changes are made to a PKS study or scenario, and the new data are saved back to the PKS, the system will display the following dialogs to document all changes. A change reason must be entered for every changed value in study data, and for each scenario update. All change records, including date, change made, PKS user name and change reason, are stored and made available via the PKS web client's audit trail.

Audit information

If the study data have changed, the Audit Information dialog appears. A reason for change is required for every alteration to study data.

See your system administrator to inquire about standard reason text or click the **Lookup** button for a list of company-standard values.

Note: If 100 records or fewer have changed, fill in a reason for each item. Use the **Copy** and **Paste All** buttons to speed the process. If more than 100 records have changed, the Audit Information dialog presents only one Reason field. Enter information for all changes. This reason will be recorded for every change that was made.

Obs:	Change description	Original Value	New Value	Reason
1	Subject (subject=1, WEIGHT=49) Period=1 Phase=BABA Nominal Time=0 hr Treatment=B	PH101=BQL (ng/ml)	PH101=0 (ng/ml)	Replacing BQL value with zero

Help Copy Paste All Clear All Lookup OK Cancel

Update reason

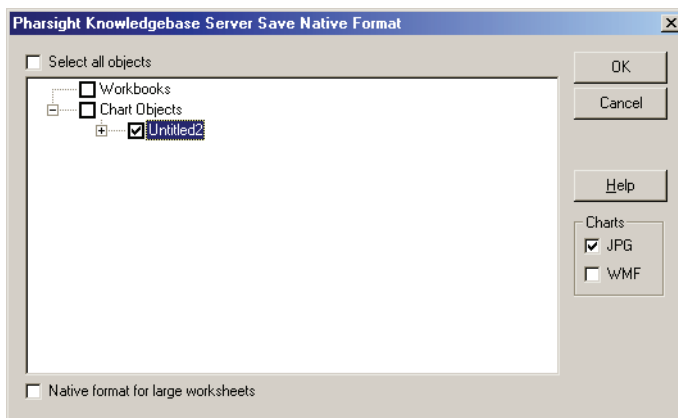
When saving a new scenario or scenario version, you will see the Update Reason dialog. Enter the scenario name (for new scenarios), update reason and (optional) scenario description. Use **Ctrl+Enter** to insert line breaks.

Use the **Lookup** button to select a reason for change from a standard list, created by your system administrators.

The **Format** button opens the PKS Save Native Format dialog, which provides a means to save worksheets and charts in binary formats. Worksheets may be saved as binary files, rather than saving individual data values to the database. This will speed loading of those data objects to and from PKS, but make data values unavailable to query and other tools within the PKS Web Client. Charts may be saved in JPG or Windows metafile (WMF) format, instead of as Pharsight chart objects.

Check items to save in binary format.

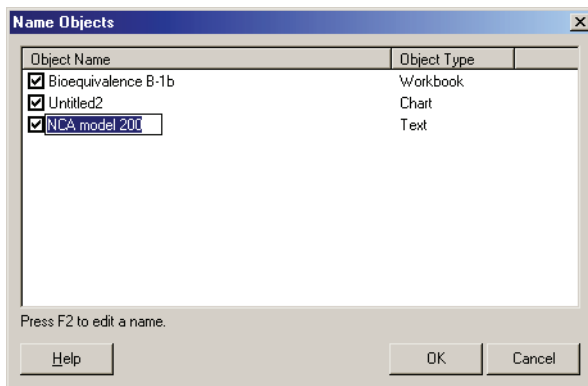
Check **Native format for large worksheets** to save all worksheets larger than a set limit as binary objects, rather than saving individual data values to the database. Enter the minimum worksheet size to be saved in binary format under # cells and click **Apply**. Worksheets with fewer cells will save individual data values to PKS.



Once the Audit Information dialog is complete, click **OK** to enter the update reason(s). If no scenario was loaded, PKS asks whether to create a base scenario with the updated data and any other changes that have been made in WinNonlin. Clicking **Yes** adds a new scenario with the current WinNonlin settings and objects.

Name objects

WinNonlin will prompt you to provide names for any unnamed objects in the scenario. Select an object name, then click on it again to edit the name.



Dosing information

Dose data are added to PKS studies either through a separate dose worksheet at study creation, or as part of a modeling scenario (via the WinNonlin Dosing

Regimen dialog, after loading the model). See “[Dosing](#)” on page 452 for discussion.

After loading a study or scenario that contains dose information, WinNonlin contains a workbook with three worksheets: observation data (Observations), one containing dosing data (Dosing), and one that records the data history (History). The Dosing worksheet can contain dose information for one or more analysis scenarios. See “[Dosing worksheet](#)” on page 452 for data requirements.

Changing the dose regimen

When the dosing data exists on the Dosing worksheet, it is possible to indicate which fields apply to the current scenario. Thus different dosing schedules can be saved in a study or scenario, and applied to different scenarios by including different data columns [Dose_A, Dose_B, etc.] as follows.

For example, suppose that a PKS study contains time-concentration data for a crossover trial comparing IV bolus and oral drug delivery. Non-compartmental analysis will be performed for each. Dose times and amounts for the two formulations appear in four columns of the Dosing worksheet:

- IV_Time
- IV_Amount
- Oral_Time
- Oral_Amount.

Since the IV and Oral formulations require different NCA models, and WinNonlin can load only one model at a time, two scenarios are required:

- Oral scenario: uses NCA model 200 and dose columns Oral_Time and Oral_Amount
- IV scenario: uses NCA model 201 and dose columns IV_Time and IV_Amount

The selection of dose columns for each scenario is made as follows.

To choose dosing columns to apply in the current scenario:

1. Load a study or scenario containing multiple dose regimens as described above.

1. Select **PKS>Dosing**. The Dosing Field Selection dialog appears.
2. Drag the dosing fields for the current analysis to the Dosing Columns box. Variables listed in the Non-Dosing Columns box will not be used in the current scenario.
3. Click **OK**.

Appending or updating a study

After a study is created in the PKS, additional data can be added to the study, or existing data can be updated, from a workbook.

Note: Append/Update is a convenient way to add dosing information to the study or scenario. This method can also be used to add subject data to the study.

To append/update a study:

1. Close any PKS study or scenario that is open in WinNonlin.
2. Open the data to be added in a WinNonlin workbook window.
3. Choose **PKS>Append/Update Study...** from the menus.
4. Log in. (See “[Logging in](#)” on page 456 for details.)
5. In the Pharsight Knowledgebase Server Load dialog, select the study to which to append data, or in which to update data.
6. Click **OK**. The Study Creation Wizard appears, to map the variables in the data set to the selected study.
7. Drag variables from the workbook and drop them on or under the corresponding study variables. The column(s) containing the subject identifiers must be identified. If there are new data variables, such as extra subject data or other observations, add them beneath the existing study variables.
8. Click **OK**.

Note: The PKS documentation set covers a number of other operations that affect studies and scenarios in the PKS. For example, it is possible to specify or convert units in the PKS without loading the data in WinNonlin. It is also possible to load data into WinNonlin then specify units or convert units—then save them back to the PKS. Whenever there are significant differences between the actions in WinNonlin and those in PKS, the documentation systems point them out.

PKS libraries and object shares

PKS provides two features for sharing models, data, code and other data objects across studies and scenarios. First, any PKS scenario can share data objects that are part of another scenario, and maintain a link to that object so that updates are shared. Second, the PKS library feature supports creation of system-level objects that can be referenced by any scenario. Details appear under the following headings.

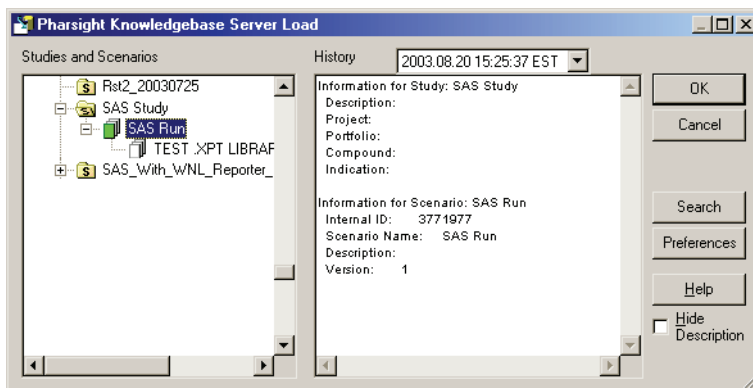
- [“PKS share” on page 479](#)
- [“Libraries” on page 480](#)

PKS share

A new analysis in WinNonlin can share non-Pharsight analysis and data files contained in any PKS scenario. When an analysis that uses shared files is saved to PKS, the share associations are saved with the new scenario. When the scenario is loaded in WinNonlin, any shared files may be updated to reflect changes in their source files as described under [“Library and share status” on page 482](#).

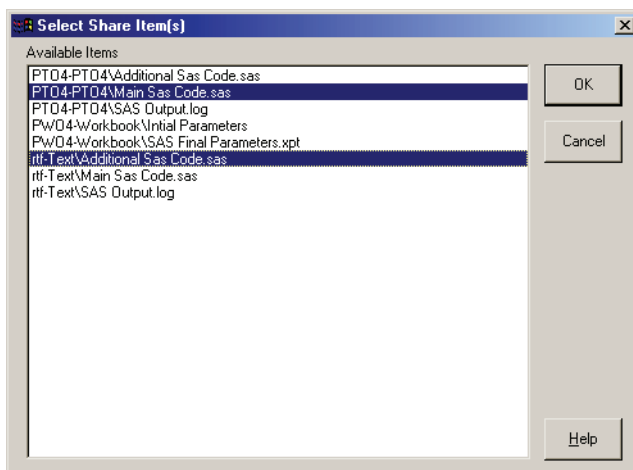
To load files from the Pharsight Knowledgebase Server:

1. Choose **PKS>Share** from the WinNonlin menus.
2. Log in to the PKS, as described under [“Logging in” on page 456](#).
3. In the Pharsight Knowledgebase Server Load dialog, choose the study and scenario containing the file(s) to be shared.



4. Click **OK**.

The Select Share Item(s) dialog will appear, presenting a list of non-Pharsight files that can be referenced in the current analysis.



5. Select items to load. Hold down the **Ctrl** key to select multiple items.

6. Click **OK** to load the items.

Libraries

Note: Each application has its own folder/container for storing the binary files. Consequently, applications cannot see other applications' library systems. For example, PKS Client may add a library item and it will not be visible by the WinNonlin application.

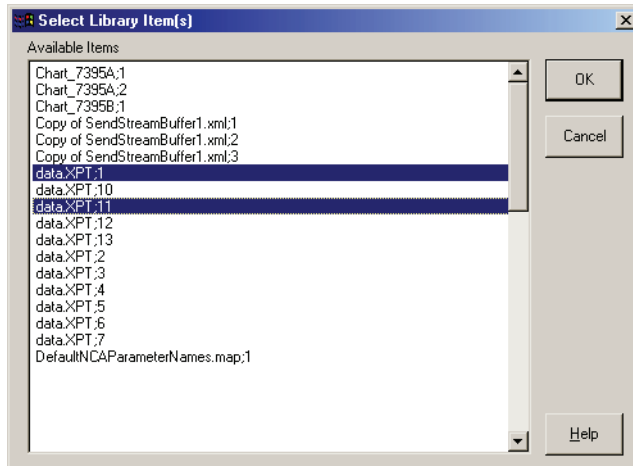
WinNonlin can save any type of binary file as a PKS system level object, and load it back again, using the Library function. Library items are not associated with any PKS study or scenario. The library function is intended to store a small number of files containing code, parameter names, and other items that will be used as standards across many studies and analyses. The library functions include the following:

- “Loading a library from the PKS”
- “Adding or updating a library in the PKS” on page 481

Loading a library from the PKS

Use WinNonlin to load standardized parameter names, NONMEM or SAS code, and other files stored as [Libraries](#) in the PKS.

1. Choose **PKS>Libraries>Load** from the WinNonlin menus.
2. Log in to the PKS, as described under “[Connecting to the PKS](#)” on page 455. The Select Library Item(s) dialog will appear.



3. Select the library(ies). Hold down the **Ctrl** key to make multiple selections.
4. Click **OK**.

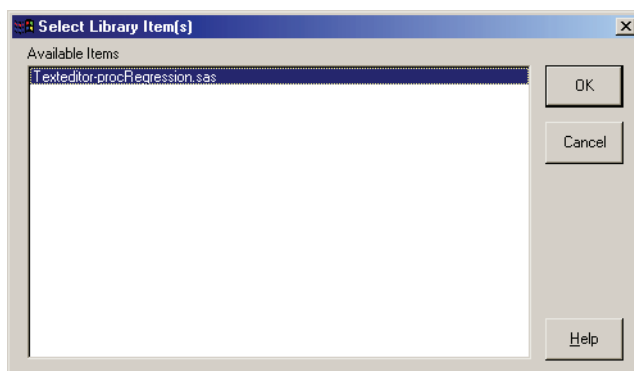
Adding or updating a library in the PKS

Use this feature to save standardized parameter names, NONMEM or SAS code, and other files as [Libraries](#) in the PKS.

To save a file to the PKS as a library or update an existing library:

1. Open the file in the WinNonlin using **File>Open**, or load an existing PKS library as described under “[Loading a library from the PKS](#)” on page 481.
2. Make any desired changes. You may save the file to a local disk to save interim changes before uploading them to the PKS.
3. Choose **PKS>Libraries>Add/Update** from the WinNonlin menus.
4. A dialog will appear, to confirm that you wish to add PKS data. Click **Yes**.
5. Log in to the PKS, as described under “[Connecting to the PKS](#)” on page 455.

The Select Library Item(s) dialog appears.



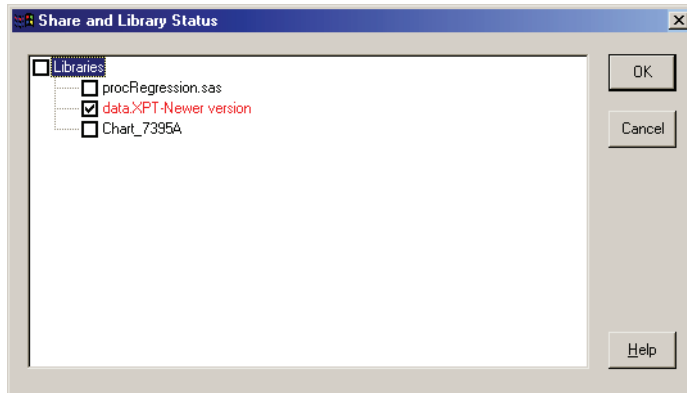
6. Select the item(s) to be loaded to the PKS and click **OK**. Hold down the **Ctrl** key to select multiple items.
7. In the PKS SAS Save dialog, enter the Library Description and the Reason for Update, following any standards set within your company. Click **OK**.
8. Click **OK** after the save operation is complete.

Library and share status

WinNonlin tracks whether [PKS share](#) items and [Libraries](#) are current with the latest version in the PKS. Items that are out-of-date can be updated as follows.

To synchronize shared files and/or libraries with the latest versions in the PKS:

1. With PKS share and/or library files loaded in WinNonlin, choose **PKS>Status...** from the menus. The Share and Library Status dialog appears. Items that have newer versions available in the PKS appear in red.



2. Check the items to be synchronized with the latest PKS version and click **OK**.

WinNonlin will load the latest version from the PKS.

PKS Information

Information about a study and/or scenario loaded into WinNonlin is summarized in the PKS Information dialog.

To view summary information for the currently-loaded study:

1. With a PKS study or scenario loaded in WinNonlin, choose **PKS>Information** from the menus.

The PKS Information dialog appears. It shows the following information.

- *Server Information:* time zone of the PKS server. All dates and times, including time stamps for study updates in PKS, will use this time zone.
- *Study information:*
 - *Locked:* if the study is locked, its data cannot be updated in PKS.
 - *User selected read only:* if true, the study is loaded as read-only; no changes may be saved to PKS.
 - *Name:* study name in PKS.
 - *Internal ID:* study identifier used in PKS.
 - *Description:* description of the most-recent version of the study, entered by the user when the study was last saved or updated.
 - *Type:* study type.

- *Design*: study design.
 - *BlindingType*: study blinding scheme.
 - *State*: state of the study in the PKS.
- *Scenario Name* in PKS
- *Scenario Description*: entered when the scenario was last saved in PKS.
- *Current user rights*: PKS access rights of the PKS user account used to load the study into WinNonlin. This level determines whether the user can save changes to this study in the PKS. See the PKS user documentation for a description of various levels of user access.

WinNonlin and PKS differences

WinNonlin is closely integrated with the Pharsight Knowledgebase Server. However, because each is also designed to work with other tools as well, there are some differences that are important to note.

Units

- Units are required for some fields in the PKS. While units are useful in WinNonlin, they are never required.
- New units can be added and defined in the PKS, with conversion factors to define relationships to existing units. WinNonlin performs calculations and conversions using only its internal set of units; custom units can be entered and carried through operations, but they do not affect calculations. Conversions are not available for custom units in WinNonlin. By default, PKS includes all of WinNonlin's default units.
- Units can be handled at the cellular level in the PKS. That is, each record in a given column can use different units. (The units would be stored in an adjacent column.) In WinNonlin, units are associated with an entire column. Thus there are situations when valid data from a PKS data set cannot load correctly in WinNonlin until conversions are performed to make units uniform within each column. The PKS provides the means perform this conversion.
- PKS requires that all time-dependent data (Relative Actual Time, Relative Nominal Time, etc., for dosing and observations) use the same time unit. This is not required in WinNonlin.

Studies and scenarios

- Saving scenarios to PKS is analogous to saving a workspace in WinNonlin, except that workspaces require all child windows to have filenames prior to being saved. PKS requires the child windows to have “object names.”
- Time and subject data must be present to save and import data from WinNonlin to the PKS. If a data set does not include time or subject data, dummy columns must be created.
- The PKS can handle multiple drugs within one study. WinNonlin analyzes only one drug at a time.

Glossary

WinNonlin and PKS terminology

The following terms and definitions pertain specifically to objects and operations in WinNonlin and its operations with Pharsight Knowledgebase Server. *Italics* indicate terms that are defined elsewhere in the glossary.

A

A In Modeling, the zero time intercept associated with the Alpha phase. In Deconvolution A and alpha refer to the parameters of a polyexponential unit impulse response function of the form:

$$\text{UIR}(t) = \sum[A(j) * \exp(-\alpha(j) * t)], j = 1 \dots N.$$

actual time The time that a study dose or observation actually happened; this time may differ from the *nominal time*.

accumulation index

$$\frac{1}{[1 - e^{-(\lambda_z \times \tau)}]}$$

actual clock time In PKS, a placeholder, available for entering clock times from research data. Not used in any WinNonlin calculations.

AIC Akaike Information Criterion (AIC). A measure of goodness of fit based on maximum likelihood. When comparing several models for a given data set, the model associated with the smallest AIC is regarded as giving the best fit. Appropriate only for comparing models that use the same weighting scheme.

$AIC = N \log (WRSS) + 2P$ for modeling in WinNonlin. N is the number of observations with positive weight. $WRSS$ is the weighted residual sum of squares. P is the number of parameters.

In LinMix, AIC is defined as: $AIC = -2\hat{L}_R + 2S$ where \hat{L}_R is the restricted likelihood evaluated at converged estimates, and s is the rank of the fixed-effects design matrix plus the number of variance parameters, and variance components for linear mixed-effects modeling calculations.

Alpha In modeling, the macro rate constant associated with the distribution phase. In Deconvolution, A and alpha refer to the parameters of a polyexponential unit impulse response function of the form:

$$UIR(t) = \sum[A(j) * \exp(-\alpha(j) * t)], j = 1 \dots N.$$

Alpha half-life The half life associated with the macro constant Alpha. Sometimes denoted ALPHA_HL.

Anderson-Hauck pval The p -value of the Anderson-Hauck test statistic. This is derived from a non-central t distribution.

Anderson-Hauck statistic Test statistic applicable for testing equality of means in a bioequivalence study. The test statistic t is:

$$t = \frac{\bar{X}_E - \bar{X}_S - \frac{1}{2}(A + B)}{S \sqrt{1/n_E + 1/n_S}}$$

where the X 's are the respective sample means, n_E and n_S are the group sample sizes, and S is the estimate of the residual standard deviation. A and B are the lower and upper limits.

ANOVA command file (*.anv) Legacy ANOVA command file (*.ANV) containing all information required to recall and reproduce a given ANOVA model, for use in scripting. Replaced with the *LinMix command file* (*.LML) in WinNonlin versions 4.0 and later. These versions can open and run and AVNOVA command files and call them in scripts, but cannot save to that format.

AUC Area under a concentration of analyte vs. time curve. If $C(t)$ denotes the concentration of analyte at time t , then:

$$AUC_{a,b}^b = \int_a^b c(t) dt$$

AUCall *AUC* computed from time zero to the time of the last Y value.

AUCINF *AUC* from time zero extrapolated to infinity.

AUClast *AUC* computed from time zero to the time of the last positive *Y* value.

AUMC Area under the moment curve:

$$AUMC_a^b = \int_a^b c(t)tdt$$

AUMCINF Area under the moment curve when the time concentration curve is extrapolated to infinity.

$$AUMCINF = AUMC_0^\infty$$

AUMClast Area under the moment curve computed to the last observation.

$$AUMClast = AUMC_0^{t_{last}}$$

B

B The zero time intercept associated with the beta phase.

balanced design An experimental design in which the number of observations is the same for every combination of the experimental effects. For example, a crossover design with effects for formulation and period, each with two levels, is balanced if each of the four combinations of formulation by period has the same number of observations. If a subject drops out of one sequence, the design is no longer balanced.

Beta “Macro” rate constant associated with the elimination phase.

Beta half-life The half life associated with the macro constant Beta. Also called BETA_HL.

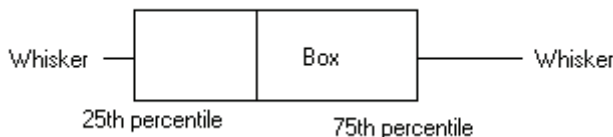
bioequivalence Bioequivalence is said to exist when a test formulation has a bioavailability that is similar to that of the reference. There are three types of bioequivalence: average, individual, and population. Average bioequivalence states that aver-

age bioavailability for test formulation is the same as for the reference formulation. Population bioequivalence is meant to assess equivalence in prescribability and takes into account both differences in mean bioavailability and in variability of bioavailability. Individual bioequivalence is meant to assess switchability of products, and is similar to population bioequivalence.

bounds Upper and lower constraints on the initial parameter estimates for compartmental modeling. The lower and upper values limit the range of values the parameters can take on during model fitting. This can be very valuable if the parameter values become unrealistic or the model will not converge. Although bounds are not always required, it is recommended that Lower and Upper bounds be used routinely. If they are used, every parameter must be given lower and upper bounds.

The WinNonlin default bounds are 0 for one bound and 10 times the initial estimate for the other bound. For models with parameters that may be either positive or negative, user-defined bounds are preferred.

box and whiskers plot The box and whiskers plot is designed to show the distribution of values.



The box marks the 25th and 75th percentiles, and the median between them. The whiskers end at the Max and Min points unless there are outliers beyond 1.5 times the H-spread [the distance between the hinges] from the hinges. Profiles with fewer than five data points plot the data points directly with no box.

BQL Below the lower limit of quantification (*LLOQ*) for a given assay.

C

C The zero-time intercept associated with the absorption phase for an extravascular model or the zero time intercept associated with the gamma phase for a 3 compartment IV bolus model.

carry along variables Variables that are not required for the current analysis, but will be copied and included in the output data set.

Cavg For steady-state data: computed as $AUC(0-\tau)$ divided by τ .

CL	Total body clearance. $CL = \text{Dose}/AUC$
classification variables	In a LinMix model, classification variables or covariates are independent variables that employ values from a discrete set (e.g. gender, smoking status), rather than continuous values. See also <i>regressors</i> .
clock time	Time expressed as a date and/or time, rather than relative to the first dose or the start of a study.
CLR	Renal Clearance.
Clss, Clss/F	For steady-state data: an estimate of the total body clearance. For extravascular models the fraction of dose absorbed cannot be estimated, therefore Clearance for these models is actually Clearance/F where F is the fraction of dose absorbed: $= \frac{\text{Dose}}{AUC_0^\tau}$
Cmax	The peak or maximum concentration.
Cmin	For steady-state data: the minimum concentration between 0 and τ (corresponding to Tmin).
command file	Legacy command file (*.CMD) from an earlier version of WinNonlin. A command file can contain all information required to reproduce a compartmental or noncompartmental model. Replaced by Pharsight Model Objects (*.PMO). WinNonlin can open and run, but not save *.CMD files. See also <i>LinMix command file</i> and <i>table definition file</i> .
compound	A drug molecule under study. It may include one or more <i>formulations</i> . The concept is used in PKS to group <i>study</i> objects that pertain to the same molecule.
condition number (COND)	Condition number of the matrix of partial derivatives. COND is the square root of the ratio of the largest to the smallest eigenvalue of the matrix of partial derivatives.
constant	Dosing information such as number of doses, magnitude of each dose, and dosing times. Each library model requires specific constants.

contrasts	In LinMix, linear combinations of levels of an effect, such that the sum of the coefficients is zero. For example, suppose that effect Trt has three levels: Placebo, 10mg, and 20mg. To compare the two dose groups against placebo, construct the contrast: Placebo=1, 10mg=-0.5, 20mg=-0.5. Note that $1-0.5-0.5=0$. This compares the average of the 10mg and 20mg groups versus placebo.
convergence criterion	The convergence criterion determines when WinNonlin can stop with a successful fit. Convergence is achieved when the relative change in the residual objective function is less than the convergence criteria.
covariates	Independent variables in a LinMix model. Generally, some subject characteristic of interest, such as age, blood pressure, or body weight. Covariates are continuous measurements.
crossed factors	In a LinMix crossed-classification model every level of every factor could be used in combination with every level of every other factor. In this way, the factors “cross” each other. It is not required that data be available at every intersection of the factors. See also <i>nested factors</i> .
curve stripping	A graphical method for estimating parameters associated with models that can be written as a sum of exponentials. This method can be used to determine initial estimates required as a prelude to nonlinear least-squares regression.

D

dependent variable	Dependent variables have some functional relationship, possibly unknown, to the independent variables. A variable such as “drug concentration” is usually a dependent variable because it depends on factors like “formulation,” “period,” or “time of sample.” See also <i>independent variable</i> .
dosing regimens	The schedule of dosing. This normally includes the designated dose and the time of the dose. It may also include the dose cycle, the scheduled times for observations, etc., depending on the particular model's requirements.

E

E	Denotes Effect.
EC50	Concentration that achieves 50% of predicted maximum effect in an Emax model.

eigenvalue An eigenvalue of matrix A is a number λ , such that $Ax = \lambda x$ for some vector x , where x is the eigenvector. Eigenvalues and their associated eigenvectors can be thought of as building blocks for matrices.

The eigenvalues included in WinNonlin's modeling output are associated with the variance - covariance matrix. For compartmental modeling, WinNonlin prints a condition number that is equal to the square root of the ratio of the largest to the smallest eigenvalue. Very large values of the condition number may be indicative of instability in the model fitting process.

E0 Effect at time 0.

E_{max} Maximum effect.

engine Scripting language engines execute tasks in WinNonlin using [methods](#). WinNonlin engines are: PK, DESC, ANOVA, PLOT, TABLE, TRANSFORM, MERGE, LINMIX, BIOEQUIVALENCE.

exponential curve stripping A graphical method for estimating parameters associated with models that can be written as a sum of exponentials. This method can be used to determine initial estimates required as a prelude to nonlinear least-squares regression.

F

Fabs Fraction absorbed over time, estimated from *in-vivo* time-concentration data.

Fdiss Fraction dissolved over time *in-vitro*.

first-order Used to denote rate constants when the transfer rates are proportional to the amount in the source compartment.

fixed effects Generally associated with explanatory variables, as in a standard linear model. The variables can be either qualitative, as in traditional ANOVA, or quantitative, as in standard linear regression.

% fluctuation For steady-state data: % fluctuation is computed as $100 \cdot (C_{\max} - C_{\min}) / C_{\text{avg}}$, where C_{\min} and C_{\max} were obtained between 0 and τ .

formulation The point of dose input into a *structural* (PK/PD) model.

function variable A column in the data set that indicates which observations belong with which model functions, for fitting multiple functions to one data set. Required for simultaneous link models.

G

Gamma	The terminal elimination rate associated with a three compartment, first-order model. Sometimes denoted GAMMA_HL.
Gamma half-life	The half-life associated with the macro constant Gamma.
Gauss-Newton algorithm	A model fitting algorithm. Modifications of the Gauss-Newton algorithm include the Hartley and Levenberg modifications. Hartley Modification: This method is not as robust as others but is very fast. Levenberg Modification: A model fitting algorithm that performs well on a wide class of problems, including ill-conditioned data sets. This method requires the estimated variance - covariance matrix of the parameters, however the Levenberg modification suppresses the magnitude of the change in parameter values from iteration to iteration to a reasonable amount. This method is the default.
geometric mean	The geometric mean is the nth root of the product of the n Y's: $GM = \sqrt[n]{Y_1 Y_2 Y_3 \dots Y_n}$. This requires that each $Y > 0$.
group variables	In the Chart Wizard, a group variable breaks the data into separate plot lines. In LinMix, group variables allow for heterogeneity of variances while retaining the same mean model. For the Table Wizard, group variables are much like ID variables, with the exception that only unique values of group variables are printed. They are printed only when their value changes and at page breaks. Multiple group variables may be used. Note that group variables may appear as a column in the body of the table, or as an information line above the column titles, as selected using the Tools>Options menu item.

H

harmonic mean	The harmonic mean is the reciprocal of the arithmetic mean of the reciprocals of the Ys ($Y > 0$).
---------------	--

$$HM = \left(\frac{1}{n} \sum \frac{1}{Y} \right)^{-1}$$

I

ID variables Used by the Table Wizard to identify data in adjacent “Data” variable columns. When summary statistics are selected these variables are not considered. Note that multiple ID variables may be used, and that this may be a convenient way to include units.

increment for partial derivatives Nonlinear algorithms require derivatives of the models with respect to the parameters. The program estimates these derivatives using a forward difference. For example:

$$\frac{dF}{dP(1)} \cong \frac{F(P(1) + \Delta) - F(P(1))}{\Delta}$$

where Δ = Increment times the parameter value. The default is 0.001.

independent variable The independent variables are controlled by the experimenter (e.g., formulation) or do not depend on other factors. See also *dependent variable*.

indirect response models The measured response, R, to a drug may be produced by an indirect mechanism. Factors controlling the input or production, k_{in} , of a response may be either inhibited or stimulated, or determinants of loss, k_{out} , of the response may be inhibited or stimulated.

interaction Interaction is the degree to which the effect of one factor differs with varying values of another factor. For example, if a treatment has a different effect in one period than it does in another period, then Treatment and Period are said to interact. Interaction terms would be included in a LinMix or Bioequivalence model using the “*” symbol: Treatment*Period.

intercept For LinMix ANOVA and regression models, the value of the response when all effects are zero.

IPR Indirect Pharmacodynamic Response Model.

iterative reweighting A weighting scheme where weights associated with the individual observations are functions of the predicted values.

IVIVC *In-vitro in-vivo* correlation; a model describing the relationship between dissolution *in-vitro* and absorption *in-vivo*, and its related uses in formulation development.

J

join variables In the Tables Wizard, when merging data from two workbooks into one table, joins identify corresponding variables to be used in merging profiles from the two data sets, e.g., “Subj” and “Subject.”

K

K half-life The half-life associated with the rate constant K. Sometimes denoted K_HL.

K01 The rate at which the drug enters the central compartment from outside the system. It is sometimes denoted as Ka.

K10 The rate at which the drug leaves the system from the central compartment. The elimination rate.

K10 half-life The half-life associated with the rate constant K10. Often denoted K10_HL.

Kij For first-order rate constants; the rate from Compartment i to Compartment j.

K0 Zero-order input or infusion rate constant.

KM Michaelis constant for models involving nonlinear kinetics.

L

lag time (LT) Lag time is a time delay between drug administration and the onset of absorption. An example of such a process is the stomach-emptying time, following oral administration of a drug that is not absorbed from the stomach. When a drug is administered by bolus intravenous injection there is no lag phase.

Lambda Z (λ_z) First order rate constant associated with the terminal (log-linear) elimination phase. This is estimated via linear regression of time vs. log concentration.

least squares fitting Minimizes the sum of the squares of the deviations between the observed and predicted values of the *dependent variable*.

least squares means Least squares means are estimates of what the mean values would have been had the data been balanced (see *balanced design*). That is, least squares means are the means predicted by the fixed-effects model. If a data set is balanced, the least square means will be identical to the raw (observed) means.

linear kinetics	A model in which the transfer rates are first order.
linear regression	A model relating an independent variable to dependent variables such that the unknown parameters enter into the model linearly. In the simplest case, a straight line is fit to the data. The intercept and slope are determined by least squares. The slope and intercept are called model coefficients, or parameters.
LinMix command file	File (*.LML) storing all information needed to reproduce a LinMix analysis, often used in scripting. See “ Linear Mixed Effects Modeling ” on page 327.
LLOQ	Lower limit of quantification, the concentration whose lower bound (of variability in the assay) includes zero. Analyte may be detectable below this concentration even though it is not quantifiable.

M

macro rate constants	Compartmental models involving first order kinetics can be written as a sum of exponentials. The associated parameters are called macro rate constants and are functions of the underlying micro rate constants.
max	Maximum value.
mean	The arithmetic average.
mean of the logs	Arithmetic average of the logarithms of the Ys.
mean square	<p>The variance is normally a function of weighted residual SS/df, or the mean square. For certain types of problems, when computing the variance the mean square needs to be set to a fixed value.</p> <p>It is possible to use a specified value other than the residual mean square in the estimates of the variances and standard deviations. This value replaces the residual sum of squares in the estimates of variances and standard deviations.</p>
method	In the scripting language, a method is a set of computer code that performs functionality on an <i>object</i> or <i>engine</i> . For example, the method 'LOAD' causes an associated data file (an object) to be loaded. The method 'RUN' causes an associated PK Model (engine) to run the model.
Michaelis-Menten	A type of model often employed for models involving nonlinear kinetics. The Michaelis-Menten equation is: $V = V_{max} \times C / (K_m + C)$

micro rate constants	Transfer rates associated with a compartmental model. See also <i>macro rate constants</i> .
min	Minimum value.
MRT	Mean Residence Time is the average amount of time a particle remains in a compartment or system, equal to $AUMC / AUC$.
MRTINF	Mean Residence Time when the drug concentration profile is extrapolated to infinity. This can be estimated from single dose data or steady state data.
MRTlast	Mean Residence Time when the drug concentration profile is not extrapolated to infinity, but rather is based on values up to and including the last measured concentration: $MRT_{last} = AUMC_{last} / AUC_{last}$

N

N	In modeling output, the number of observations with non-missing data.
Nmiss	In modeling output, the number of observations with missing data.
Nelder-Mead simplex algorithm	A model fitting algorithm that does not require the estimated variance—covariance matrix as part of its algorithm. It often performs very well on ill-conditioned data sets (data sets that do not contain enough information to precisely estimate all of the parameters in the model). When used to fit a system of differential equations, it can be very slow unless the model has been compiled.
nested factors	In a nested classification, unlike a crossed classification, the levels of the nested factor are unrelated to one another and are nested within a level of another factor. For example, in a crossover study, Subjects are assigned to Treatment Sequences. This nested factor would be included in a LinMix or bioequivalence model using parentheses: Subject(Sequence). Subjects are not crossed with Sequences, because the subjects within one sequence are unrelated to the subjects within another sequence. See also <i>crossed factors</i> .
Nobs	The number of observations.
nominal time	The time specified in the protocol for a dose or observation—which may differ from the <i>actual time</i> that a dose or observation was taken.
noncompartmental analysis	Noncompartmental, or model independent analysis describes the concentration-time curve by estimating noncompartmental parameters such as: area

under the curve (AUC), Tmax, Cmax, and Lambda Z (the rate constant associated with the terminal elimination phase), among others. Noncompartmental parameters are estimated using the Linear or Linear/Log trapezoidal rule. If Lambda Z can be estimated, then the noncompartmental parameters will be extrapolated to infinity.

nonlinear kinetics Kinetics that can be modeled by nonlinear differential equations, for example,

$$V \frac{dc}{dt} = \frac{V_{max}C}{K_m + C}$$

nonlinear model In nonlinear models, at least one of the parameters appears as other than a coefficient of an independent variable. One such model is the sum of two or more exponentials, such as $Y = A1 * \exp(-B1 * X) + A2 * \exp(-B2 * X)$. A nonlinear model is one for which one or more of the partial derivatives with respect to the model parameters involve model parameters. Note that nonlinear models may arise from compartmental models having either linear or nonlinear kinetics.

nonlinear regression The process of fitting *nonlinear models* to data.

O

ODBC Open Database Connectivity. A standard for data exchange, in common use on Microsoft Windows platforms. It is an application programming interface for database access that uses Structured Query Language (SQL) as its database access language.

object In scripting, objects are described by one or more *property*/ies, and acted upon by *methods*. In WinNonlin, there are five main objects: data (workbook) objects, text objects, model objects, graph objects, and WinNonlin itself.

P

partial tests LinMix tests of whether each model term contributes to the fit after all other model terms have been included. Partial tests are invariant to the model specification order.

percentiles	The p^{th} percentile divides the distribution at a point where p percent of the distribution are below this point.
pharmacodynamic (PD) models	Models of drug effect as a function of a drug concentration.
pharmacokinetic (PK) models	Models of drug concentration as a function of time.
PK/PD link	A PK/PD Link model uses drug concentration data to estimate concentrations at an effect site, which are then used to model the pharmacodynamics.
planar confidence interval	A confidence interval obtained from the tangent planes to the joint confidence ellipsoid of all the parameter estimates.
power	In the Bioequivalence Wizard, the probability of detecting a difference greater than or equal to a specified percentage of the reference mean.
profile	A set of observed data to be fit at one time, producing one set of individual model parameters. A profile is identified as data records with a given, unique set of values for all <i>sort variables</i> .
property	Attribute or characteristic of an object in scripting, defining how the object should appear and interact. Properties also set values for an <i>object</i> or <i>engine</i> .

Q

R

random effect	Also known as <i>covariance parameters</i> , random effects distinguish a linear mixed effects models from standard linear models. In general, random effects are additional unknown random parameters assumed to impact the variability of the data. The variances of the random effects, commonly known as variance components, become the covariance parameters for this structure.
rank	Rank of the matrix of partial derivatives. If the matrix is of full rank, Rank= # of parameters. If the Rank is less than the number of parameters, then there is not enough information in the data to estimate all parameters in the model.

regressors	In a LinMix model, regressor variables or covariates are independent variables that employ values from a continuous set (e.g. temperature, body weight), rather than categories. See also <i>classification variables</i> .
relative actual time	In PKS, the time that observations or doses really happened, as opposed to <i>relative nominal time</i> . See also <i>relative actual end time</i> .
relative actual end time	In PKS, the times when infusion doses actually ended, as opposed to <i>relative nominal end time</i> . See also <i>relative actual time</i> .
relative nominal time	In PKS, scheduled dose and observation times, as set in the study protocol. Compare with <i>relative actual time</i> . See also <i>relative nominal end time</i> .
relative nominal end time	In PKS, the times when infusion doses were scheduled to end, as set in the study protocol. Compare with <i>relative actual end time</i> . See also <i>relative nominal end time</i> .
relative time	Elapsed time since the first dose or the beginning of the study, period or phase.
residuals	Observed minus predicted values, where the predicted values come from fitting the specified model.

S

S	Standard error of the weighted residuals: $S = \sqrt{(WRSS)/(DF)}$ where <i>WRSS</i> is the weighted residual sums of squares, and <i>DF</i> is the number of observations minus the number of zero weights and the rank of the variance/covariance matrix.
sampling times	The actual times at which the concentration or effects are observed.
SBC	Schwarz Bayesian Criterion. A measure of goodness of fit based on maximum likelihood. When comparing several models for a given data set, the model with the smallest SBC value is regarded as giving the best fit. Appropriate only for comparing models that use the same weighting scheme.

$$SBC = NOBS \times \log(WRSS) + \log(NOBS) \times NPARM$$

For LinMix, the SBC calculation is: $SBC = -2\hat{L}_R + s \log(N - r)$ where \hat{L}_R is the restricted log-likelihood function evaluated at final estimates β

and $\hat{\theta}$; r is $\text{dim}(x)$; s is the number of parameters; and N is the number of observations.

scenario	In PKS, a set that includes one structural model, initial parameters, and/or statistical tests to be applied to a data set for fitting or analysis. See also study .
SD	Standard deviation.
SD of the Logs	Standard deviation of the logs of the Ys.
SE	Standard error.
secondary parameters	Parameters that are functions of the primary model parameters. For example, AUC and half lives are secondary parameters. Note that, in the case of multiple-dose data, secondary parameters dependent on dose use the first dose.
SE - Yhat	Standard error of Predicted Y.
sigmoid	Denotes the “S” shaped nature of certain models.
sort variables	A separate analysis or plot is done for each level of the sort variable(s). If <i>gender</i> and <i>smoking</i> are sort variables, a separate set of analysis results or plots will be created for each of female non-smokers, male non-smokers, female smokers and male smokers.
stripping dose	Dose associated with initial parameter estimates for macro constant models. In these models, which can be written as a sum of exponentials, dose does not appear explicitly in the model. Rather, the macro constants are a function of dose. Thus, when fitting macro constant models with user-specified initial estimates, the user must specify the associated dose. For single dose macro constant models, when WinNonlin determines initial parameter estimates by curve stripping, the stripping dose is identical to the administered dose.
structural model	The set of functions defining the shape of a model, i.e., a structural PK, PD, or PK/PD model. It does not include parameter distributions.
study	In PKS, a study comprises a collection of data that can include observations, dosing, and protocols. A study can also contain one or more scenarios .
summary variables	The variable(s) for which descriptive statistics will be calculated.

T

table definition file	A file (*.TDF) that stores a user-defined table format and settings. See also “ Table definition files ” on page 148.
Tau (τ)	The (assumed equal) dosing interval for steady-state data.
TI	Infusion length.
Tmax	The time of peak concentration.
Tmin	For steady-state data: time of minimum concentration based on samples collected during a dosing interval.
treatment	A combination of one or more drugs given at specific doses and time(s).

U

UIR	Unit impulse response function, describing the PK profile following an instantaneous dose of one unit of drug into the central compartment.
univariate confidence interval	A confidence interval based on standard error. The univariate confidence interval does not take into account correlations with other parameters.
user libraries	Libraries of user-defined ASCII models. See “ User Models ” on page 219.

V

V	Volume of distribution of the central compartment. This is also denoted V_C .
variance	The variance, v , in descriptive statistics: $\sum_{i=1}^n (Y_i - \bar{Y})^2 / (n - 1)$ <p>Note: $SD = \sqrt{v}$</p>
VIF	Variance Inflation Factor, the multiplier of the residual mean square, used in deriving the asymptotic variance of the parameter estimates. Useful in simulations, to find optimal experimental design. Smaller values are better.
Vmax	Theoretical maximum rate of process in Michaelis-Menten kinetics.

Vss An estimate of the volume of distribution at steady state. $V_{ss} = MRTINF \cdot CL$, where MRT = Mean Residence Time and CL= total body clearance. For steady-state data, $V_{ss} = MRTINF \cdot Cl_{ss}$. Not computed for model 200, as calculating MRT from oral data requires estimating Mean Input Time, which requires compartmental methods.

VZ, VZ/F Volume of distribution based on the terminal phase. For extravascular models, the fraction of dose absorbed cannot be estimated. Therefore, Volume for these models is actually Volume/F, where F is the fraction of dose absorbed.

W

weight variable The weight variable is a numeric data column used in LinMix fitting to compensate for systematic differences in variance across observations, e.g., variance that increases with observed value. Each observation is multiplied by the square root of its weight. Weight values should generally be proportional to reciprocals of variances. This variable should not be included with *classification variables*, but can be among *regressors* or used in the model.

weighted computed Y Square root of the weight times Y Predicted.

weighted predicted Y Predicted Y times the square root of the weight.

weighted residual Y Residual Y times the square root of the weight.

weighted - SS Weighted Sum of Squared Residuals. See also *WRSS*.

Westlake confidence interval Confidence intervals that are constrained to be symmetric about 100%.

workspace A workspace saves all open windows and settings in WinNonlin, including model and/or analysis settings.

WRSS Weighted residual sums of squares, an estimate of the variance of the residuals. This is the quantity WNL minimizes during modeling.

X

X variable The *independent variable* to be used in PK modeling or Noncompartmental analysis, usually, time relative to the first dose.

Y

Y variable The *dependent variable* to be used PK modeling or Noncompartmental analysis, generally Concentration.

WinNonlin Model Libraries

Parameterization and equations for WinNonlin compiled and ASCII models

WinNonlin is distributed with an extensive library of models, including most of the commonly used one, two and three compartment pharmacokinetic (PK) models, linear, pharmacodynamic (PD), link, indirect response^A and Michaelis-Menten models, as well as seven “models” for noncompartmental analysis of plasma, urine or drug effect data. The following are included in the library of compiled models and detailed in the sections below, following the syntax noted under “[Notation and conventions](#)” on page 508.

Table B-1. Model libraries

Model	Compiled	ASCII
“Noncompartmental analysis” on page 509	Yes	No
“Pharmacokinetic models” on page 523	Yes	Yes
“Pharmacodynamic models” on page 537	Yes	Yes
“PK/PD link models” on page 540	Yes	No
“Indirect response models” on page 543	Yes	No
“Michaelis-Menten models” on page 546	No	Yes
“Simultaneous PK/PD link models” on page 549	No	Yes
“Linear models” on page 550	Yes	No
“Sigmoidal/dissolution models” on page 551 with IVIVC licence only	Yes	No

Some of the models are provided in ASCII (text) model libraries; others are compiled into machine language. Each PK and PD model is included in both

A. Dayneka NL, Garg V, Jusko W (1993); Jusko W (1990); Nagashima R, O’Reilly RA, Levy G (1969).

ASCII and compiled forms. Compiled models run faster. The ASCII versions provide the user with examples and starting templates for custom models.

Note: When using the ASCII models, be sure to examine the ASCII code to ascertain what dosing constants need to be created.

Although WinNonlin can be used to estimate the parameters in any system of nonlinear equations, these libraries contain classical PK and PD models. Note that it is possible to create custom model libraries, as described under “[User Models](#)” on page 219.

Notation and conventions

[WinNonlin Model Libraries](#) number the central compartment 1, and the exterior to the compartments, zero. First-order rate constants are expressed as K_{ij} , meaning the rate from Compartment i to Compartment j . Thus, K_{01} is the rate at which drug enters the central compartment from outside the system, and K_{10} is the rate at which drug leaves the system from the central compartment.

Some models compute secondary parameters, such as half-lives and areas, based the estimates of the primary parameters. WinNonlin also estimates standard errors for these secondary parameters. For the two and three-compartment models some pharmacokineticists prefer to estimate the micro rate constants (K_{ij}) as primary parameters; others prefer to estimate the macro rate constants (alpha and beta). The WinNonlin libraries contain both forms. Thus, Model 11, for example, is the two-compartment model with first-order input and alpha and beta as secondary parameters; Model 13 is the same model with alpha and beta as primary parameters and the rate constants K_{01} , K_{12} , and K_{21} as secondary parameters.

Since these models are normally used to fit concentration data as a function of time, t represents the independent variable, and $C(t)$ is the mathematical equation being fit to the observations. All of the models require the dose(s) and time(s) of dosing to be input as constants. In addition, for those models defined in terms of the macro constants, the dose associated with the initial parameter estimates (this dose is hereafter referred to as the *stripping dose*) must also be input. The reason for this is as follows. Such models can be written as:

$$C(t) = \sum A_i \times \exp(-\alpha_i \times t)$$

Note that the dose, D , does not appear directly in the expression for $C(t)$. Instead, the intercepts, $\{A_i\}$, are functions of D (as well as the micro constants). To support multiple-dose data, WinNonlin defines the macro constants model as:

$$C(t) = \sum_j D_j \times \frac{A_j}{D} \times \exp(\alpha_j \times t)$$

where D_j denotes the j^{th} administered dose. Writing the model in this way enables WinNonlin to fit multiple dose data for the macro constant models. In order to do so, WinNonlin requires the stripping dose (D).

With the exception of the Noncompartmental Analysis “models,” WinNonlin models assume that the Time variable has been coded such that the time of the first dose is 0. This applies for both WinNonlin modeling and simulation.

The factor that converts dose amount to concentration is denoted as V , volume. In absorption models there is no way to estimate both volume and fraction of dose absorbed, so both of these are included in V (thus V/F is estimated where F is the fraction of the dose that was absorbed).

The presentation of the mathematical equations of the models follows FORTRAN notation. For example, $\exp(-K01 \cdot t)$ is the constant e with an exponent of the negative product of time, t and the rate constant $K01$.

The models with first-order input are presented with and without a lag time. Since this amounts only to a shift on the time axis, the lag time is not explicitly shown in the expression for $C(t)$, and the estimated T_{max} should have the lag time added to it for those models that include a lag time.

Note: For discussion of one and two-compartment models with first-order exchange see Gibaldi and Perrier (1982).

Noncompartmental analysis

The [WinNonlin Model Libraries](#) include seven compiled “models” for non-compartmental analysis of plasma, urine or drug effect data.

Table B-2. Noncompartmental analysis models

Model	Type of data	Dose input	Constants required
200	Plasma or blood	Extravascular	Dose, time of last dose, dose interval (Tau) ^a
201	Plasma or blood	Bolus IV	Dose, time of last dose, dose interval (Tau) ^a
202	Plasma or blood	Constant infusion	Dose, length of infusion, time of last dose, dose interval (Tau) ^a
210	Urine	Extravascular	Dose, time of last dose
211	Urine	Bolus IV	Dose, time of last dose
212	Urine	Constant infusion	Dose, time of last dose
220	Drug effect	Any	Dose time, baseline effect, threshold (optional)

a. Tau is required for steady-state data only.

Models 200-202 can be used for either single-dose or steady-state data. For steady-state data, the computation assumes equal dosing intervals (Tau) for each profile, and that the data are from a “final” dose given at steady-state. Models 210-212 (urine concentrations) assume single-dose data.

NCA output parameters and their computation formulas

NCA output parameters and their computation are covered under:

- “Plasma or serum data” on page 511
- “Urine data” on page 516
- “Sparse sampling (pre-clinical) data” on page 518
- “Drug effect data (model 220)” on page 519

See also “Computational rules for WinNonlin’s noncompartmental analysis engine” on page 183 and “Noncompartmental Analysis” on page 161. For additional reading, see Gouyette (1983) and Chan and Gilbaldi (1982).

Plasma or serum data

Data structure

NCA for blood concentration data requires the following input data:

- Time of each sample
- Plasma or serum concentrations

For extravascular and constant infusion models (models 200 and 202), if the input data do not contain an observation at time zero, WinNonlin will generate one. When using steady-state data, if the concentration at the dosing time is missing, then that concentration is set equal to the minimum concentration observed in the dosing interval.

Output

Models 200-202 estimate the parameters in the following tables.

- [Table B-3, “Parameters that do not require \$\lambda_z\$,” on page 511](#)
- [Table B-4, “Parameters available only if \$\lambda_z\$ is estimable,” on page 512](#)
- [Table B-5, “Calculations for steady-state data,” on page 515](#)

Table B-3. Parameters that do not require λ_z

C0	Initial concentration. Given only for bolus IV models. It is equal to the first observed concentration value if that value occurs at the dose time. Otherwise, it is estimated by back-extrapolating from the first two concentration values according to the rules given under “Insertion of initial time points” on page 184 .
Dosing_time	Time of last administered dose. Assumed to be zero unless otherwise specified. Used mainly with steady-state data, which may code time as the time elapsed since the first dose, or the elapsed time since the time of the first dose.
Tlag	Extravascular input (model 200) only. Tlag is the time prior to the first measurable (non-zero) concentration.
Tmax	Time of maximum observed concentration. For non-steady-state data, the entire curve is considered. For steady-state data, Tmax corresponds to points collected during a dosing interval.
Cmax	Maximum observed concentration, occurring at Tmax.
Cmax_D	Maximum observed concentration divided by dose.

Table B-3. Parameters that do not require λ_z (continued)

Tlast	Time of last measurable (positive) concentration.
Clast	Concentration corresponding to Tlast.
AUClast	Area under the curve from the time of dosing (Dosing_time) to the last measurable concentration.
AUCall	Area under the curve from the time of dosing (Dosing_time) to the time of the last observation. If the last concentration is non-zero AUClast=AUCall. Otherwise, AUCall will be greater than AUClast as it includes the additional area from the last measurable concentration down to zero.
AUMClast	Area under the moment curve from the time of dosing (Dosing_time) to the last measurable concentration.
MRTlast	Mean residence time from the time of dosing (Dosing_time) to the time of the last measurable concentration. For non-infusion models: MRTlast = AUMClast/AUClast. For infusion models: MRTlast = (AUMClast/AUClast) - TI/2.
No_points_ Lambda_z	Number of points used in computing λ_z . If λ_z cannot be estimated, zero.
AUC_%Bac k_Ext	Computed for Bolus IV (model 201) only: the percentage of AUCINF that was due to back extrapolation to estimate C(0). If C2<C1, the C(0) is estimated by log-linear regression of the first two time points; otherwise, C(0) is set to C1.
AUClower _upper	(Optional) user-requested area(s) under the curve from time <i>lower</i> to <i>upper</i> .

Note: Parameters that are extrapolated to infinity, in Table B-4 below, are calculated two ways: Parameters called “*_obs” use the last observed value for *Clast*, and parameters called “*_pred” used the predicted value for *Clast*, which is defined as: $Clast_pred = \exp(intercept - \lambda_z * Tlast)$. The values *intercept* and *lambdaZ* are those values found during the regression for *Lambda_z*. The value for *intercept* is not given in the normal output, but can be obtained by checking **Output intermediate calculations** in the Model Options.

Table B-4. Parameters available only if λ_z is estimable

Rsq	Goodness of fit statistic for the terminal elimination phase.
Rsq_adjusted	Goodness of fit statistic for the terminal elimination phase, adjusted for the number of points used in the estimation of λ_z .

Table B-4. Parameters available only if λ_z is estimable (continued)

Corr_XY	Correlation between time (X) and log concentration (Y) for the points used in the estimation of λ_z .
Lambda_z	First order rate constant associated with the terminal (log-linear) portion of the curve. Estimated by linear regression of time vs. log concentration.
Lambda_z_lower	Lower limit on Time for values to be included in the calculation of λ_z .
Lambda_z_upper	Upper limit on Time for values to be included in the calculation of λ_z .
HL_Lambda_z	Terminal half-life = $\ln(2)/\lambda_z$
AUCINF_obs	AUC from Dosing_time extrapolated to infinity, based on the last observed concentration (obs). See also AUCINF_pred, below. $= \text{AUC}_{\text{last}} + \frac{C_{\text{last}_{\text{obs}}}}{\lambda_z}$
AUCINF_D_obs	AUCINF_obs divided by dose.
AUC_%Extrap_obs	Percentage of AUCINF_obs due to extrapolation from Tlast to infinity. $= \frac{\text{AUCINF}_{\text{obs}} - \text{AUC}_{\text{last}}}{\text{AUCINF}_{\text{obs}}} \cdot 100$
Vz_obs, Vz_F_obs ^a	Volume of distribution based on the terminal phase. $= \frac{\text{Dose}}{\lambda_z \cdot \text{AUCINF}_{\text{obs}}}$
Cl_obs, Cl_F_obs ^a	Total body clearance for extravascular administration. = Dose/AUCINF_obs.
AUCINF_pred	AUC from Dosing_time extrapolated to infinity, based on the last predicted concentration, i.e., concentration at the final observation time estimated using the linear regression performed to estimate Lambda Z. $= \text{AUC}_{\text{last}} + \frac{C_{\text{last}_{\text{pred}}}}{\lambda_z}$
AUCINF_D_pred	AUCINF_pred divided by dose.
AUC_%Extrap_pred	Percentage of AUCINF_pred due to extrapolation from Tlast to infinity. $= \frac{\text{AUCINF}_{\text{pred}} - \text{AUC}_{\text{last}}}{\text{AUCINF}_{\text{pred}}} \cdot 100$

Table B-4. Parameters available only if λ_z is estimable (continued)

Vz_pred, Vz_F_pred ^a	Volume of distribution based on the terminal phase. $= \frac{\text{Dose}}{\lambda_z \cdot \text{AUCINF_pred}}$
Cl_pred, Cl_F_pred ^a	Total body clearance for extravascular administration. $= \text{Dose}/\text{AUCINF_pred}$
AUMCINF_obs	Area under the first moment curve (AUMC) extrapolated to infinity, based on the last observed concentration. $= \text{AUMClast} + \frac{\text{Tlast} \cdot \text{Clast}_{\text{obs}}}{\lambda_z} + \frac{\text{Clast}_{\text{obs}}^2}{\lambda_z^2}$
AUMC_%Extrap_obs	Percent of AUMCINF_obs that is extrapolated. $= \frac{\text{AUMCINF_obs} - \text{AUMClast}}{\text{AUMCINF_obs}} \cdot 100$
AUMCINF_pred	Area under the first moment curve (AUMC) extrapolated to infinity, based on the last predicted concentration, i.e., concentration at the final observation time estimated using the linear regression for Lambda Z. $= \text{AUMClast} + \frac{\text{Tlast} \cdot \text{Clast}_{\text{pred}}}{\lambda_z} + \frac{\text{Clast}_{\text{pred}}^2}{\lambda_z^2}$
AUMC_%Extrap_pred	Percent of AUMCINF_pred that is extrapolated. $= \frac{\text{AUMCINF_pred} - \text{AUMClast}}{\text{AUMCINF_pred}} \cdot 100$
MRTINF_obs	Mean residence time (MRT) extrapolated to infinity. For non-infusion models = AUMCINF_obs/AUCINF_obs. Note that for oral model 200, MRTINF includes Mean Input Time as well as time in systemic circulation. For infusion models = (AUMCINF_obs/AUCINF_obs) - TI/2, where TI represents infusion duration.
MRTINF_pred	Mean residence time (MRT) extrapolated to infinity. For non-infusion models = AUMCINF_pred/AUCINF_pred. Note that for oral model 200, MRTINF includes Mean Input Time as well as time in systemic circulation. For infusion models = (AUMCINF_pred/AUCINF_pred) - TI/2, where TI represents infusion duration.

Table B-4. Parameters available only if λ_z is estimable (continued)

V _{ss_obs}	An estimate of the volume of distribution at steady state = MRTINF*Cl _{ss} , based on the last observed (obs) or last predicted (pred) concentration. Not computed for model 200, as MRTINF for oral models includes Mean Input Time as well as time in systemic circulation and therefore is not appropriate to use in calculating V _{ss} .
V _{ss_pred}	
a. For extravascular models (model 200), the fraction of dose absorbed cannot be estimated; therefore Volume and Clearance for these models are actually Volume/F or Clearance/F where F is the fraction of dose absorbed.	

The following additions and changes are made for steady-state data.

Table B-5. Calculations for steady-state data

Tau	The (assumed equal) dosing interval for steady-state data.
Tmin	Time of minimum concentration sampled during a dosing interval.
Cmin	Minimum concentration between dose time and dose time + Tau (at Tmin).
Cavg	AUC from dose time to Tau divided by (dose time + Tau).
%_Fluctuation	$= 100 \cdot (C_{max} - C_{min}) / C_{avg}$, for Cmin and Cmax between dose time and Tau.
Accumulation Index	$= \frac{1}{1 - e^{-\lambda_z \tau}}$
$Cl_{ss}, Cl_{ss_F^a}$	An estimate of the total body clearance = $\frac{\text{Dose}}{AUC _0^\tau}$
Cmax	Maximum concentration between dose time and dose time + Tau (at Tmax).
MRTINF_obs	Mean residence time (MRT) extrapolated to infinity.
Non-infusion:	$\frac{AUMC _0^\tau + \tau(AUCINF_obs - AUC _0^\tau)}{AUC _0^\tau}$
Infusion:	$\frac{AUMC _0^\tau + \tau(AUCINF_obs - AUC _0^\tau)}{AUC _0^\tau} - \frac{TI}{2}$
where TI represents infusion duration.	

Table B-5. Calculations for steady-state data (continued)

MRTINF_pred	Mean residence time (MRT) extrapolated to infinity.
Non-infusion:	$\frac{AUMC _0^{\tau} + \tau(AUCINF_pred - AUC _0^{\tau})}{AUC _0^{\tau}}$
Infusion:	$\frac{AUMC _0^{\tau} + \tau(AUCINF_pred - AUC _0^{\tau})}{AUC _0^{\tau}} - \frac{TI}{2}$
where TI represents infusion duration.	
$V_z, V_z \cdot F^a$	$= \frac{\text{Dose}}{\lambda_z \cdot AUC _0^{\tau}}$
V_{ss_obs} V_{ss_pred}	An estimate of the volume of distribution at steady state = MRTINF*Clss, based on the last observed (obs) or last predicted (pred) concentration. Not computed for model 200, as MRTINF for oral models includes Mean Input Time as well as time in systemic circulation and therefore is not appropriate to use in calculating V_{ss} .

- a. For extravascular models (model 200), the fraction of dose absorbed cannot be estimated; therefore Volume and Clearance for these models are actually Volume/F or Clearance/F where F is the fraction of dose absorbed.

Urine data

Data structure

NCA for urine data requires the following input data:

- Starting and ending time of each urine collection interval
- Urine concentrations
- Urine volumes

From this data, models 210-212 compute the following for the analysis:

- Midpoint of each collection interval
- Excretion rate for each interval (amount eliminated per unit of time) = (Concentration * Volume) / (Ending time - Starting time)

Output

Models 210-212 estimate the following parameters.

Table B-6. Parameters that do not depend on λ_z

Dosing_time	Time of the last administered dose (assumed to be zero unless otherwise specified).
Tlag	Midpoint prior to the first measurable (non-zero) rate.
Tmax_Rate	Midpoint of collection interval associated with the maximum observed excretion rate.
Max_Rate	Maximum observed excretion rate.
Mid_Pt_last	Midpoint of collection interval associated with Rate_last.
Rate_last	Last measurable (positive) rate.
AURC_last	Area under the urinary excretion rate curve from time 0 to the last measurable rate.
Amount_Recovered	Cumulative amount eliminated. $= \Sigma(\text{Concentration} * \text{Volume})$
Vol_UR	Sum of Urine Volumes
Percent_Recovered	$100 * \text{Amount_Recovered} / \text{Dose}$
AURC_all	Area under the urinary excretion rate curve from time 0 to the last rate. This equals AURC_last if the last rate is measurable.
No_points_lambda_z	Number of points used in the computation of λ_z .
AUClower_upper	(Optional) partial area(s) under the curve from time <i>lower</i> to <i>upper</i> .

Table B-7. Parameters that are estimated only when λ_z is estimable

Rsq	Goodness of fit statistic for the terminal elimination phase.
Rsq_adjusted	Goodness of fit statistic for the terminal elimination phase, adjusted for the number of points used in the estimation of λ_z .
Corr_XY	Correlation between midpoints and log excretion rates for the points used in the estimation of λ_z .
Lambda_z	Lambda Z. First order rate constant associated with the terminal (log-linear) portion of the curve. This is estimated via linear regression of midpoints vs. log excretion rates.
Lambda_z_lower	Lower limit on midpoint for values to be included in λ_z estimation.

Table B-7. Parameters that are estimated only when λ_z is estimable

Lambda_z_upper	Upper limit on midpoint for values to be included in λ_z estimation.
HL_Lambda_z	Terminal half-life = $\ln(2) / \lambda_z$
AURC_INF_obs	Area under the urinary excretion rate curve extrapolated to infinity, based on the last observed excretion rate.
AURC_%Extrap_obs	Percent of AURC_INF_obs that is extrapolated.
AURC_INF_pred	Area under the urinary excretion rate curve extrapolated to infinity, based on the last predicted rate, i.e., excretion rate at the final midpoint estimated using the linear regression for λ_z .
AURC_%Extrap_pred	Percent of AURC_INF_pred that is extrapolated.

Note that AURC_INF is theoretically equal to the amount recovered but will differ due to experimental error.

Sparse sampling (pre-clinical) data

When an NCA model is loaded with the Sparse Sampling option (see “[NCA model selection](#)” on page 162), the data are treated as a special case of plasma or urine concentration data. The NCA engine computes the mean concentration or rate at each unique time value or interval. Using the mean concentration curve across subjects, it estimates the same parameters normally calculated for plasma or urine data, plus those listed below. See “[Sparse sampling computations](#)” on page 190 for additional details of NCA computations with sparse data.

Plasma or serum concentration parameters

Sparse sampling methods for plasma data (models 200-202) compute the following parameters in addition to those listed in “[Plasma or serum data](#)” on page 511.

SE_Cmax	Standard error of data at Tmax (time of maximum mean concentration).
SE_AUClast	Standard error of the area under the mean concentration curve from dose time to Tlast, where Tlast is the time of last measurable (positive) mean concentration.
SE_AUCall	Standard error of the area under the mean concentration curve from dose time to the final observation time.

Urine concentration parameters

Sparse sampling methods for urine data (models 210-212) compute the following parameters in addition to those listed under “[Urine data](#)” on page 516.

SE_Max_Rate	Standard error of the data at the time of maximum mean rate.
SE_AURC_last	Standard error of the area under the mean urinary excretion rate curve from dose time through the last interval that has a measurable (positive) mean rate.
SE_AURC_all	Standard error of the area under the mean urinary excretion rate curve from dose time through the final interval.

Individuals by time

This sheet includes the individual subject data, along with N (number of non-missing observations), the mean and standard error for each unique time value (plasma data) or unique midpoint value (urine data).

Drug effect data (model 220)

Data structure

NCA for drug effect data requires the following input data variables:

- Dependent variable, i.e., response or effect values (continuous scale)
- Independent variable, generally the time of each observation. For non-time-based data, such as concentration-effect data, take care to interpret output parameters such as “Tmin” (independent variable value corresponding to minimum dependent variable value) accordingly.

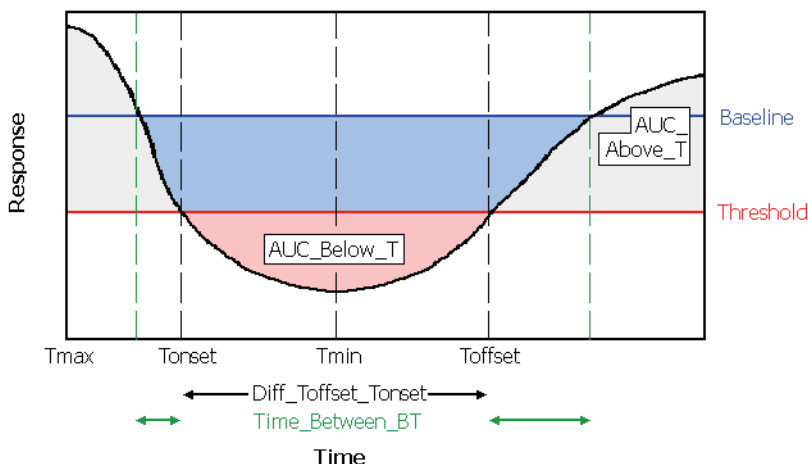
and the following constants:

- Dose time (relative to observation times), entered in the [Dosing regimen](#)
- Baseline response value, entered in the [Therapeutic response](#) tab of the Model Properties or pulled from a Dosing worksheet as described below
- Threshold response value (optional), entered in the [Therapeutic response](#) tab of the Model Properties

By default, the user must enter a baseline response value unless working from a PKS study that includes a Dosing worksheet. In that case, if no baseline is provided by the user, WinNonlin will use the response value at dose time as baseline. If the data set does not include a measurement at dose time, the user must supply a baseline value (see [“Therapeutic response” on page 170](#)). If the dosing time is not specified ([“Dosing regimen” on page 173](#)), the dosing time is assumed to be zero and WinNonlin will insert the point (0, baseline).

Instead of Lambda Z calculations, as computed in NCA PK models, the NCA PD model can calculate the slope of the time-effect curve for specific data ranges in each profile. Unlike Lambda Z, these slopes are reported as their actual value rather than their negative. See [“Lambda Z or slope estimation settings” on page 164](#) for more information.

Like other NCA models, the PD model can compute partial areas under the curve. Partial areas are set up as described under [“Partial areas” on page 186](#).



Estimated parameters

Output parameter names use the following conventions.

- B = baseline effect value (discussed above).
- T = user-supplied threshold effect value.
- “Above” means towards increasing Y values, even for inhibitory effects.

Table B-8. Estimated parameters for model 220

Baseline	Baseline response (Y) value supplied by the user, or, for PKS studies with no user-supplied baseline value, effect value at dose time.
AUC_Above_B	Area under the response curve that is above the baseline (dark gray areas in the above diagram).
AUC_Below_B	Area that is below the baseline and above the response curve (combined blue and pink areas in the above diagram).
AUC_Net_B	= AUC_Above_B - AUC_Below_B. This is likely to be a negative value for inhibitory effects.
Slope1 (or 2)	Slope of the first (or second) segment of the curve. See “Lambda Z or slope estimation settings” on page 164 .
Rsq_Slope1 (or 2)	Goodness of fit statistic for slope 1 or 2.
Rsq_adj_Slope1 (or 2)	Goodness of fit statistic for slope 1 or 2, adjusted for the number of points used in the estimation.
Corr_XY_Slope1 (or 2)	Correlation between time (X) and effect (or log effect, for log regression) (Y) for the points used in the slope estimation.
No_points_Slope1 (or 2)	The number of data points included in calculation of slope 1 or 2.
Slope1_lower or Slope2_lower	Lower limit on Time for values to be included in the slope calculation.
Slope1_upper or Slope2_upper	Upper limit on Time for values to be included in the slope calculation.
Rmax	Maximum observed response value.
Rmin	Minimum observed response value.
Time_Above_B	Total time that Response >= Baseline.
Time_Below_B	Total time that Response < Baseline.
Time_%Below_B	= $100 * \text{Time_Below_B} / (\text{Tfinal} - \text{Tdose})$ where Tfinal is the final observation time and Tdose is dosing time.

When a threshold value is provided, model 220 also computes the following.

Table B-9. Additional parameters for model 220 with threshold

Threshold	Threshold value used.
-----------	-----------------------

Table B-9. Additional parameters for model 220 with threshold (continued)

AUC_Above_T	Area under the response curve that is above the threshold value (combined light and dark gray areas in the above diagram).
AUC_Below_T	Area that is below the threshold and above the response curve (pink area in the above diagram).
AUC_Net_T	= AUC_Above_T - AUC_Below_T.
Time_Above_T	Total time that Response \geq Threshold.
Time_Below_T	Total time that Response $<$ Threshold.
Time_%Below_T	= $100 * \text{Time_Below_T} / (\text{Tlast} - \text{Tdose})$ where Tlast is the final observation time and Tdose is dosing time.
Tonset	Time that the response first crosses the threshold coming from the direction of the baseline value, as shown in the above diagram. Tonset = Tdose if the first response value is across the threshold, relative to baseline. The time will be interpolated using the calculation method selected in the model options. (See “NCA settings” on page 179.) ^a
Toffset	Time greater than Tonset at which the curve first crosses back to the baseline side of threshold, as shown in the diagram above. ^a
Diff_Toffset_Tonset	= Toffset - Tonset
Time_Between_BT	Total time spent between baseline and threshold (sum of length of green arrows in diagram).
Tmax	Time of maximum observed response value (Rmax).
Tmin	Time of minimum observed response value (Rmin).
AUClower_upper	(Optional) user-requested area(s) under the curve from time <i>lower</i> to time <i>upper</i> .

a. Use caution in interpreting Tonset and Toffset for noisy data if Baseline and Threshold are close together.

Note: For PD data, the default and recommended method for AUC calculation is the Linear Trapezoidal with Linear Interpolation (set as described under “NCA settings” on page 179.) Use caution with log trapezoidal since areas are calculated both under the curve and above the curve. If the log trapezoidal rule is appropriate for the area under a curve, then it would underestimate the area over the curve. For this reason, the program will use linear trapezoidal for area where the curve is below baseline when computing AUC_Below_B, and similarly for threshold and AUC_Below_T.

Pharmacokinetic models

The [WinNonlin Model Libraries](#) include 19 pharmacokinetic (PK) models. Each is available in both compiled and ASCII text forms.

Model	Input	Number of compartments	Parameterization	Lag time	Elimination rate
Model 1	IV-bolus	1	-	no	1st order
Model 2	IV-infusion	1	-	no	1st order
Model 3	1st order	1	-	no	1st order
Model 4	1st order	1	-	yes	1st order
Model 5	1st order	1	K10 = K01	no	1st order
Model 6	1st order	1	K10 = K01	yes	1st order
Model 7	IV-bolus	2	micro	no	1st order
Model 8	IV-bolus	2	macro	no	1st order
Model 9	IV-infusion	2	micro	no	1st order
Model 10	IV-infusion	2	macro	no	1st order
Model 11	1st order	2	micro	no	1st order
Model 12	1st order	2	micro	yes	1st order
Model 13	1st order	2	macro	no	1st order
Model 14	1st order	2	macro	yes	1st order
Model 15	IV-bolus plus IV-infusion	1	micro	no	1st order
Model 16	IV-bolus plus IV-infusion	2	micro	no	1st order
Model 17	IV-bolus plus IV-infusion	2	macro	no	1st order
Model 18	IV-bolus	3	macro	no	1st order
Model 19	IV-infusion	3	macro	no	1st order

Note: All models except models 15-17 accept multiple dose data.

The models shown in this section give equations for compartment 1 (central).

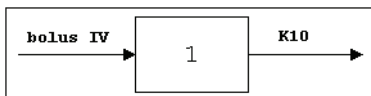
ASCII models and files

WinNonlin PK models are available as compiled models, and also in the ASCII library files PK1.LIB - PK10.LIB, as follows.

Model #	File name
1, 2	PK1.LIB
3, 4	PK2.LIB
5, 6	PK3.LIB
7, 8	PK4.LIB
9, 10	PK5.LIB
11, 12	PK6.LIB
13, 14	PK7.LIB
15, 16	PK8.LIB
17, 18	PK9.LIB
19	PK10.LIB

Model 1

One compartment with bolus input and first-order output



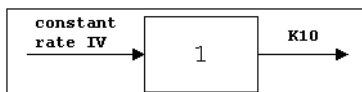
$$C(T) = D/V$$

$$C(T) = (D/V) \exp(-K10 \cdot T)$$

Required constants	Estimated parameters	Secondary parameters	Clearance estimated	Clearance secondary
N doses	V = Volume	$AUC = D/V/K10$	V	AUC
dose N	$K10$ = elimination rate	$K10$ half-life	CL	$K10$ half-life
time of dose N		$C_{max} = D/V$		C_{max}
(Repeat for each dose)		CL		$K10$
		$AUMC$		$AUMC$
		MRT		MRT
		V_{ss}		V_{ss}

Model 2

One-compartment with constant IV input, first-order absorption



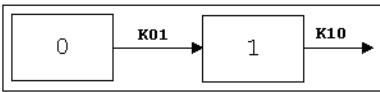
$$C(T) = \left(\frac{D}{TI}\right) \frac{1}{VK_{10}} [\exp(-K_{10}TSTAR) - \exp(-K_{10}T)]$$

where *TI* = infusion length; *TSTAR* = *T-TI* for *T*>*TI*; *TSTAR* = 0 for *T*≤*TI*.

Required constants	Estimated parameters	Secondary parameters	Clearance estimated	Clearance secondary
<i>N</i> doses	V = Volume	AUC = D/V/K10	V	AUC
dose <i>N</i>	K10 = elimination rate	K10 half-life	CL	K10 half-life
start time <i>N</i>		Cmax = C(TI)		Cmax
end time <i>N</i>		CL		K10
(Repeat for each dose)		AUMC		AUMC
		MRT		MRT
		Vss		VSS

Model 3

One-compartment with first-order input and output, no lag time



$$C(T) = \frac{DK_{01}}{V(K_{01} - K_{10})} [\exp(-K_{10}T) - \exp(-K_{01}T)]$$

Required constants	Estimated parameters	Secondary parameters	Clearance estimated	Clearance secondary
<i>N</i> doses	V_F	AUC = D/V/K10	V_F	AUC
dose <i>N</i>	K01=absorption rate	K01 half-life	K01	K01 half-life
time of dose <i>N</i>	K10=elimination rate	K10 half-life	CL_F	K10 half-life
(Repeat for each dose)		CL_F		K10
		Tmax = time of Cmax = $\frac{\ln(K01/K10)}{(K01 - K10)}$		Tmax
		Cmax = max concentration		Cmax

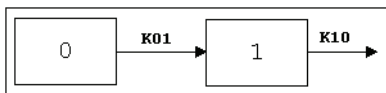
Model 4

One-compartment with first-order input and output with lag time

Identical to [Model 3](#), with an additional estimated parameter: Tlag = lag time.

Model 5

One-compartment, equal first-order input and output, no lag time



$$C(T) = (D/V) K \cdot T \cdot \exp(-K \cdot T)$$

Required constants	Estimated parameters	Secondary parameters	Clearance estimated	Clearance secondary
N doses	V_F	AUC = D/V/K	V_F	AUC
dose N	K=absorption and elimination rate	K half-life	CL_F	K half-life
time of dose N		CL_F		K
(Repeat for each dose)		Tmax = time of Cmax = 1/K		Tmax
		Cmax = max concentration		Cmax

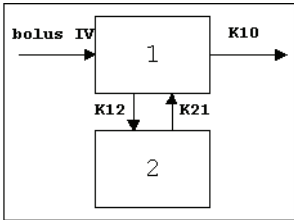
Model 6

One-compartment, equal first-order input and output with lag time

Identical to [Model 5](#), with an additional estimated parameter: Tlag = lag time.

Model 7

Two-compartment with bolus input and first-order output; micro-constants as primary parameters



$C(T) = A \exp(-ALPHA \cdot T) + B \exp(-BETA \cdot T)$

where:

$A = (D/V) (ALPHA - K21)/(ALPHA - BETA)$

$B = (D/V) (BETA - K21)/(ALPHA - BETA)$

where $-ALPHA$ and $-BETA$ ($ALPHA > BETA$) are the roots of the quadratic equation: $(r \cdot r + (K12 + K21 + K10) \cdot r + K21 \cdot K10 = 0)$.

Required constants	Estimated parameters	Secondary parameters	Clearance estimated	Clearance secondary
N doses	$V1 = \text{Volume1}$	$AUC = D/N/K10$	$V1$	AUC
dose N	$K10 = \text{elimination rate}$	$K10 \text{ half-life}$	CL	$K10 \text{ half-life}$
time of dose N	$K12 = \text{transfer rate, 1 to 2}$	$ALPHA$	$V2$	$ALPHA$
(Repeat for each dose)	$K21 = \text{transfer rate, 2 to 1}$	$BETA$	$CLD2$	$BETA$
		$ALPHA \text{ half-life}$		$ALPHA \text{ half-life}$
		$BETA \text{ half-life}$		$BETA \text{ half-life}$
		A		A
		B		B
		$C_{\text{max}} = D/V$		C_{max}
		CL		$K10$
		$AUMC$		$AUMC$
		MRT		MRT
		V_{ss}		V_{ss}
		$V2$		$K12$
		$CLD2$		$K21$

Model 8

Two-compartment with bolus input and first-order output; macro-constants as primary parameters

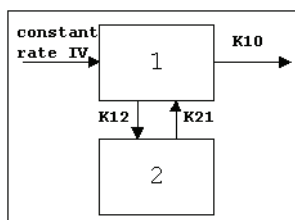
As [Model 7](#) with macroconstants as the primary (estimated) parameters.

Required constants	Estimated parameters	Secondary parameters	
stripping dose	A	AUC = A/ALPHA + B/BETA	V1
<i>N</i> doses	B	K10 half-life	CL
dose <i>N</i>	ALPHA	ALPHA half-life	AUMC
time of dose <i>N</i>	BETA	BETA half-life	MRT
(Repeat for each dose)		K10	Vss
		K12	V2
		K21	CLD2
		Cmax	

Clearance parameters are not available for Model 8.

Model 9

Two-compartment with constant IV input and first-order output; micro-constants as primary parameters



$$C(T) = A_1 [\exp(-ALPHA \cdot T) - \exp(-ALPHA \cdot TSTAR)] + B_1 [\exp(-BETA \cdot T) - \exp(-BETA \cdot TSTAR)]$$

where *TI* = infusion length; *TSTAR* = *T* - *TI* for *T* > *TI*; *TSTAR* = 0 for *T* ≤ *TI*.

$$A_1 = \frac{D}{TI(V)(ALPHA - BETA)ALPHA} (K21 - ALPHA)$$

$$B_1 = \frac{-D}{TI(V)(ALPHA - BETA)BETA} (K21 - BETA)$$

and $-ALPHA$ and $-BETA$ ($ALPHA > BETA$) are the roots of the quadratic equation: $r \cdot r + (K12 + K21 + K10) \cdot r + K21 \cdot K10 = 0$.

Required constants	Estimated parameters	Secondary parameters	Clearance estimated	Clearance secondary
N doses	$V1 = \text{Volume1}$	$AUC = D/V/K10$	$V1$	AUC
dose N	$K10 = \text{elimination rate}$	$K10 \text{ half-life}$	CL	$K10 \text{ half-life}$
start time N	$K12 = \text{transfer rate, 1 to 2}$	$ALPHA$	$V2$	$ALPHA$
end time N	$K21 = \text{transfer rate, 2 to 1}$	$BETA$	$CLD2$	$BETA$
(Repeat for each dose)		$ALPHA \text{ half-life}$		$ALPHA \text{ half-life}$
		$BETA \text{ half-life}$		$BETA \text{ half-life}$
		A		A
		B		B
		$C_{max} = C(TI)$		C_{max}
		CL		$K10$
		$AUMC$		$AUMC$
		MRT		MRT
		V_{ss}		V_{ss}
		$V2$		$K12$
		$CLD2$		$K21$

A and B are the zero time intercepts following an IV injection.

Model 10

Two-compartment with constant IV input and first-order output; macroconstants as primary parameters

As [Model 9](#) with macroconstants as the primary (estimated) parameters.

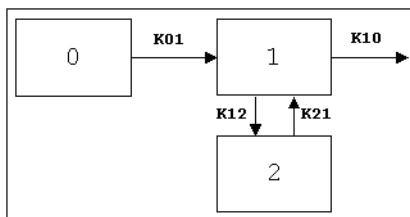
Required constants	Estimated parameters	Secondary parameters	
N doses	V1	K10	Cmax
dose N	K21	K12	CL
start time N	ALPHA	K10 half-life	AUMC
end time N	BETA	AUC	MRT
(Repeat for each dose)		ALPHA half-life	Vss
		BETA half-life	V2
		A	CLD2
		B	

A and B are the zero time intercepts following an IV injection.

Clearance parameters are not available in Model 10.

Model 11

Two-compartment with first-order input, first-order output, no lag time and micro-constants as primary parameters



$$C(T) = A \exp(-ALPHA \cdot T) + B \exp(-BETA \cdot T) + C \exp(-K01 \cdot T)$$

where:

$$A = \frac{\left(\frac{D}{V}\right) K01 (K21 - ALPHA)}{(ALPHA - BETA)(ALPHA - K01)}$$

$$B = \frac{\left(-\frac{D}{V}\right) K01 (K21 - BETA)}{(ALPHA - BETA)(BETA - K01)}$$

$$C = \frac{\left(\frac{D}{V}\right) K_{01}(K_{21} - K_{01})}{(BETA - K_{01})(ALPHA - K_{01})}$$

and $-ALPHA$ and $-BETA$ ($ALPHA > BETA$) are the roots of the quadratic equation: $r \cdot r + (K_{12} + K_{21} + K_{10}) \cdot r + K_{21} \cdot K_{10} = 0$.

Required constants	Estimated parameters	Secondary parameters	Clearance estimated	Clearance secondary
N doses	V1_F	AUC = D/V/K10	V1_F	AUC
dose N	K01 = absorption rate	K01 half-life	K01	K01 half-life
time of dose N	K10 = elimination rate	K10 half-life	CL_F	K10 half-life
(Repeat for each dose)	K12 = transfer rate, 1 to 2	ALPHA	V2_F	ALPHA
	K21 = transfer rate, 2 to 1	BETA	CLD2_F	BETA
		ALPHA half-life		ALPHA half-life
		BETA half-life		BETA half-life
		A		A
		B		B
		CL_F		K10
		V2_F		K12
		CLD2_F		K21
		Tmax ^a		Tmax
		Cmax ^a		Cmax

a. Estimated for the compiled model only.

Model 12

Two-compartment with first-order input, first-order output, lag time and micro-constants as primary parameters

As [Model 11](#), with an additional estimated parameter: Tlag = lag time.

Model 13

Two-compartment with first-order input, first-order output, no lag time and macro-constants as primary parameters

As [Model 11](#), with macroconstants as the primary (estimated) parameters.

Required constants	Estimated parameters	Secondary parameters	
Stripping dose	A	K10	BETA half-life
N doses	B	K12	V1_F
dose N	K01 = absorption rate	K21	CL_F
time of dose N	ALPHA	AUC = $D/V/K10$	V2_F
(Repeat for each dose)	BETA	K01 half-life	CLD2_F
	Note: $C = -(A + B)$	K10 half-life	Tmax ^a
		ALPHA half-life	Cmax ^a

a. Estimated for the compiled model only.

Clearance parameters are not available for Model 13.

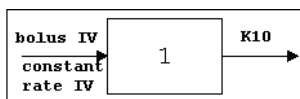
Model 14

Two-compartment with first-order input, first-order output, lag time and macro-constants as primary parameters

As [Model 13](#), with an additional estimated parameter: TLAG = lag time.

Model 15

One compartment with simultaneous bolus IV and constant IV infusion



$$C_B(T) = (D_b/V) \exp(-K10 \cdot T)$$

$$C_{IV}(T) = \left(\frac{D_{IV}}{TI} \right) \frac{1}{V(K10)} [\exp(-K10 \cdot TSTAR) - \exp(-K10 \cdot T)]$$

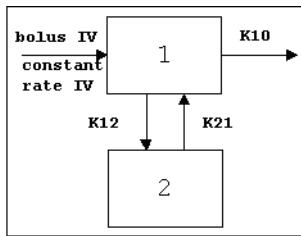
$$C(T) = C_B(T) + C_{IV}(T)$$

where TI = infusion length; $TSTAR = T - TI$ for $T > TI$; $TSTAR = 0$ for $T \leq TI$

Required constants	Estimated parameters	Secondary parameters	Clearance estimated	Clearance secondary
bolus dose	V = Volume	CL	V	K10
IV dose	K10	K10 half-life	CL	K10 half-life
length of infusion = TI				

Model 16

Two compartment with simultaneous bolus IV and constant infusion input, micro constants as primary parameters



$$C_B(T) = A_1 \exp(-ALPHA \cdot T) + B_1 \exp(-BETA \cdot T)$$

where:
$$A_1 = \frac{D_B}{V} \frac{(ALPHA - K21)}{(ALPHA - BETA)}$$

$$B_1 = \frac{-D_B}{V} \frac{(BETA - K21)}{(ALPHA - BETA)}$$

and:

$$C_{IV}(T) = A_2 [\exp(-ALPHA \cdot T) - \exp(-ALPHA \cdot TSTAR)] \\ + B_2 [\exp(-BETA \cdot T) - \exp(-BETA \cdot TSTAR)]$$

where:

$$A_2 = \frac{D_{IV}}{TI(V)} \frac{(K21 - ALPHA)}{(ALPHA - BETA)ALPHA}$$

$$B_2 = \frac{-D_{IV}}{TI(V)} \frac{(K21 - BETA)}{(ALPHA - BETA)BETA}$$

where TI = infusion length; $TSTAR = T - TI$ for $T > TI$; $TSTAR = 0$ for $T \leq TI$

$$C(T) = C_B(T) + C_{IV}(T)$$

and $-ALPHA$ and $-BETA$ ($ALPHA > BETA$) are the roots of the quadratic equation: $r \cdot r + (K12 + K21 + K10) \cdot r + K21 \cdot K10 = 0$.

Required constants	Estimated parameters	Secondary parameters	Clearance estimated	Clearance secondary
bolus dose	V1	K10 half-life	V1	K10 half-life
IV dose	K10	ALPHA	CL	ALPHA
length of infusion (= TI)	K12	BETA	V2	BETA
	K21	ALPHA half-life	CLD2	ALPHA half-life
		BETA half-life		BETA half-life
		A		A
		B		B
		CL		K10
		V2		K12
		CLD2		K21

Model 17

Two compartment with simultaneous bolus IV and constant infusion input and macro constants as primary parameters

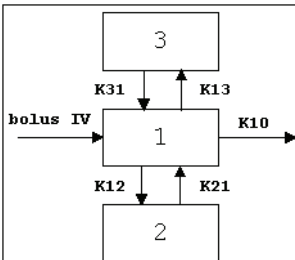
As [Model 16](#) with macroconstants as the primary (estimated) parameters.

Required constants	Estimated parameters	Secondary parameters	
stripping dose	A	K10	BETA half-life
bolus dose	B	K12	V1
IV dose	ALPHA	K21	CL
length of infusion (= TI)	BETA	K10 half-life	V2
		ALPHA half-life	CLD2

Clearance parameters are not available in Model 17.

Model 18

Three-compartment with bolus input, first-order output and macro constants as primary parameters



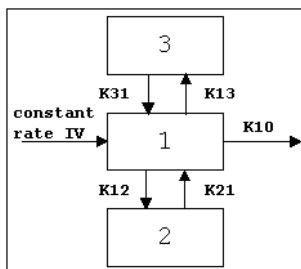
$C(T) = A \exp(-ALPHA \cdot T) + B \exp(-BETA \cdot T) + C \exp(-GAMMA \cdot T)$

Required constants	Estimated parameters	Secondary parameters	
stripping dose	A	Cmax	GAMMA half-life
N doses	B	V1	AUC
dose N	C	K21	CL
time of dose N	ALPHA	K31	AUMC
(Repeat for each dose)	BETA	K10	MRT
	GAMMA	K12	Vss
		K13	V2
		K10 half-life	CLD2
		ALPHA half-life	V3
		BETA half-life	CLD3

Clearance parameters are not available in Model 18.

Model 19

Three compartment model with constant IV infusion; macro constants as primary parameters



$$\begin{aligned}
 C(T) = & A_1 [\exp(-ALPHA \cdot TSTAR) - \exp(-ALPHA \cdot T)] \\
 & + B_1 [\exp(-BETA \cdot TSTAR) - \exp(-BETA \cdot T)] \\
 & + C_1 [\exp(-GAMMA \cdot TSTAR) - \exp(-GAMMA \cdot T)]
 \end{aligned}$$

where:

$$A_1 = \frac{\left(\frac{D}{TI}\right)(K21 - ALPHA)(K31 - ALPHA)}{V(ALPHA)(GAMMA - ALPHA)(BETA - ALPHA)}$$

$$B_1 = \frac{\left(\frac{D}{TI}\right)(K21 - BETA)(K31 - BETA)}{V(BETA)(GAMMA - BETA)(ALPHA - BETA)}$$

$$C_1 = \frac{\left(\frac{D}{TI}\right)(K21 - GAMMA)(K31 - GAMMA)}{V(GAMMA)(ALPHA - GAMMA)(BETA - GAMMA)}$$

where TI = infusion length; $TSTAR = T - TI$ for $T > TI$; $TSTAR = 0$ for $T \leq TI$

Required constants	Estimated parameters	Secondary parameters		
N doses	V1	Cmax	K10 half-life	MRT
dose N	K21	K10	ALPHA half-life	Vss
start time of dose N	K31	K12	BETA half-life	V2

Required constants	Estimated parameters	Secondary parameters		
end time of dose N	ALPHA	K13	GAMMA half-life	CLD2
(Repeat for each dose)	BETA	A	AUC	V3
	GAMMA	B	CL	CLD3
		C	AUMC	

Clearance parameters are not available in Model 19.

A, B and C are the zero time intercepts following an IV injection.

Pharmacodynamic models

Load and set up pharmacodynamic (PD) models as described under “[Loading the model](#)” on page 195 and “[Model properties](#)” on page 198. Available PD models in the [WinNonlin Model Libraries](#) include the following.

Model	Description	Effect at C=0	Effect at C=infinity
Model 101	Simple Emax Model	0	Emax
Model 102	Simple Emax Model	E0	Emax
Model 103	Inhibitory Effect Emax Model	Emax	0
Model 104	Inhibitory Effect Emax Model	Emax	E0
Model 105	Sigmoid Emax Model	0	Emax
Model 106	Sigmoid Emax Model	E0	Emax
Model 107	Inhibitory Effect Sigmoid Emax Model	Emax	0
Model 108	Inhibitory Effect Sigmoid Emax Model	Emax	E0

ASCII library

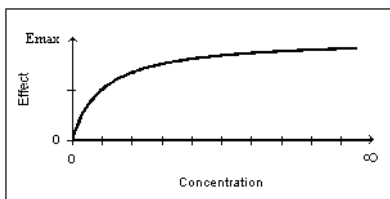
These models are available as compiled models and also as ASCII library files. The ASCII library files are stored in the WINNONLN\LIB subdirectory.

Model #	File name
101, 102	PK51.LIB
103, 104	PK52.LIB
105, 106	PK53.LIB

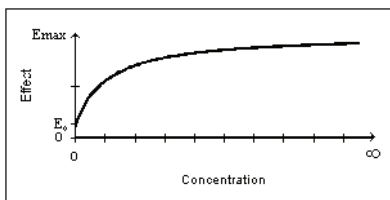
Model #	File name
107, 108	PK54.LIB

Table B-10. Model notation

Term	Definition
E _{max}	Maximum effect
E ₀	Baseline effect
EC ₅₀	Drug concentration required to produce 50% of the maximal effect (E _{max} - E ₀)
E _{max} - E ₀	Drug-induced maximal effect
Gamma	Shape parameter

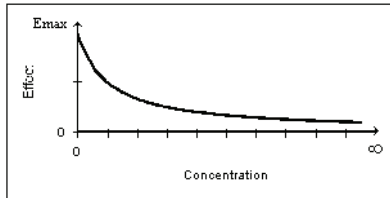
Model 101*Simple Emax model*

$$E = (E_{\max} \cdot C) / (C + EC_{50})$$

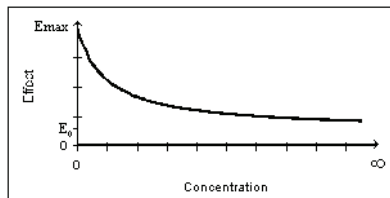
Estimated parameters: E_{max}, EC₅₀**Model 102***Simple Emax model with a baseline effect parameter*

$$E = E_0 + (E_{\max} - E_0) [C / (C + EC_{50})]$$

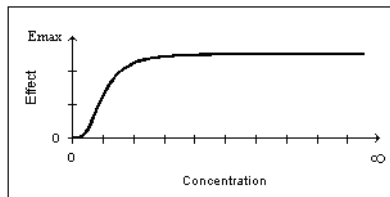
Estimated parameters: E_{max}, E₀, EC₅₀**Secondary parameter:** E_{max} - E₀

Model 103*Inhibitory effect model*

$$E = E_{\max} [1 - (C / (C + EC_{50}))]$$

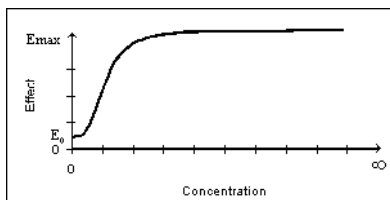
Estimated parameters: Emax, EC₅₀**Model 104***Inhibitory effect model with a baseline effect parameter*

$$E = E_{\max} - (E_{\max} - E_0) [C / (C + EC_{50})]$$

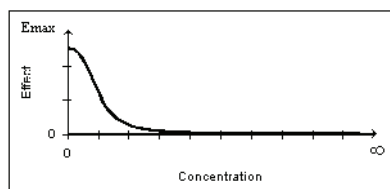
Estimated parameters: Emax, EC₅₀, E₀**Secondary parameter:** Emax - E₀**Model 105***Sigmoid Emax model*

$$E = (E_{\max} \cdot C^{\gamma}) / (C^{\gamma} + EC_{50}^{\gamma})$$

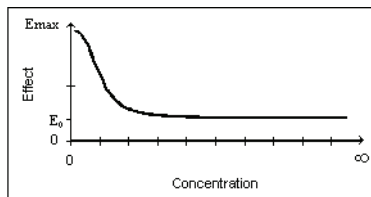
Estimated parameters: Emax, EC₅₀, Gamma

Model 106*Sigmoid Emax model with a baseline effect parameter*

$$E = E_0 + (E_{\max} - E_0) [C^\gamma / (C^\gamma + EC_{50}^\gamma)]$$

Estimated parameters: Emax, EC₅₀, E₀, Gamma**Secondary parameter:** Emax - E₀**Model 107***Sigmoid inhibitory effect model*

$$E = E_{\max} [1 - C^\gamma / (C^\gamma + EC_{50}^\gamma)]$$

Estimated parameters: Emax, EC₅₀, Gamma**Model 108***Sigmoid inhibitory effect model with a baseline effect parameter*

$$E = E_{\max} - (E_{\max} - E_0) [C^\gamma / (C^\gamma + EC_{50}^\gamma)]$$

Estimated parameters: Emax, EC₅₀, E₀, Gamma**Secondary parameter:** Emax - E₀**PK/PD link models**

When pharmacological effects are seen immediately and are directly related to the drug concentration, a pharmacodynamic model is applied to characterize the relationship between drug concentrations and effect. When the pharmacologic response takes time to develop and the observed response is not

directly related to plasma concentrations of the drug a *link model* is usually applied to relate the pharmacokinetics of the drug to its pharmacodynamics.

The PK/PD Link models can use any combination of WinNonlin compiled [Pharmacokinetic models](#) and [Pharmacodynamic models](#). The PK model is used to predict concentrations, and these concentrations are then used as input to the PD model. Thus, the PK data are not modeled; the PK/PD Link models treat the pharmacokinetic parameters as fixed, and generate concentrations at the effect site to be used by the PD model. Model parameter information will be required for the PK model in order to simulate the concentration data.

Note: To fit PK and PD data simultaneously, see “[Simultaneous PK/PD link models](#)” on page 549.

Notation

Term	Definition
Emax	Maximum effect
EC ₅₀	Concentration required to produce one-half of the maximal drug effect.
E ₀	Effect at time 0
Ke0	Rate of drug loss from the effect compartment. The Ke0 T _{1/2} is the half life of the amount of time required to collapse the hysteresis curve.
Emax–E ₀	Drug induced maximal effect
Gamma	Shape parameter

Linking PK and PD models

WinNonlin can generate concentrations at effect sites using any PK model in the compiled library, and using these concentrations in any PD model.

To link any PK and PD models:



1. Click the **PK/PD/NCA Analysis Wizard** button on the tool bar or choose **Tools>PK/PD/NCA Analysis Wizard** from the menus.
2. Select **PK/PD Link** in the Model Types dialog.

3. Select a PK model from those detailed under “[Pharmacokinetic models](#)” on page 523.
4. Click **Next**.
5. Select a PD model from those detailed under “[Pharmacodynamic models](#)” on page 537.
6. Click **Next**. The Selection Summary dialog appears.
7. Click **Finish**.

To assign parameter values for the PK model:

1. Enter the Data Variables, Model Parameters, Dosing Regimen/ Constants, and model options as for other compartmental models, as described under “[Model properties](#)” on page 198.
2. Click the **Model Parameters** button or choose **Model>Model Parameters** from the menus. Notice that the Model Parameters dialog now has a two tabs, one for the PK Model and one for the PD Model.
3. Select the **PK Model** tab.
4. Enter the parameter values. The initial parameter values must be entered.
5. Click **Apply** or **OK**.

Note: The units for PK concentration can be entered in the **PK Concentration Unit** field or set using the **Units Builder** button.

To assign initial estimates and bounds for the PD model:

1. Select the **PD Model** tab.
2. Enter initial estimates and bounds.
3. Click **OK**. The model parameters are set and the dialog closes.

Output parameters

The output parameters for the PD models differ from the parameters for these models when they are not linked. [Table B-11](#) lists the PD model parameters.

Table B-11. Link model output parameters

PD model	Estimated parameters	Secondary parameter
101 Simple Emax	Emax, KE0, ECE ₅₀	
102 Simple Emax with baseline effect	Emax, E0, ECE ₅₀ , KE0	Emax-E ₀
103 Inhibitory effect	Emax, ECE ₅₀ , KE0	
104 Inhibitory effect with a baseline effect	Emax, E0, ECE ₅₀ , KE0	Emax-E ₀
105 Sigmoid Emax	Emax, Gamma, ECE ₅₀ , KE0	
106 Sigmoid Emax with a baseline effect	Emax, E0, Gamma, ECE ₅₀ , KE0	Emax-E ₀
107 Sigmoid inhibitory effect	Emax, Gamma, ECE ₅₀ , KE0	
108 Sigmoid inhibitory effect model with a baseline effect	Emax, E0, Gamma, ECE ₅₀ , KE0	Emax-E ₀

Indirect response models

When pharmacological effects are seen immediately and are directly related to the drug concentration, a pharmacodynamic model is applied to characterize the relationship between drug concentrations and effect. When the pharmacologic response takes time to develop and the observed response is not directly related to plasma concentrations of the drug a *link model* is usually applied to relate the pharmacokinetics of the drug to its pharmacodynamics.

One kind of link model available in WinNonlin is the family of indirect pharmacodynamic response (IPR) models proposed by Jusko and co-workers Jusko (1990) and Dayneka, Garg, and Jusko (1993).

The indirect response models differ from other models in the [WinNonlin Model Libraries](#) in that they use a PK model to predict concentrations, and then use these concentrations as input to the indirect response model.

Four basic models have been developed for characterizing indirect pharmacodynamic responses after drug administration. These models are based on drug effects (inhibition or stimulation) on the factors controlling either the input or the dissipation of drug response.

Note: These models are available only as compiled models

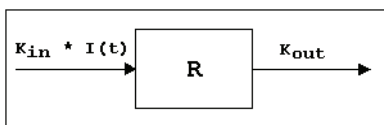
Models

Notation

Term	Definition
R	Measured response to a drug
k_{in}	The zero order constant for the production of response
k_{out}	The first order rate constant for loss of response
C_p	Plasma concentration of a drug
C_e	Drug concentration at the effect site
IC_{50}	The drug concentration that produces 50% of maximum inhibition
E_{max}	Maximum effect
EC_{50}	The drug concentration that produces 50% of maximum stimulation

Model 51

Inhibition of input



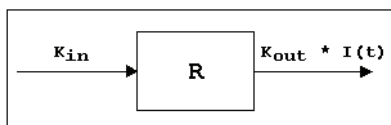
$$\frac{dR}{dt} = k_{in} \left(1 - \frac{C_p}{C_p + IC_{50}} \right) - k_{out} R$$

Estimated parameters

k_{in} k_{out} IC_{50}

Model 52

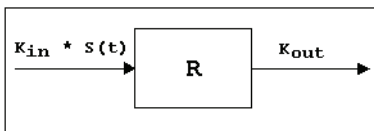
Inhibition of output



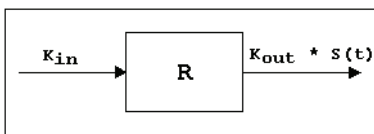
$$\frac{dR}{dt} = k_{in} - k_{out} \left(1 - \frac{C_p}{C_p + IC_{50}} \right) R$$

Estimated parameters

k_{in} k_{out} IC_{50}

Model 53*Stimulation of input*

$$\frac{dR}{dt} = k_{in} \left(1 + \frac{Emax \cdot C_p}{C_p + EC_{50}} \right) - k_{out} R$$

Estimated parametersKin Kout EC₅₀ Emax**Model 54***Stimulation of output*

$$\frac{dR}{dt} = k_{in} - k_{out} \left(1 + \frac{Emax \cdot C_p}{C_p + EC_{50}} \right) R$$

Estimated parametersKin Kout EC₅₀ Emax**Running an indirect response model***To select an Indirect Response model:*

1. Click the **PK/PD/NCA Analysis Wizard** button on the tool bar or select the **PK/PD/NCA Analysis Wizard** from the **Tools** menu. The PK/PD Analysis Wizard appears, displaying the Model Types dialog.
2. Select **Indirect Response** from the selections in the **Compartmental Models** group box. The WinNonlin Compiled Models dialog appears.
3. Select a PK model.
4. Click **Next**. The list of Indirect Response Models appears.
5. Select an Indirect Response model.
6. Click **Next**. The Selection Summary box appears.
7. If everything described in this summary box is correct, click **Finish**. (Otherwise, click **Back** and return to correct the mistake.) The modeling window appears.

Note: Enter the Data Variables, Dosing Regimen/Constants, and Model Options precisely as for other models.

To assign parameter values for the PK model:

1. Enter the Data Variables, Model Parameters, Dosing Regimen/ Constants, and model options as for other compartmental models, as described under “[Model properties](#)” on page 198.
2. Click the **Model Parameters** button or choose **Model>Model Parameters** from the menus. Notice that this dialog now has tabs for both the PK Model and the indirect response model.

It is possible to choose to have WinNonlin generate the initial parameter values or to specify user-supplied initial values.

3. Select the **PK Model** tab located under the parameters table.
4. Enter the parameter values for the PK model.

Note: Initial parameter values for the PK model must be specified.

5. (Optional) Enter the PK units in the PK Concentration Unit field.
6. Click **Apply**.

Initial estimates and bounds can be specified for the IPR model.

To assign initial estimates and bounds for the IPR model:

1. Select the **IPR Model** tab in the Model Properties dialog.
2. Enter initial estimates and bounds.
3. Click **Apply** or **OK** to specify the settings and close the dialog.

Michaelis-Menten models

The WinNonlin library contains four Michaelis-Menten models:

Model	Description
Model 301	One compartment with bolus input and Michaelis-Menten output
Model 302	One compartment with constant IV input and Michaelis-Menten output
Model 303	One compartment with first order input, Michaelis-Menten output and no time lag
Model 304	One compartment with first order input, Michaelis-Menten output and a lag time

These models are stored as ASCII library files in \LIB\MM.LIB. As with all WinNonlin ASCII models, be sure to examine the ASCII code to determine which dosing constants need to be created.

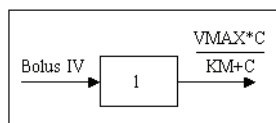
WinNonlin assumes that the time of the first dose is zero. For these models, times in the data set must correspond to the dosing times, even if these times contain no observations. In this case, include a column for Weight on the data set, and weight these observations as 0.

These models parameterize Vmax in terms of concentration per unit time.

For discussion of difficulties inherent in fitting the Michaelis-Menten models see [Tong and Metzler \(1980\)](#) (1980) and [Metzler and Tong \(1981\)](#) (1981).

Model 301

One-compartment with bolus input and Michaelis-Menten output



$C(T)$ is the solution to the differential equation:

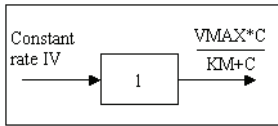
$$dC/dt = (-VMAX \cdot C)/(KM + C),$$

with initial condition $C(0) = D/V$.

Required constants	Estimated parameters	Secondary parameters
N doses	V = Volume	$AUC = (D/V)/(D/V/2 + KM)/VMAX$
Dose N	VM = max elimination rate	
Time of dose N	KM = Michaelis constant	

Model 302

One-compartment with constant IV input and Michaelis-Menten output



$C(T)$ is the solution to the differential equations:

$$dC/dt = (D/TI/V) - VMAX \cdot C / (KM + C) \text{ for } T \leq TI,$$

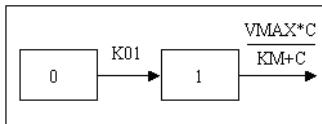
$$dC/dt = - VMAX \cdot C / (KM + C) \text{ for } T > TI,$$

with initial condition $C(0) = 0$.

Required constants	Estimated parameters	Secondary parameters
N doses	V = Volume	(none)
Dose N	VM = max elimination rate	
Start time for dose N	KM = Michaelis constant	
End time for dose N		

Model 303

One-compartment with first-order input, Michaelis-Menten output and no lag time



$C(T)$ is the solution to the differential equation:

$$dC/dt = K01 \cdot (D/V) \exp(-K01 \cdot T) - VM \cdot C / (KM + C),$$

with initial condition $C(0) = 0$.

Required constants	Estimated parameters	Secondary parameters
N doses	V_F	$K01$ half-life
Dose N	$K01$ = absorption rate	
Time of dose N	VM = max elimination rate	
	KM = Michaelis constant	

Model 304

One-compartment with first-order input, Michaelis-Menten output and lag time

As [Model 303](#), with one additional estimated parameter: Tlag = lag time.

Simultaneous PK/PD link models

WinNonlin has a library of link models that simultaneously fit PK and PD data. As it is difficult to rewrite these models for changes in the PK model, but easy to rewrite them for a different PD model, this library includes each PK models with PD [Model 105](#). These models are described below.

Data

The data for these models must contain the concentration and effect data in the same column, with a function variable in a separate column to distinguish the two. Function 1 should include the time and concentration data, Function 2 should include the time and effect data. An example is shown below.

Time	Response	Function
0	0	1
0.5	5	1
1	15	1
...
36	1	1
0	148	2
0.5	145	2
1	123	2
...
36	121	2

ASCII library

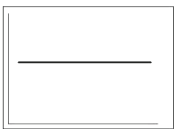
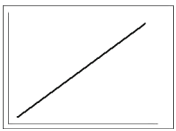
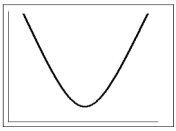
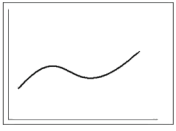
These models are stored as ASCII library files in the \LIB subdirectory. As with all of the WinNonlin ASCII models, be sure to examine the ASCII code to see what dosing constants need to be created.

Model	File	PK model	PK model description
401	KEO1.LIB	Model 1	1 compartment - bolus input
402	KEO1.LIB	Model 2	1 compartment - constant IV input
403	KEO2.LIB	Model 3	1 compartment - extravascular input
404	KEO2.LIB	Model 4	1 compartment - extravascular input, with a lag time
405	KEO3.LIB	Model 5	1 compartment - extravascular input, equal input and output rates
406	KEO3.LIB	Model 6	1 compartment - extravascular input, equal input and output with a lag time
407	KEO4.LIB	Model 7	2 compartment - bolus input, defined in microconstants
408	KEO4.LIB	Model 8	2 compartment - bolus input, defined in macroconstants
409	KEO5.LIB	Model 9	2 compartment - constant IV input, defined in microconstants
410	KEO5.LIB	Model 10	2 compartment - constant IV input, defined in macroconstants
411	KEO6.LIB	Model 11	2 compartment - 1st order input, defined in microconstants
412	KEO6.LIB	Model 12	2 compartment - 1st order input, defined in macroconstants, includes a lag time
413	KEO7.LIB	Model 13	2 compartment - 1st order input, defined in macroconstants
414	KEO7.LIB	Model 14	2 compartment - 1st order input, defined in macroconstants, includes a lag time
415	KEO8.LIB	Model 15	1 compartment - bolus input + constant infusion
416	KEO8.LIB	Model 16	2 compartment - bolus input + constant infusion, defined in microconstants
417	KEO9.LIB	Model 17	2 compartment - bolus input + constant infusion, defined in macroconstants
418	KEO10.LIB	Model 18	3 compartment - bolus input, defined in macroconstants
419	KEO10.LIB	Model 19	3 compartment - constant infusion, defined in macroconstants

Linear models

WinNonlin includes a selection of models that are linear in the parameters. See also “[Linear Mixed Effects Modeling](#)” on page 327 for more sophisticated linear models.

Table B-12. Linear models

Plot	Model #	Name	Model	Estimated parameters (units)
	501	Constant	$y(t) = \text{constant}$	CONSTANT (y-unit)
	502	Linear	$y(t) = \text{INT} + \text{SLOPE} * t$	INT (y-unit) SLOPE (y-unit / t-unit)
	503	Quadratic	$y(t) = A0 + A1*t + A2*t^2$	A0 (y-unit) A1 (y-unit / t-unit) A2 (y-unit / t-unit ²)
	504	Cubic	$y(t) = A0 + A1*t + A2*t^2 + A3*t^3$	A0 (y-unit) A1 (y-unit / t-unit) A2 (y-unit / t-unit ²) A3 (y-unit / t-unit ³)

Where y-unit is the preferred unit of the y (concentration) variable and t -unit is the preferred unit of the t (time) variable. See “Units” on page 210 to set preferred units for the output parameters.

The linear models require no dosing information (constants).

Note: The default computation method, Gauss-Newton (Hartley) (**Model Options>PK Settings**), and the model option “Do Not Use Bounds” (**Model Options>Model Parameters**) are recommended for all linear models.

Sigmoidal/dissolution models

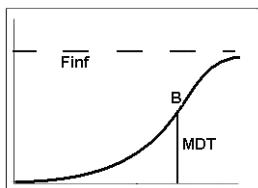
If you have purchased and installed an IVIVC Toolkit for WinNonlin license, the WinNonlin model library includes the following models for smoothing dissolution data as part of analysis of *in vivo-in vitro* correlations (IVIVC): Hill, Weibull, Double Weibull and Makoid-Banakar.

For these models, the parameter Tlag may be strongly correlated with other parameters, especially the slope factor b . It may not be possible to estimate all parameters simultaneously; you may need to fix a parameter while estimating the others.

See “Units” on page 210 to set preferred units for the output. Model parameters can be estimated or set to a fixed value. See “Model parameters” on page 199 for details. The IVIVC models require no dosing (constants).

Model 601

Hill



$$y(t) = \text{int} + ((F_{\text{inf}} - \text{int}) * t^b) / (\text{MDT}^b + t^b)$$

where the estimated parameters are:

F_{inf} = amount released at time infinity, using the preferred units for y

MDT = mean dissolution time, in the preferred units for x (time)

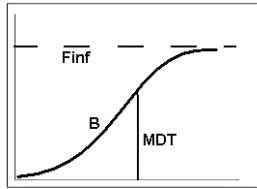
b = slope factor (no units)

int = y -intercept, zero (default) or estimated intercept, using the preferred units for y .

This model includes an optional lag time in the form of a step function, i.e.:

$$y(t) = \text{int} + ((F_{\text{inf}} - \text{int}) * U(t - t_{\text{lag}})^b) / (\text{MDT}^b + U(t - t_{\text{lag}})^b)$$

where $U(x) = x$ if $x \geq 0$; $U(x) = 0$ if $x < 0$.

Model 602*Weibull*

$$y(t) = \text{int} + (\text{Finf} - \text{int}) * (1 - \exp[-(t / \text{MDT})^b])$$

where the estimated parameters are:

Finf = amount released at time infinity, using the preferred units for *y*

MDT = mean dissolution time, in the preferred units of *x* (time)

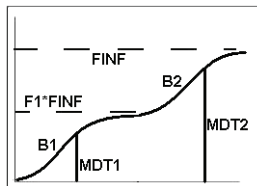
b = slope factor (no units)

int = *y*-intercept, zero (default) or estimated intercept, using the preferred units for *y*.

This model includes an optional lag time in the form of a step function, i.e.:

$$y(t) = \text{int} + (\text{Finf} - \text{int}) * (1 - \exp[-(U(t - t_{\text{lag}}) / \text{MDT})^b])$$

where $U(x) = x$ if $x \geq 0$; $U(x) = 0$ if $x < 0$.

Model 603*Double Weibull*

$$y(t) = \text{int} + f1 * (\text{Finf} - \text{int}) (1 - \exp[-(t / \text{MDT1})^{b1}]) \\ + (1 - f1) * (\text{Finf} - \text{int}) (1 - \exp[-(t / \text{MDT2})^{b2}])$$

where the estimated parameters are:

f1 = weighting factor, setting the fraction due to each Weibull (no units)

Finf = amount released at time infinity, using the preferred units for *y*

MDT1, *MDT2* = mean dissolution time for each Weibull, (*x* units)

$b1, b2$ = slope factors (no units)

int = y-intercept, zero (default) or estimated intercept, using the preferred units for y.

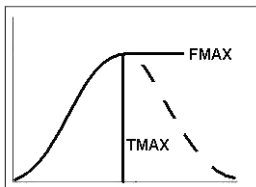
This model includes an optional lag time in the form of a step function, i.e.:

$$y(t) = int + f1 * (Finf - int) (1 - \exp[-(U(t - t_{lag}) / MDT1)^{b1}]) \\ + (1-f1) * (Finf - int) (1 - \exp[-(U(t - t_{lag}) / MDT2)^{b2}])$$

where $U(x) = x$ if $x \geq 0$; $U(x) = 0$ if $x < 0$.

Model 604

Makoid-Banakar



$$y(t) = int + (Fmax - int) * (t/Tmax)^b * \exp[b * (1 - t/Tmax)], \text{ for } t \leq Tmax$$

$$y(t) = Fmax, \text{ for } t > Tmax$$

where the estimated parameters are:

$Fmax$ = maximum y value, using the preferred units for y

$Tmax$ = time of maximum y value, using the preferred units for x (time)

b = slope factor (no units)

int = y-intercept, zero (default) or estimated intercept, using the preferred units for y.

This model includes an optional lag time in the form of a step function, i.e.:

$$y(t) = int + (Fmax - int) * [U(t - t_{lag})/Tmax]^b * \exp[b * (1 - U(t - t_{lag})/Tmax)]$$

where $U(x) = x$ if $x \geq 0$; $U(x) = 0$ if $x < 0$.

Worksheet Functions

Syntax, arguments, and examples for functions in WinNonlin worksheets

WinNonlin worksheets support the functions detailed in [Table C-1](#), wherein:

- *Number* is a number or reference to a cell that contains a number, e.g., B5.
- *Number list* is a comma-separated list of up to 30 numbers or cell references.
- *Serial number* refers to the stored value for a date or time entered in a cell.
- *Value list* is a comma-separated list of up to 30 values or cell references.
- *Expression list* is a comma-separated list of up to 30 expressions.
- *Significance* is the multiple to which to round.
- *Precision* is the number of decimal places allowed.
- *Text* represents a text string.
- [] denotes optional arguments.

Table C-1. Worksheet functions

Function(arguments)	Description
ABS(number)	Absolute value of a number.
ACOS(number)	Returns the arc cosine of a number.
ACOSH(number)	Returns the inverse hyperbolic cosine of a number.
ADDRESS(row, column, ref_type[,a1] [,sheet])	Creates a cell address as text.
AND(logical_list)	Returns True if all arguments are true; False if at least one argument is false.
ASC(text)	In DBCS systems this function returns a copy of text in which the double-byte characters are converted to single-byte characters, if possible. Characters that cannot be converted are left unchanged.
ASIN(number)	Returns the arcsine of a number.

Table C-1. Worksheet functions (continued)

Function(arguments)	Description
ASINH(number)	Returns the inverse hyperbolic sine of a number.
ATAN(number)	Returns the arctangent of a number.
ATAN2(y, x)	Returns the arctangent of the specified coordinates.
ATANH(number)	Returns the inverse hyperbolic tangent of a number.
AVERAGE(number list)	Returns the arithmetic mean of the supplied numbers.
CEILING(number, significance)	Rounds a number up to the nearest multiple of a specified significance.
CHAR(number)	Returns a character that corresponds to the supplied ASCII code.
CHOOSE(index, item_list)	Returns a value from a list of numbers based on the index number supplied.
CLEAN(text)	Removes all nonprintable characters from the supplied text.
CODE(text)	Returns a numeric code representing the first character of the supplied text.
COLUMN(reference)	Returns the column number of the supplied reference.
COLUMNS(range)	Returns the number of columns in a range reference.
CONCATENATE(text1, text2,...)	Joins several text items into one item.
COS(number)	Returns the cosine of an angle.
COSH(number)	Returns the hyperbolic cosine of a number.
COUNT(value list)	Returns the number of values in the supplied list.
COUNTA(expression list)	Returns the number of non-blank values in the supplied list.
COUNTIF(range, criteria)	Returns the number of cells within a range that meet the given criteria.
DATE(year, month, day)	Returns the serial number of the supplied date.
DATEVALUE(text)	Returns the serial number of a date supplied as a text string. Example DATEVALUE ("3/6/94") returns 34399.
DAY(text)	Returns the day of the month that corresponds to the given date. Example: DAY(34399) returns 6; DAY(6-21-94) returns 21.
DB(cost, salvage, life, period [,months])	Returns the real depreciation of an asset for a specific period of time using the fixed-declining balance method.
DDB(cost, salvage, life, period [, factor])	Returns the depreciation of an asset for a specific period of time using the double-declining balance method or a declining balance factor.
DOLLAR(number [, precision])	Returns the specified number as text, using the local currency format and the supplied precision.
EVEN(number)	Rounds the specified number up to the nearest even integer.

Table C-1. Worksheet functions (continued)

Function(arguments)	Description
EXACT(expression1, expression2)	Compares two expressions for identical, case-sensitive matches. True is returned if the expressions are identical; False is returned if they are not.
EXP(number)	Returns e raised to the specified power.
FACT(number)	Returns the factorial of the specified number.
FIND(search_text, text [, start_position])	Searches for a string of text within another text string and returns the character position at which the search string first occurs.
FINDB(search_text, text [, start_position])	Searches for a string of text within another text string and returns the byte position at which the search string first occurs.
FIXED(number [,precision][,no_commas])	Rounds a number to the supplied precision, formats the number in decimal format, and returns the result as text. Example: FIXED(2000.5, 3) returns 2,000.500. FIXED(2009.5, -1, 1) returns 2010.
FLOOR(number, significance)	Rounds a number down to the nearest multiple of specified significance.
FV(interest, nper, payment [, pv] [, type])	Returns the future value of an annuity based on regular payments and a fixed interest rate.
HLOOKUP(search_item, search_range, row_index)	Searches the top row of a table for a value and returns the contents of a cell in that table that corresponds to the location of the search value.
HOUR(serial_number)	Returns the hour component of the specified time in 24-hour format.
IF(condition, true_value, false_value)	Tests the condition and returns the specified value.
INDEX(reference [, row] [, column] [, range_number])	Returns the contents of a cell from a specified range.
INDIRECT(ref_text [, a1])	Returns the contents of the cell referenced by the specified cell.
INT(number)	Rounds the supplied number down to the nearest integer.
IPMT(interest, per, nper, pv, [fv], [type])	Returns the interest payment of an annuity for a given period, based on regular payments and a fixed periodic interest rate.
IRR(cash_flow [, guess])	Returns internal rate of return for a series of periodic cash flows.
ISBLANK(reference)	Determines if the specified cell is blank.
ISLOGICAL(expression)	Determines if the specified expression returns a logical value.
ISNONTEXT(expression)	Determines if the specified expression is not text.
ISNUMBER(expression)	Determines if the specified expression is a number.
ISREF(expression)	Determines if the specified expression is a range reference.
ISTEXT(expression)	Determines if the specified expression is text.

Table C-1. Worksheet functions (continued)

Function(arguments)	Description
LEFT(text [, num_chars])	Returns the left-most characters from the specified text string. Examples: LEFT("2nd Quarter") returns 2; LEFT("2nd Quarter", 3) returns 2nd.
LEFTB(text [,num_bytes])	Returns the left-most byte from the specified text string.
LEN(text)	Returns the number of characters in the supplied text string. Examples: LEN("3rd Quarter") returns 11; LEN("1-3") returns 3.
LENB(text)	Returns the number of bytes in the supplied text string.
LN(number)	Returns the natural logarithm of a number.
LOG(number, [,base])	Returns the logarithm of a number to the specified base, omitting the base assumes a base of 10.
LOG10(number)	Returns the base 10 logarithm of a number.
LOWER(text)	Changes alphabetic characters in the specified string to lower case.
MAX(number list)	Returns the largest value in the specified list of numbers.
MID(text, start_position, num_chars)	Returns the specified number of characters from a text string, beginning with the specified starting position. Example: MID("07/04/96", 4, 2) returns 04.
MIDB(text, start_position, num_bytes)	Returns the specified number of bytes from a text string, beginning with the specified starting position.
MIN(number list)	Returns the smallest value in the specified list of numbers.
MINUTE(number list)	Returns the minute that corresponds to the supplied date.
MIRR(cash_flows, finance_rate, reinvest_rate)	Returns the modified internal rate of return for a series of periodic cash flows.
MOD(number, divisor)	Returns the remainder after dividing a number by a specified divisor.
MONTH(serial_number)	Returns the month that corresponds to the supplied date.
N(value)	Tests the supplied value and returns the value if it is a number.
NOW()	Returns the current date and time as a serial number.
NPER(interest, pmt, pf [, fv] [, type])	Returns the number of periods of an investment based on regular periodic payments and a fixed interest rate.
NPV(discount_rate, value_list)	Returns the net present value of an investment based on a series of periodic payments and a discount rate.
ODD(number)	Rounds the specified number up to the nearest odd integer.
OR(logical_list)	Returns True if at least one of a series of logical arguments is true.
PI()	Returns the value of p. (Empty parentheses are required).

Table C-1. Worksheet functions (continued)

Function(arguments)	Description
PMT(interest, nper, pv [, fv] [, type])	Returns the periodic payment of an annuity, based on regular payments and a fixed periodic interest rate.
PPMT(interest, per, nper, pv, [fv], [type])	Returns the principle paid on an annuity for a given period.
PRODUCT(number list)	Multiplies a list of numbers and returns the result.
PV(interest, nper, pmt [, fv] [, type])	Returns the present value of an annuity, considering a series of constant payments made over a regular payment period.
RAND()	Returns a number selected randomly from a uniform distribution greater than or equal to 0 and less than 1. (Empty parentheses are required).
RATE(nper, pmt, pv [, fv] [, type] [, guess])	Returns the interest rate per period of an annuity, given a series of constant cash payments made over a regular payment period.
REPLACE(orig_text, start_position, num_chars, repl_text)	Replaces part of a text string with another text string. Example: REPLACE("For the year: 1996", 18, 1, "7") returns "For the year: 1997".
REPLACEB(orig_text, start_position, num_bytes, repl_text)	Replaces part of a text string with another text string.
REPT(text, number)	Repeats a text string the specified number of times.
RIGHT(text [, num_chars])	Returns the right-most characters from the given text string.
RIGHTB(text [, num_chars])	Returns the right-most bytes from the given text string.
ROUND(number, precision)	Rounds the given number to the supplied number of decimal places.
ROUNDDOWN(number, number-OfDigits)	Rounds a number down.
ROUNDUP(number, number-Of-Digits)	Rounds the given number up to the supplied number of decimal places.
ROWS(range)	Returns the number of rows in a range reference.
SEARCH(search_text, text [, start position])	Locates the position of the first character of a specified text string within another text string. Example: SEARCH("/", "07/04/96") returns 3.
SEARCHB(search_text, text [, start_position])	Locates the position of the first byte of a specified string within another string.
SECOND(serial_number)	Returns the second that corresponds to the supplied date.
SIGN(number)	Returns the sign of the specified number.
SIN(number)	Returns the sine of the supplied angle.

Table C-1. Worksheet functions (continued)

Function(arguments)	Description
SINH(number)	Returns the hyperbolic sine of the specified number.
SLN(cost, salvage, life)	Returns the depreciation of an asset for a specific period of time using the straight-line balance method.
SQRT(number)	Returns the square root of the specified number.
STDEV(number list)	Returns the standard deviation of a list of as many as 30 numbers.
SUBSTITUTE(text, old_text, new_text [, instance])	Replaces a specified part of a text string with another text string. Example: SUBSTITUTE("07-04-97", "-", "/") returns 07/04/97.
SUM(number list)	Returns the sum of the specified numbers.
SUMIF(range, criteria, sum_range)	Returns the sum of the specified cells based on the given criteria.
SUMPRODUCT(range_list)	Multiplies the cells in the given ranges and returns the sum of those products.
SUMSQ(number list)	Squares each of the supplied numbers and returns the sum of the squares.
SYD(cost, salvage, life, period)	Returns the depreciation of an asset for a specified period using the sum-of-years method.
TAN(number)	Returns the tangent of the specified angle.
TANH(number)	Returns the hyperbolic tangent of a number.
TEXT(number,format)	Returns the given number as text, using the specified formatting.
TIME(hour, minute, second)	Returns a serial number for the supplied time.
TIMEVALUE(text)	Returns a serial number for the supplied text representation of time.
TODAY()	Returns the current date as a serial number.
TRIM(text)	Removes all spaces from text except single spaces between words.
TRUNC(number [,precision])	Truncates the given number to an integer.
TYPE(expression)	Returns the argument type of the given expression.
UPPER(text)	Changes the characters in the specified string to uppercase characters.
VAR(number list)	Returns the variance of a population based on a sample of up to 30 values.
VDB(cost, salvage, life, start_period, end_period [, factor] [, method])	Returns the depreciation of an asset for a specified period using a variable method of depreciation.
VLOOKUP(search_item, search_range, column_index)	Searches the first column of a table for a value and returns the contents of a cell in that table that corresponds to the location of the search value.
WEEKDAY(serial_number)	Returns the day of the week that corresponds to the supplied date.
YEAR(serial_number)	Returns the year that corresponds to the supplied date.

Commands, Arrays and Functions

Commands and syntax for ASCII text user models

Notation

Items enclosed in brackets, i.e., [], are optional.

The symbol “|” means “or.”

E

BASELINE	COMMAND
Usage	BASE N
Description	Sets the (optional) baseline value for use analyzing effect data with NCA model 220. See “Drug effect data (model 220)” on page 519 for details.
Example	BASE 0
BEGIN	COMMAND
Usage	BEGIN
Description	Indicates that all commands have been specified and processing should start.
Example	BEGIN
BTIME	COMMAND
Usage	BTIME lower_val, upper_val, numexcl, #, #, etc.

Description	Selects points within the Lambda Z time range to be excluded from computations of Lambda Z. In the following example, Lambda Z is to be computed using times in the range of 12 to 24, excluding the concentrations at 16 and 20 hours. These “excluded” points will, however, still be used in the computation of the pharmacokinetic parameters.
Example	BTIME 12, 24, 2, 16, 20
Notes	If there are no times to be excluded, set numexcl to zero or leave it blank. Do not mark a point for exclusion by setting the case weight to 0 as that will also exclude this data point from the calculation of the PK parameters.

F

CARRYALONG COMMAND

G

Usage	CARRY #, etc.
Description	Specifies variables from the input data grid to be copied into each of the output worksheets. These variables are not required for the analysis.
Examples	CARRY 3 CARR 1 CARRYALONG 2
Notes	See the NCARRY command. <i>How Carry Along Data is Migrated to Output Multigrids</i> The data for carry along columns is moved to all output worksheets from PK (PK/PD) or NCA analyses, with the exception of the last worksheet, named Settings. For all output worksheets other than the Summary Table, the carry along data will appear in the right-most column(s) of the worksheet. The first cell of data for any given profile within a carry along column will be copied over the entire profile range. There is one exception: generation of the “Summary Table” worksheets. The carry along columns will be the first columns, preceded only by any sort key columns. Also, as there are almost the same number of observations in the raw data set as in the Summary Table, all the cells within a profile of a carry along column are copied over to the Summary Table—not just the first value. There is one special case here as well: an NCA without a first data point of t=0, c=0. When such a data point does not exist on the raw data set, depending on which NCA model is selected, the core program may generate a pseudo-point of t=0, c=0 to support all necessary calculations. In this event, the first cell within a carry along column is copied 'up' to the core-generated data value of 0,0.

COMMANDS MODEL BLOCK

Usage	COMMANDS Statements END
Description	The COMMANDS block allows WinNonlin commands to be permanently inserted into a model. Following the COMMANDS statement, any WinNonlin command such as NPARAMETERS, NCONSTANTS, PNAME, etc., may be given. To end the COMMAND section use the END statement.
Example	COMMANDS NCON 2 NPARM 4 PNAME 'A', 'B', 'Alpha', 'Beta' END

CON ARRAY

Usage	CON(N)
Description	Contains the values of the constants, such as dosing information, used to define the model. The index N indicates the Nth constant.
Example	CON (3)
Note	The values specified by the CONSTANT command, described below, correspond to CON(1), CON(2), etc. respectively.

CONSTANTS COMMAND

Usage	CONSTANTS [are =] C1, C2, ..., CN
Description	Specifies the values of the constants (such as dosing information) required by the model. The arguments C1, C2, ..., CN are the values.
Examples	CONSTANTS 20, 2.06, -1 CONS are 5, 2
Note	The NCONSTANTS command must appear before CONSTANTS.

CONVERGEN
CE COMMAND

Usage	CONVERGENCE [is =] C
Description	Specifies the convergence criterion, C. $0 < C < 0.1$. The default value is 10^{-4} .
Examples	CONVERGENCE = 1.e-6 CONV .001

Note	<p>The test for convergence is</p> $\left \frac{SS_{new} - SS_{old}}{SS_{old}} \right < C, \quad \text{where SS is the residual sum of squares.}$ <p>When CONVERGENCE = 0, convergence checks and halvings are turned off (useful for maximum likelihood estimates).</p>
------	--

G**DATA COMMAND**

Usage	DATA [[are in,] 'path']
Description	Begins data input and optionally specifies an external data file. If arguments are omitted, the data are expected to follow the DATA statement. If the arguments are used, path is the full path of the data set, up to 64 characters.
Examples	<p>DATA</p> <p>DATA 'C:\MYLIB\MYFILE.DAT'</p>

DATE COMMAND

Usage	DATE
Description	Places the current date in the upper right corner of each page in ASCII output.
Example	DATE

DHEADERS COMMAND

Usage	DHEADERS
Description	Statement indicating that data included within the model file contain variable names as the first row.
Example	DHEADERS
Note	If a DNAMES command is also included, it will override the names in the data.

DIFFERENTIAL MODEL BLOCK

Usage	<p>DIFFERENTIAL</p> <p>Statements</p> <p>END</p>
Description	Statements in the DIFFERENTIAL - END block define the system of differential equations to fit. The values of the equations are assigned to the DZ array.

Example	<pre> DIFFERENTIAL DZ(1) = -(K1+KE)*Z(1) + K2*Z(2) DZ(2) = K1*Z(1) - K2*Z(2) END </pre>
Note	NDERIVATIVES must be used to set the number of differential equations to fit. NFUNCTIONS specifies the number of differential equations and/or other functions with associated data.

DINCREMENT COMMAND

T

Usage	DINCREMENT [is =] D
Description	Specifies the increment to use in the estimation of the partial derivatives. The argument D is the increment. $0 < D < 0.1$. The default value is 0.0001
Examples	<pre> DINCREMENT is 0.001 DINC 0.005 </pre>

DNAMES COMMAND

Usage	DNAMES [are =] 'name1', 'name2', ..., 'nameN'
Description	Assigns names to the columns on the data set. The arguments 'name1', 'name2', ..., 'nameN' are comma- or space-separated variable names. Each may use up to 8 letters, numbers or underscores, and must begin with a letter.
Example	DNAMES 'Subject', 'Time', 'Conc'
Note	The names appear in the output and may also be used in the modeling code. If both DNAMES and DHEADERS are included, DNAMES takes precedence.

DTA ARRAY

Usage	DTA(N)
Description	Returns the value of the Nth column for the current observation.
Example	<pre> T = DTA(1) W = DTA(3) </pre>

DTIME COMMAND

Usage	DTIME #
Description	Specifies the time of the last administered dose. This value will appear on the Final Parameters table as Dosing_time. Computation of the final parameters will be adjusted for this dosing time.

	Example	DTIME 48
--	---------	----------

DZ ARRAY

Usage	DZ(N)
Description	Contains the current values of the differential equations. The index N specifies the equation, in the order listed in the DIFFERENTIAL block.
Example	$DZ(3) = -P(3) * Z(3)$
Note	The maximum value of N is specified by the NDERIVATIVES command.

H

F VARIABLE

Usage	F
Description	Returns the value of the function for the current observation.
Example	$F = A * \exp(-B * X)$

FINISH COMMAND

Usage	FINISH
Description	Included for compatibility with PCNonlin Version 4. FINISH should immediately follow the BEGIN command.
Example	BEGIN FINISH

FNUMBER COMMAND

Usage	FNUMBER [is =] N
Description	Indicates the column in the data set that identifies data for the FUNCTION block when NFUNCTIONS > 1. The function variable values must correspond to the FUNCTION numbers, i.e., 1 for FUNC 1, 2 for FUNC 2, etc.
Examples	FNUMBER is 3 FNUM 3
Note	FNUMBER should be specified prior to DATA .

FUNCTION	MODEL BLOCK
Usage	FUNCTION N Statements END
Description	The FUNCTION - END block contains statements to define the function to be fit to the data. The argument N indicates the function number.
Examples	<pre> FUNC 1 F=A*EXP (B) + C*EXP (D) END FUNC 1 F=A*EXP (B) + C*EXP (D) END FUNC 2 F=Z (1) END </pre>
Note	<p>A function block must be used for every data set to be fit. NFUNCTIONS specifies the number of function blocks.</p> <p>The variable F must be assigned the function value. WT can be assigned a value to use as the weight for the current observation.</p>

/

INITIAL	COMMAND
Usage	INITIAL [ARE =] I1, I2, I3, . . . , IN
Description	Specifies initial values for the estimated parameters. The arguments I1, I2, I3, . . . , IN are the initial values, separated by commas or spaces.
Examples	<pre> INITIAL = 0.01, 5.e2, -2.0 INIT are 0.02, 10, 1.3e-5 INIT 0.1 .05 100 </pre>
Note	INITIAL must be preceded by NPARAMETERS .

ITERATIONS	COMMAND
Usage	ITERATIONS [are =] N
Description	Specifies the maximum number of iterations to be executed, N.
Examples	<pre> ITERATIONS are 30 ITER 20 </pre>

Note	If ITERATIONS = 0, the usual output is produced using the initial estimates as the final values of the parameters.
------	--

J

LOWER COMMAND

Usage	LOWER [are =] L1, L2, L3, . . . , LN
Description	Specifies lower bounds for the estimated parameters. The arguments L1, L2, L3, . . . , LN are the lower bounds, separated by commas or spaces.
Example	LOWER 0 0 10
Note	If LOWER is used, a lower bound must be provided for each parameter, and UPPER must also be used. LOWER must be preceded by NPARAMETERS.

K

MEANSQUAR COMMAND
E

Usage	MEANSQUARE [is =] M
Description	Allows a specified value other than the residual mean square to be used in the estimates of the variances and standard deviations. The argument M replaces the residual sum of squares in the estimates of variances and standard deviations. If given, MEANSQUARE must be greater than 0.
Examples	MEANSQUARE = 100 MEAN=1 . 0

METHOD COMMAND

Usage	METHOD [is =] N
-------	------------------

Description	<p>In modeling, the METHOD command specifies the type of fitting algorithm WinNonlin is to use. The argument N takes the following values:</p> <ol style="list-style-type: none"> 1. Nelder-Mead Simplex 2. Levenberg and Hartley modification of Gauss-Newton (default) 3. Hartley modification of Gauss-Newton <p>In noncompartmental analysis, METHOD specifies the method to compute area under the curve. The argument N takes on the following values:</p> <ol style="list-style-type: none"> 1. Lin/Log - Linear trapezoidal rule up to Tmax, and log trapezoidal rule after 2. Linear trapezoidal rule (Linear interpolation) (default) 3. Linear up/Log down rule – Uses the linear trapezoidal rule any time the concentration data is increasing and the logarithmic trapezoidal rule any time that the concentration data is decreasing. 4. Linear Trapezoidal (Linear/log Interpolation)
Example	<pre>METHOD is 2 METH 3</pre>

MODEL COMMAND

Usage	MODEL [N[,] 'path']
Description	In a command file, MODEL specifies the compiled or source model to use. If “MODEL N” is specified, model N is used from the built-in library. If “MODEL N, PATH” is used, model N from the source file path is used.
Examples	<pre>MODEL 5 MODEL 7, 'PK' MODEL 3 'D:\MY.LIB'</pre>

MODEL LINK COMMAND

Usage	MODEL LINK M1 [,] M2
Description	<p>MODEL with the option LINK specifies that compiled models are to be used to perform PK/PD or Indirect Response modeling. The arguments M1 and M2 are model numbers where:</p> <p>M1 is the PK model number, and</p> <p>M2 is the model number of the PD model or Indirect Response model.</p> <p>When this option is used a PKVAL command must also be used.</p>
Examples	<pre>MODEL LINK 4 101 PKVAL 10 1.5 1.0 1.0 MODE LINK 1, 53 PKVA 10, 2.5</pre>
Note	See also PKVALUE .

L

NCARRY COMMAND

Usage	NCARRY [are =] N
Description	Specifies the number of carry-over variables specified. The argument N is the number of carry-over variables.
Examples	NCAR are 3 NCAR 1
Note	NCARRY must be specified before CARRYALONG .

NCATTRANS COMMAND

Usage	NCATTRANS
Description	When using NCA, NCATTRANS specifies that the Final Parameters output will present each parameter value in a separate column, rather than row. See “Transpose final parameters” in the <i>WinNonlin User's Guide</i> .
Examples	NCATTRANS NCAT

NCONSTANT COMMAND

S

Usage	NCONSTANTS [are =] N
Description	The NCONSTANTS command specifies the number of constants, N, required by the model. If used, NCONSTANTS must precede CONSTANTS .
Examples	NCONSTANTS are 3 NCON 1

NDERIVATIV COMMAND

ES

Usage	NDERIVATIVES [are =] N
Description	Indicates the number, N, of differential equations in the system being fit.
Examples	NDERIVATIVES are 2 NDER 5

NFUNCTION COMMAND

S

Usage	NFUNCTIONS [are =] N
-------	------------------------

Description	Indicates the number, N, of functions to be fit (default = 1). There must be data available for each function. If used, NFUNCTIONS must precede DATA .
Examples	NFUNCTIONS are 2 NFUN 3
Note	The functions can contain the same parameters so that different data sets can contribute to the parameter estimates. When fitting multiple functions one must delineate which observations belong to which functions. To do this, include a function variable in the data set or use NOBSERVATIONS. See FNUMBER or NOBSERVATIONS .

NOBOUNDS COMMAND

Usage	NOBOUNDS
Description	Tells WinNonlin not to compute bounds during parameter estimation.
Examples	NOBOUNDS NOBO
Note	Unless a NOBOUNDS command is supplied WinNonlin will either use bounds that the user has supplied, or compute bounds.

NOBSERVATIONS COMMAND

Usage	NOBSERVATIONS [are =] N ₁ , N ₂ , N ₃ , . . . , N _F
Description	Provides backward compatibility with PCNonlin command files. When more than one function is fit simultaneously, NOBS can indicate which observations apply to each function. The arguments N ₁ , N ₂ , . . . , N _F are the number of observations to use with each FUNCTION block. The data must appear in the order used: FUNC 1 data first, then FUNC 2, etc. Note that this statement requires counting the number of observations for each function, and updating NOBS if the number of observations changes. NOBS must precede DATA .
Example	NOBS 10, 15, 12
Note	The recommended approach is to include a function variable whose values indicate the FUNCTION block to which the data belong. (See FNUMBER .)

NOPAGE COMMAND

Usage	NOPAGE
Description	Tells WinNonlin to omit page breaks in the ASCII output file.
Examples	NOPAGE NOPAGE BREAKS ON OUTPUT

NOPLOTS

COMMAND

Usage	NOPLOTS
Description	Specifies that no plots are to be produced.
Example	NOPLOTS
Note	This command turns off both the high resolution plots and ASCII output.

NOSCALE

COMMAND

Usage	NOSCALE
Description	Turns off the automatic scaling of weights when WEIGHT is used. By default, the weights are scaled such that the sum of the weights equals the number of observations with non-zero weights.
Example	NOSCALE
Note	This command will have no effect unless WEIGHT is used.

NOSTRIP

COMMAND

Usage	NOSTRIP
Description	Turns off curve stripping for all profiles.
Example	NOSTRIP

NPAUC

COMMAND

Usage	NPAUC #
Description	Specifies the number of partial area under the curve computations requested.
Examples	NPAUC 3
Note	See PAUC .

NPARAMETERS

COMMAND

Usage	NPARAMETERS [are =] N
Description	Sets the number of parameters, N, to be estimated. NPARAMETERS must precede INITIAL , LOWER or UPPER . (LOWER and UPPER are optional.)
Examples	NPARAMETERS 5 NPAR 3
Note	See PNAMES .

NPOINTS

COMMAND

Usage	NPOINTS [is =] N
Description	Determines the number of points in the Predicted Data file. Use this option to create a data set to plot a smooth curve. $10 \leq N \leq 20,000$.
Example	NPOINTS 200
Notes	For compiled models without differential equations the default value is 1000. If NDERIVATIVES > 0 (i.e., differential equations are used) this command is ignored and the number of points is equal to the number in the data set. The default for user models is the number of data points. This value may be increased only if the model has only one independent variable.

NSECONDAR COMMAND

Y

Usage	NSECONDARY [are =] N
Description	Specifies the number, N, of secondary parameters to be estimated.
Examples	NSECONDARY 2 NSEC = 3
Note	See SNAMES .

NVARIABLES COMMAND

Usage	NVARIABLES [are =] N
Description	Specifies the number of columns in the data grid. N is the number of columns (default = 2). If used, NVARIABLES must precede DATA .
Examples	NVARIABLES 5 NVAR 3

M

P

ARRAY

Usage	P(N)
Description	Contains the values of the estimated parameters for the current iteration. The index N specifies the Nth estimated parameter.
Examples	P (1) T=P (3) - A

Note	The values of the INITIAL command correspond to P(1), P(2), etc., respectively. NPARAMETERS specifies the value of N. PNames can be used to assign aliases for P(1), P(2), etc.
------	---

PAUC COMMAND

Usage	PAUC #, # etc.
Description	Lower, upper values for the partial AUC computations. If a specified time interval does not match times on the data set, then WinNonlin will generate concentrations corresponding to those times.
Examples	PAUC 0, 5, 17, 24
Note	See NPAUC .

PKVALUE COMMAND

Usage	PKVALUE N1, N2, . . . , NPK
Description	Used to assign parameter values to the PK model when using MODEL LINK .
Examples	MODEL LINK 4 101 PKVAL 10 1.5 1.0 1.0 MODE LINK 1, 53 PKVA 10, 2.5

PNames COMMAND

Usage	PNames [are =] 'name1', 'name2', . . . , 'nameN'
Description	Specifies names associated with the estimated parameters. The arguments 'name1', 'name2', . . . , 'nameN' are names of the parameters. Each may have up to eight letters, numbers, or underscores, and must begin with a letter. These names appear in the output and may be used in the modeling code.
Examples	PNames are 'ALPHA', 'BETA', 'GAMMA' PNAM 'K01', 'K10', 'VOLUME'
Note	The number of names specified must equal that for NPARAMETERS .

PUNITS COMMAND

Usage	PUNITS [are =] 'unit1', 'unit2', . . . , 'unitN'
Description	Specifies the units associated with the estimated parameters. The arguments 'unit1', 'unit2', . . . , 'unitN' are units and may be up to eight letters, numbers, operators or underscores in length. These units are used in the output.

Examples	PUNITS are 'hr', 'ng/mL', 'L/hr' PUNI 'hr'
Note	The number of units specified must equal that for NPARAMETERS .

N**REMARK COMMAND**

Usage	REMARK [comment]
Description	Allows comments to be placed in a command file.
Examples	REMARK Y transformed to $\log(y)$ REMA GAUSS-NEWTON
Note	If a command takes no arguments or a fixed number of arguments, comments may be placed on the same line as a command after any required arguments.

REWEIGHT COMMAND

Usage	REWEIGHT W
Description	In the PK modeling module, REWEIGHT specifies that iteratively reweighted least squares estimates of the parameters are to be computed. The weights have the form $WT = FW$ where W is the argument of REWEIGHT and F is the current value of the estimated functions. If $W < 0$ and $F < 0$ then the weight is set to 0.
Examples	REWEIGHT 2 REWE - .5
Note	REWEIGHT is useful for computing maximum likelihood estimates of the estimated parameters. The following statement in a model is equivalent to a REWEIGHT command: $WT = \exp(W \log(F))$, or $WT = F^{**}W$ This command is not used with noncompartmental analysis.

O**S ARRAY**

Usage	S(N)
-------	------

Description	Contains the estimated secondary parameters. The index N specifies the Nth secondary parameter.
Example	$S(1) = -P(1)/P(2)$
Note	The range of N is set by NSECONDARY . Aliases may be assigned for S(1), S(2), etc. using SNAMES . The secondary parameters are defined in the SECONDARY section of the model.

SECONDARY MODEL BLOCK

Usage	SECONDARY Statements END
Description	The secondary parameters are defined in the SECONDARY - END block. Assignments are made to the array S.
Examples	SECONDARY S(1)=0.693/K10 END Or, if the command SNAMES 'K10_HL' and a PNames command with the alias K10 have been included: SECONDARY K10_HL=0.693/K10 END
Note	NSECONDARY must be used to set the number of secondary parameters.

SIMULATE COMMAND

Usage	SIMULATE
Description	Causes the model to be estimated using only the time values in the data set and the initial values. It must precede the DATA statement, if one is included.
Example	SIMULATE
Note	SIMULATE may be placed in any set of WinNonlin commands. When it is encountered, no data fitting is performed, only the estimated parameters and predicted values from the initial estimates are computed. All other output such as plots and secondary parameters are also produced.

SIZE COMMAND

Usage	SIZE N
Description	N is the model size. Model size sets default memory requirements and may range from 1 to 6. The default value is 2.

Example	SIZE 3
Note	Applicable only when using ASCII models.

SNAMES COMMAND

Usage	SNAMES [are =] 'name1', 'name2', . . . , 'nameN'
Description	Specifies the names associated with the secondary parameters. The arguments 'name1', 'name2', . . . , 'nameN' are names of the secondary parameters, aliases for S(1), S(2), etc. Each may be up to eight letters, numbers or underscores in length, and must begin with a letter. They appear in the output and may also be used in the programming statements.
Examples	SNAMES are 'AUC', 'Cmax', 'Tmax' SNAM 'K10_HL' 'K01_HL'

SORT COMMAND

Usage	SORT 1 [, s1 [, s2 . . .
Description	Causes WinNonlin to segment a large data file into smaller subsets. Each subset will be run through the modeling separately. Arguments s1 and the options s2, s3, etc. are the column numbers of the sort keys. The 1 that follows the command SORT has no function. It is included to provide compatibility with PCNonlin version 4.
Examples	SORT 1, 1 SORT 1 3 2 SORT 1, 1, 2
Note	SORT must be on one line, i.e., it can not use the line continuation symbol '&.'

SPARSE COMMAND

Usage	SPARSE
Description	Causes WinNonlin to use sparse data methods for noncompartmental analysis. The default is to not use sparse methods. See “Sparse sampling computations” on page 190 .

STARTING MODEL BLOCK

Usage	STARTING Statements END
-------	-------------------------------

Description	Statements in the STARTING - END block specify the starting values of the differential equations. Values are placed in the Z array. Parameters can be used as starting values.
Example	<pre>START Z (1) = CON (1) Z (2) = 0 END</pre>

**STEADY
STATE****COMMAND**

Usage	STEADY STATE #
Description	Informs WinNonlin that data were obtained at steady state, so that different PK parameters are computed. # is Tau, the (assumed equal) dosing interval.
Examples	<pre>STEADY STATE 12 STEA 12</pre>

SUNITS**COMMAND**

Usage	SUNITS [are =] 'unit1', 'unit2', . . . , 'unitN'
Description	Specifies the units associated with the secondary parameters. The arguments 'unit1', 'unit2', . . . , 'unitN' are units and may be up to eight letters, numbers, operators or underscores in length. These units are used in the output.
Examples	<pre>SUNITS are 'ml', 'mL/hr' SUNI '1/hr'</pre>
Note	The number of units specified must equal NSECONDARY .

P**TEMPORARY MODEL BLOCK**

Usage	<pre>TEMPORARY Statements END</pre>
Description	Statements placed on the TEMPORARY - END block define global temporary variables that can be used in the MODEL statements.

Example	TEMP T=X DOSE1 = CON(1) TI=CON(2) $K21 = (A * BETA + B * ALPHA) / (A + B)$ $K10 = ALPHA * BETA / K21$ $K12 = ALPHA + BETA - K21 - K10$ END
---------	---

THRESHOLD COMMAND

Usage	THRE N
Description	Sets the (optional) threshold value for use analyzing effect data with NCA model 220. See “Drug effect data (model 220)” on page 519 for details.
Example	THRE 0

TIME COMMAND

Usage	TIME
Description	Places the current time in the upper right hand corner of each output page.
Example	TIME

TITLE COMMAND

Usage	TITLE [N]								
Description	Indicates that the following line contains a title to be printed on each page of the output. Up to five titles are allowed per run. To specify any title other than the first, a value N must be included with TITLE. $1 \leq N \leq 5$.								
Examples	<table><tr><td>TITLE</td><td>TITLE 1</td></tr><tr><td>Experiment EXP-4115</td><td>Experiment EXP-4115</td></tr><tr><td></td><td>TITLE 2</td></tr><tr><td></td><td>Subject #1</td></tr></table>	TITLE	TITLE 1	Experiment EXP-4115	Experiment EXP-4115		TITLE 2		Subject #1
TITLE	TITLE 1								
Experiment EXP-4115	Experiment EXP-4115								
	TITLE 2								
	Subject #1								

TRANSFORM MODEL BLOCK

Usage	TRANSFORM Statements END
Description	Statements placed in the TRANSFORM - END block are executed for each observation before the estimation begins.

Example	<pre> TRANSFORM Y = exp(Y) END </pre>
---------	---------------------------------------

Q**UPPER COMMAND**

Usage	UPPER [are =] $U_1, U_2, U_3, \dots, U_N$
Description	Sets upper bounds for the estimated parameters. The arguments $U_1, U_2, U_3, \dots, U_N$ are the bounds, separated by commas or spaces.
Example	UPPER 1, 1, 200
Note	If UPPER is used, an upper bound must be provided for each parameter, and LOWER must also be specified. UPPER must be preceded by NPARAMETERS .

URINE COMMAND

Usage	URINE LOWER [is =] C1 UPPER [is =] C2 VOLUME [is =] C3 [is =] CONCENTRATION [is =] C4
Description	Used to specify variables when doing noncompartmental analysis for urine data (Models 210-212). The user must supply: the beginning time of the urine collection interval, the ending times of the urine collection interval, the volume of urine collected, and the concentration in the urine. The four keywords LOWER UPPER VOLUME, and CONCENTRATION identify the required data columns; C1 - C4 give their column numbers.
Examples	<pre> URINE LOWER=C1 UPPER=C2 VOLUME=C3 CONC=C4 URINE CONC C1 VOLU C2 LOWE C3 UPPE C4 </pre>
Note	The keywords may appear in any order, and only the first four letters of each keyword is required. Variables must be specified for all four keywords.

R**WEIGHT COMMAND**

Usage	WEIGHT [W]
-------	------------

Description	Activates weighted least squares computations. The argument W sets the weight value: YW. If W is not set, a column in the data set provides weights—by default, the third column. Another column may be set using WTNUMBER .
Examples	<p>WEIGHT Uses the values in Column 3 as the weights</p> <p>WEIGHT - .5 Uses as the weight 1/Sqrt(Y)</p> <p>WEIGHT</p> <p>WTNUM 7 Uses the values in Column 7 as the weights</p>
Note	The weights are scaled so that their sum = the number of observations with non-zero weights. See also WTNUMBER , WT , REWEIGHT and NOSCALE .

WT VARIABLE

Usage	WT
Description	In PK modeling, WT contains the weight for the current observation.
Example	$WT = 1 / (F * F)$
Note	If the variable WT is included in a model, then any weighting indicated via the Variables and Weighting dialog will be ignored.

WTNUMBER COMMAND

Usage	WTNUMBER [is =] N
Description	Specifies which column of the data set contains weights. Not used for NCA.
Examples	<p>WTNUMBER 4</p> <p>WTNU is 7</p>
Note	If WEIGHT is used and WTNUMBER is not, the default column for weights is 3. See also “ WEIGHT ” on page 580.

S

X VARIABLE

Usage	X
Description	Contains the current value of the independent variable.
Example	$F = \exp(-X)$
Note	By default, when used in a command file, the values of X are taken from the first column of the data set unless the XNUMBER command has been used.

XNUMBER COMMAND

Usage	XNUMBER [is =] N
Description	Sets the column number, N, of the independent variable. If differential equations are being fit, XNUMBER indicates the column number that the differential equations are integrated with respect to. The default value is 1.
Examples	XNUMBER is 3 XNUM 1
Note	XNUMBER should precede DATA .

T

Y VARIABLE

Usage	Y
Description	Current value of the dependent variable.
Example	$Y = \log(Y)$
Note	By default, Y is assumed to be in the 2 nd column unless YNUMBER is used.

YNUMBER COMMAND

Usage	YNUMBER [is =] N
Description	The YNUMBER command specifies the column number, N, of the dependent variable. When running Version 4 command files, the default value is 2.
Examples	YNUMBER is 4 YNUM 3
Note	The YNUMBER command should precede the DATA command.

U

Z ARRAY

Usage	Z(N)
Description	Contains the current values of the solutions of the differential equations. The index N specifies the Nth differential equation.
Example	$B = A * Z(2)$
Note	The maximum value of N is set by NDERIVATIVES .

Scripting Commands

WinNonlin scripting language commands, syntax and usage

For instructions on writing and executing WinNonlin scripts, refer to [“Scripting” on page 421](#). See the *Examples Guide* for example scripts.

A

AggregateVar Property

Applies To	Table
Description	For tables 9 and 10, sets the variable from Data Set 1 that composes the body of the table.
Syntax	Table.AggregateVar = <i>VariableName</i>
Remarks	Similar to a Variable in tables 4-8. The Aggregate Variable is divided into separate columns based on values of the cross variable(s). Only one Aggregate Variable may be used.
Example	Table.AggregateVar = "Conc"

AlphaTerms() Property

Applies To	Toolbox
Syntax	Toolbox.AlphaTerms(<i>n</i>)= <i>number</i>
Remarks	(<i>n</i>) corresponds to the number set by NumTerms
See also	NumTerms
Example	Toolbox.AlphaTerms(1) = .229 Toolbox.AlphaTerms(2) = 3.84

Append Method

Applies To	Data
Description	Add content of a file to another file.
Syntax	<i>Objectname.Append</i>
Remarks	<p>Data can be appended to a Pharsight workbook object (*.PWO), but it is not possible to append a .PWO to another file. When using Append:</p> <ul style="list-style-type: none"> • If the two data sets use different missing value codes, WinNonlin will not update these automatically. Use Find/Replace to standardize the missing value codes. • Formulas from Excel files will be lost. Only the values are appended. • The ReadOnly status is not recalled for the appended file. The ReadOnly status of the original file will be maintained. • All load options are in effect. When using the Data.Append method, if the file type is ASCII, it will use the .Headers, .Delimiter, etc. properties, just as the Data.Load method would.
See also	AppendFilename , AppendFileType , Headers , Delimiter , Missing
Example	<pre>MYDATA.AppendFilename = "C:\MYDIR\newdata.xls" MYDATA.AppendFileType = EXCEL MYDATA.Append</pre>

AppendFilename Property

Applies To	Data
Description	Specifies a file to be appended to an existing data object.
Syntax	<i>objectname.AppendFilename = string</i>
See also	Append , AppendFileType

AppendFileType Property

Applies To	Data
Description	Specifies the file format of the file to be appended to an existing data object.
Syntax	<i>objectname.AppendFileType = {ASCII EXCEL}</i>
Remarks	.PWO file types are not supported with this property.
See also	Append , AppendFilename

AppendFileType (continued) Property

Example	<pre>MYDATA.AppendFilename = "C:\MYDIR\newdata.xls" MYDATA.AppendFileType = EXCEL MYDATA.Append</pre>
----------------	---

AppendSheet Property

Applies To	Data
Description	Selects the sheet of a multisheet workbook to be used for an append operation (similar to the Multigrid Property). Used only with the Append method.
Syntax	MYDATA.AppendSheet="Final Parameters"
Remarks	Used with the Append method of the Data property.
Example	<pre>MYDATA1.Filename="Test1" MYDATA1.Load MYDATA1.Multigrid="Sheet 3" MYDATA1.AppendFilename="Test2" MYDATA1.AppendSheet="Sheet 1" <-selected sheet to append MYDATA1.Append</pre>

Arrange Method

Applies To	WinNonlin
Description	Arranges the icons representing any minimized windows in WinNonlin.
Syntax	WinNonlin Arrange

Aterms() Property

Applies To	Toolbox
Syntax	Toolbox.Aterms(<i>n</i>)= <i>number</i>
Remarks	The argument (<i>n</i>) corresponds to the number set by NumTerms .
Example	<pre>Toolbox.Aterms(1)=12.3 Toolbox.Aterms(2)=8.1</pre>

AutoFileOptions Property

Applies To	Data
Description	Turns on automatic file options when loading a data file.
Syntax	DataObject.AutoFileOptions = True/False

AutoSort Property

Applies To	Plot
Description	If AutoSort = "True" (default), the X variable column will be sorted before the plot is made.
Syntax	Plot.AutoSort = {True False}
Example	Plot.AutoSort = False

B**Bolus** Property

Applies To	Toolbox
Description	Sets whether WNL will constrain the estimated input rate to zero at the initial time (lag time).
Syntax	Toolbox.Bolus={True False}
Example	Toolbox.Bolus=True

C**CarryData** Property

Applies To	Merge
Description	Sets whether or not "Carry along data for like sort levels" is used. The default is True (yes).
Syntax	Merge.CarryData = {True False}

Cascade Property

Applies To	WinNonlin
-------------------	-----------

Cascade (continued) Property

Description	Causes the open objects to be cascaded in the WinNonlin window.
Syntax	WinNonlin.Cascade
See also	TileHorizontal and TileVertical
Example	WinNonlin.Cascade

CaseSensitive Property

Applies To	Data
Description	Specifies whether a search is case sensitive (True) or not (False).
Syntax	<i>Objectname</i> .CaseSensitive = {True False}
Remarks	Use with Include , Exclude , Remove , RemoveCol , RenameCol and Replace .
Example	<pre>MYDATA.SearchArea="CONC" MYDATA.Find="3" MYDATA.Operation="" MYDATA.Tolerance="0" MYDATA.CaseSensitive=False MYDATA.With="Missing" MYDATA.Replace</pre>

CloseAll Method

Applies To	WinNonlin
Description	Closes all window objects within WinNonlin.
Syntax	WinNonlin.CloseAll
Remarks	Any information that had not been saved will be lost.
See also	Save and Unload

Column Property

Applies To	Data
-------------------	------

Column (continued) Property

Description	Specifies a column to be renamed or removed, by column name (or letter). Note that WinNonlin recognizes only columns that contain data. In order to create and rename a new column, put data in the column first (e.g., using a Paste or Append operation) then rename it.
Syntax	<i>Objectname.Column = string</i>
Remarks	Use with RemoveCol and RenameCol .
Example	MYDATA.Column="a" MYDATA.With="Subject" MYDATA.RenameCol

ConcCol Property

Applies To	Toolbox
Description	Sets the input column for Concentration data.
Syntax	Toolbox.ConcCol= <i>column name</i>
Example	Toolbox.ConcCol="Concentration"

Confidence Property

Applies To	Desc
Description	Sets whether confidence intervals will be calculated with descriptive statistics (True) or not.
Syntax	Desc.Confidence = {True False}
Remarks	The default is False.
See also	Interval , Percentiles
Example	Desc.Confidence = True

ConseqDelim Property

Applies To	Data
Description	When ConseqDelim = "True," Load treats consecutive delimiters as one.
Syntax	<i>objectname</i> .ConseqDelim = {True False}
Remarks	Used only when loading ASCII data. If a value is not set, the system will use the value specified on the most-recent Load method.

ConseqDelim (continued)Property

See also	Delimiter , Missing , Headers
-----------------	---

Copy Method

Applies To	Data
Description	Copies a specified range of data and stores it for use with Paste .
Syntax	<i>Objectname.Copy = string</i>
Remarks	Optionally, specify a selection of cells for copying using StartCol , StartRow , EndCol , and EndRow . To specify an entire column (or entire columns) use only StartCol and EndCol. To specify an entire row (or entire rows) use only StartRow and EndRow.
Example	<pre>MYDATA.FileName = <file name> MYDATA.StartCol=a MYDATA.StartRow=1 MYDATA.EndCol=b MYDATA.EndRow=20 MYDATA.Copy MYDATA.StartCol=e MYDATA.StartRow=1 MYDATA.Paste</pre>

CrossVar() Property

Applies To	Table
Description	Defines cross variable(s), which define the columns when breaking a variable into separate columns within a table. CrossVars sets the number of cross variables.
Syntax	<i>Table.CrossVar(Variable no.) = Variable Name</i>
Remarks	Applies to tables 4, 5, 6, 7, 8, 9. For table 9, the cross variable must be from Data Set 1.
See also	CrossVars , GroupVar() , IDVar() , RegVar()
Example	<pre>Table.CrossVar(1) = "Subject" Table.CrossVar(2) = "Form"</pre>

CrossVars Property

Applies To	Table
-------------------	-------

CrossVars (continued)Property

Description	Defines the number of cross variables for the Table engine.
Syntax	Table.CrossVars = <i>integer</i>
Remarks	Applies to tables 4, 5, 6, 7, 8, 9.
See also	GroupVars , IDVars , RegVars
Example	Table.CrossVars = 2

Cut Method

Applies To	Data
Description	Cuts a specified range of data from a data set and stores it for use with the Paste function.
Syntax	<i>Objectname</i> .Cut = < <i>string</i> >
Remarks	To select cells to cut, use StartCol , StartRow , EndCol , and EndRow . To select an entire column (or columns) use only StartCol and EndCol; for rows, StartRow and EndRow.
Example	<pre> MYDATA.Filename = <file name> MYDATA.StartCol=A MYDATA.StartRow=1 MYDATA.EndCol=J MYDATA.EndRow=10 MYDATA.Cut MYDATA.StartRow=5 MYDATA.StartCol=A MYDATA.Paste </pre>

D**DefinitionFile** Property

Applies To	Table
Description	Loads a definition file containing all specifications for a table, except the data file(s) to use.
Syntax	Table.DefinitionFile = <i>string</i>

DefinitionFile (continued)Property

Remarks	Any input data file that includes the exact variable names used in the table definition may be used with a definition file. The definition file is machine readable only. It must be created and saved by the table Generator. It is necessary to turn a table definition file "off" if a script will use a table definition file, then create another table using Scripting Language commands. Use the command: TABLE.DefinitionFile=" " to nullify the definition file before using the commands.
Example	Table.DefinitionFile="c:\winnonln\mytable.tdf"

DeleteFiles Method

Applies To	WinNonlin
Description	Deletes a file selected using the FileMask property.
Syntax	WinNonlin.DeleteFiles
Remarks	Directory names and *.* are not allowed in file specifications.
See also	FileMask
Example	WinNonlin.FileMask = "data.xls" WinNonlin.DeleteFiles

Delimiter Property

Applies To	Data
Description	Specifies the field delimiter to use with the Load method.
Syntax	<i>objectname.Delimiter = character</i>
Remarks	Used only while loading ASCII data files. If this value is not specified, the system will use the value specified for the most-recent Load method.
See also	ConseqDelim , Missing , Headers
Example	MYDATA.Delimiter = ", " MYDATA.Delimiter = " "

Delta Property

Applies To	Toolbox
Description	User supplied smoothing parameter.

Delta (continued)Property

Syntax	Toolbox.Delta= <i>number</i>
Example	Toolbox.Delta=2

DestArea Property

Applies To	Trans
Description	If DestCol is defined as a new column name, then DestArea determines its location. The column can be either inserted adjacent to the SourceCol or appended to the end of the data set.
Syntax	Trans.DestArea = {Adjacent Append}
Remarks	If DestCol uses an existing column, then DestArea is ignored.
See also	DestCol
Example	Trans.DestArea = Append

DestCol Property

Applies To	Trans
Description	This property defines the destination column for the transform function.
Syntax	Trans.DestCol = <i>string</i>
Remarks	Spaces are not allowed in the new column name.
See also	DestArea
Example	Trans.DestCol = "LnConc"

DnErr() Property

Applies To	Plot
Description	Defines the column containing down error data for a plot with error bars.
Syntax	Plot.DnErr(<i>Data Set no.</i>) = <i>string</i>
Remarks	<i>Data Set no.</i> is required even if there is only one data source for the plot.
See also	UpErr() , ErrType
Example	Plot.DnErr(1) = "SE" Plot.DnErr(1) = "Minimum"

DoseFile Method

Applies To	Model
Description	Loads an ASCII data file containing model dosing information. See “Special file formats” on page 434 for details on this file.
Syntax	Model.DoseFile= <i>string</i>
Remarks	See “Special file formats” on page 434 .
See also	LamzFile , LinkFile , ParmFile , PartFile
Example	Model.DoseFile = "c:\dosedata.dat"

DoseUnit Property

Applies To	Toolbox
Description	Specifies unit for dosing.
Syntax	Toolbox.DoseUnit={unit}
Example	Toolbox.DoseUnit="ng/ml"

E**Ecol** Property

Applies To	Toolbox
Description	Specifies the input column for effect data.
Syntax	Toolbox.Ecol= <i>column name</i>
Remarks	Not used in calculations.
Example	Toolbox.Ecol="Effect"

EndCol Property

Applies To	Data
Description	Sets the last column in a range. Its value is a string (column name) or number (position). Note that the column name can be used to reference an existing column; but the column number (position from the left) must be used to designate a new column.

EndCol (continued) Property

Syntax	<i>Objectname.EndCol = string number</i>
Remarks	Use with RemoveRow , Cut , Copy or Font and FontSize . Set StartCol <= EndCol .
See also	StartRow , StartCol , EndRow
Examples	<pre>MyData.StartRow = 1 MyData.StartCol = "Time" MyData.EndRow = 12 MyData.EndCol = "Time" MyData.Copy MyData.StartRow = 1 MyData.StartCol = 1 MyData.EndRow = 12 MyData.EndCol = 3 MyData.Copy</pre>

EndRow Property

Applies To	Data
Description	Sets the last row in a range.
Syntax	<i>Objectname.EndRow = number</i>
Remarks	Use with the RemoveRow , Cut and Copy methods. StartRow <= EndRow . If StartRow is set and is not -1, setting EndRow = -1 selects all rows below and including StartRow . The following example removes all rows after row 4.
Example	<pre>MYDATA.StartRow=5 MYDATA.EndRow=-1 MYDATA.RemoveRow</pre>

EntireRow Property

Applies To	Data
Description	Specifies that the entire row, as opposed to individual cell(s), is to be included or excluded.
Syntax	<i>Objectname.EntireRow = {True False}</i>
Remarks	Use with Include and Exclude .

EntireRow (continued)Property

Example	<pre>MYDATA.SearchArea="entire data set" MYDATA.Find=" 0" MYDATA.Operation="<" MYDATA.Tolerance=" 0" MYDATA.EntireRow=True MYDATA.Exclude</pre>
----------------	--

ErrType Property

Applies To	Plot
Description	Determines handling of error data by the Plot engine. See “Error bars” on page 120 .
Syntax	Plot.ErrType = {Absolute Relative}
Remarks	The default value is Relative.
See also	UpErr() , DnErr()
Example	Plot.ErrType = Absolute

Exclude Method

Applies To	Data
Description	Excludes the selected data. See “Data exclusion and inclusion” on page 60 .
Syntax	<i>objectname</i> .Exclude
See also	Find , EntireRow , StartRow , EndRow , Tolerance , SearchArea , Operation , CaseSensitive
Example	<pre>MYDATA.SearchArea="entire data set" MYDATA.Find="censor" MYDATA.EntireRow=True MYDATA.CaseSensitive=False MYDATA.Exclude</pre>

ExportAll Method

Applies To	WinNonlin
Description	Exports all open objects to Microsoft Word, using the same options as PrintAll . Use the “Print” properties to set the object type(s) to export. The example below exports all charts.

ExportAll (continued) Method

Syntax	WinNonlin.ExportAll
See also	PrintAll
Example	<pre>WinNonlin.PrintData=False WinNonlin.PrintText=False WinNonlin.PrintGraph=True WinNonlin.PrintModel=False WinNonlin.ExportAll</pre>

Extra Property

Applies To	Trans																																							
Description	Contains additional information required for some transformations. Each transformation requires different information, as detailed under “Remarks” below.																																							
Syntax	Trans.Extra = { <i>string</i> <i>number</i> }																																							
Remarks	<table><tr><th>Function used</th><th>Use</th></tr><tr><td>TRANS_LN</td><td>Not used</td></tr><tr><td>TRANS_LOG10</td><td>Not used</td></tr><tr><td>TRANS_SQRT</td><td>Not used</td></tr><tr><td>TRANS_ABS</td><td>Not used</td></tr><tr><td>TRANS_CHANGEBASE</td><td>Variable Name - specifies Time column</td></tr><tr><td>TRANS_PCTCHANGE</td><td>Variable Name - specifies Time column</td></tr><tr><td>TRANS_RATIOBASE</td><td>Variable Name - specifies Time column</td></tr><tr><td>TRANS_MULTCOL</td><td>String - defines Column Y</td></tr><tr><td>TRANS_METABOLITE</td><td>String - defines the denominator, Column Y</td></tr><tr><td>TRANS_ADDCOL</td><td>String - defines Column Y</td></tr><tr><td>TRANS_SUBTRACTCOL</td><td>String - defines Column Y</td></tr><tr><td>TRANS_E</td><td>Not used</td></tr><tr><td>TRANS_INV</td><td>Not used</td></tr><tr><td>TRANS_POWER</td><td>number</td></tr><tr><td>TRANS_MULT</td><td>number</td></tr><tr><td>TRANS_DIV</td><td>number</td></tr><tr><td>TRANS_ADD</td><td>number</td></tr><tr><td>TRANS_SUBTRACT</td><td>number</td></tr></table>	Function used	Use	TRANS_LN	Not used	TRANS_LOG10	Not used	TRANS_SQRT	Not used	TRANS_ABS	Not used	TRANS_CHANGEBASE	Variable Name - specifies Time column	TRANS_PCTCHANGE	Variable Name - specifies Time column	TRANS_RATIOBASE	Variable Name - specifies Time column	TRANS_MULTCOL	String - defines Column Y	TRANS_METABOLITE	String - defines the denominator, Column Y	TRANS_ADDCOL	String - defines Column Y	TRANS_SUBTRACTCOL	String - defines Column Y	TRANS_E	Not used	TRANS_INV	Not used	TRANS_POWER	number	TRANS_MULT	number	TRANS_DIV	number	TRANS_ADD	number	TRANS_SUBTRACT	number	
Function used	Use																																							
TRANS_LN	Not used																																							
TRANS_LOG10	Not used																																							
TRANS_SQRT	Not used																																							
TRANS_ABS	Not used																																							
TRANS_CHANGEBASE	Variable Name - specifies Time column																																							
TRANS_PCTCHANGE	Variable Name - specifies Time column																																							
TRANS_RATIOBASE	Variable Name - specifies Time column																																							
TRANS_MULTCOL	String - defines Column Y																																							
TRANS_METABOLITE	String - defines the denominator, Column Y																																							
TRANS_ADDCOL	String - defines Column Y																																							
TRANS_SUBTRACTCOL	String - defines Column Y																																							
TRANS_E	Not used																																							
TRANS_INV	Not used																																							
TRANS_POWER	number																																							
TRANS_MULT	number																																							
TRANS_DIV	number																																							
TRANS_ADD	number																																							
TRANS_SUBTRACT	number																																							
See also	InData , SourceCol , Extra , DestCol , DestArea																																							
Example	TRANS_CHANGEBASE: Trans.Extra = “Hour” TRANS_POWER: Trans.Extra = 2																																							

F

FileMask Property

Applies To	WinNonlin
Description	Specifies a file in the current working folder.
Syntax	WinNonlin.FileMask = <i>string</i>
Remarks	Use with DeleteFiles . Do not use directory names or *.* to select files.
Example	WinNonlin.FileMask="bguide2.dat" WinNonlin.DeleteFiles

Filename Property

Applies To	WinNonlin, Data, Graph, Model, Text
Description	Sets a file name for Load and Save . For the WinNonlin object; specifies a workspace to load.
Syntax	WinNonlin.filename = <absolute or relative path to workspace file> or objectname.Filename = <absolute or relative path to data file>
Remarks	Use with FileType except when using with the WinNonlin object (assumes file type .wsp).
Example	WinNonlin.filename = "c:\winnonln\examples\profiles.wsp" WinNonlin.load or MyData.Filename = "..\data\profiles.pwo" MyData.FileType = "PWO" MyData.Load

FileType Property

Applies To	ANOVA, Data, Graph, Model, Text
Description	Sets the file format for Load and Save .
Syntax	objectname.FileType = {ASCII ANV EXCEL EXCEL95 EXCEL97 PWO BMP WMF PCO PMO RTF PTO LML SAS ODBC PKS} ^a
Remarks	When loading graph objects, the FileType must be set to PCO. When the FileType is set to anything other than PCO, Save is equivalent to export. WinNonlin can load WDO, WTO, WMO, WGO, and EXCEL file types but cannot save back to them.
See also	Filename , Load , Save

FileType (continued)Property

Example	<code>MyData.FileType = EXCEL</code>
----------------	--------------------------------------

- a. Use Excel97 to load Excel2000 files.

Find Property

Applies To	Data
Description	Specifies data to search for.
Syntax	<i>objectname.Find = string</i>
Remarks	Use with Exclude , Include , Remove and Replace .
Example	<pre>MYDATA.SearchArea="CONC" MYDATA.Find="0" MYDATA.Operation="" MYDATA.Tolerance="0.001" MYDATA.With="0" MYDATA.Replace</pre>

Font Property

Applies To	Data
Description	Sets the font for the current selection.
Syntax	<i>ObjectName.Font = string</i>
See also	FontSize , StartRow , StartCol , EndRow , EndCol
Example	<pre>MyData.StartRow=1 MyData.StartCol=1 MyData.EndRow=1 MyData.EndCol=10 MyData.Font="Arial"</pre>

FontSize Property

Applies To	Data
Description	Sets the font size for the current selection.
Syntax	<i>ObjectName.FontSize = number</i>

FontSize (continued)Property

See also	Font , StartRow , StartCol , EndRow , EndCol
Example	<pre>MyData.StartRow=1 MyData.StartCol=1 MyData.EndRow=1 MyData.EndCol=-1 MyData.FontSize=12</pre>

FootnoteFont Property

Applies To	Graph
Description	Sets the font for the chart footnote.
Syntax	<code>ObjectName.FootnoteFont=<i>string</i></code>
See also	FootnoteFontSize
Example	<code>MyGraph.FootnoteFont = "Arial"</code>

FootnoteFontSize Property

Applies To	Graph
Description	Sets the font size for the chart footnote.
Syntax	<code>ObjectName.FootnoteFontSize=<i>number</i></code>
See also	FootnoteFont
Example	<code>MyGraph.FootnoteFontSize = 12</code>

Function Property

Applies To	Transform
Description	Defines the function to be used by the transform engine.

Function (continued)Property

Syntax	Trans.Function = {TRANS_LN LN(x) TRANS_LOG10 Log10(x) TRANS_SQRT Square Root(x) TRANS_ABS Absolute Value(x) TRANS_CHANGEBASE Change from Baseline TRANS_PCTCHANGE %Change from Baseline TRANS_RATIOBASE Ratio from Baseline TRANS_MULTCOL $x*y$ TRANS_METABOLITE x/y TRANS_ADDCOL $x+y$ TRANS_SUBTRACTCOL $x-y$ TRANS_E e^x TRANS_INV $1/x$ TRANS_POWER x^n TRANS_MULT $x*n$ TRANS_DIV x/n TRANS_ADD $x+n$ TRANS_SUBTRACT} $x-n$
Remarks	The string must match exactly (not case-sensitive) one of the strings above.
See also	InData , SourceCol , Extra , DestCol , DestArea
Example	Trans.Function = TRANS_LN

G**GroupVar() Property**

Applies To	Table
Description	Defines the group variables for the Table engine.
Syntax	<i>Table.GroupVar(Variable no.) = string</i>
Remarks	This property must be specified after GroupVars , which sets the number of group variables. GroupVar applies to all tables except 9 and 10 (see JoinGroupVar(,)).
See also	GroupVars , GroupVarBreak() , CrossVar() , IDVar() , RegVar() See GroupVar(,) when using the Plot engine.
Example	Table.GroupVar(1) = "Subject"

GroupVar(,) Property

Applies To	Plot
Description	Defines the group variables for each data set for the Plot engine.
Syntax	<i>Plot.GroupVar(Data set no., Variable no.) = string</i>
Remarks	This property must be specified after GroupVars() , which sets the number of group variables. The example given below sets the first GroupVar for Data Set 1 to Subject.
See also	CrossVar() , IDVar() , RegVar()
Example	<code>Plot.GroupVar(1,1) = "Subject"</code> <code>Plot.GroupVar(2,1) = "Patno"</code>

GroupVars Property

Applies To	Table
Description	Sets the number of group variables for the Table engine.
Syntax	<i>Table.GroupVars = integer</i>
Remarks	Applies to all tables except tables 9 and 10. When using table 9 or 10, see JoinGroupVars .
See also	GroupVar() , GroupVar() , CrossVars , IDVars , RegVars
Example	<code>Table.GroupVars = 1</code>

GroupVars() Property

Applies To	Plot
Description	Sets the number of group variables from a given data set, for the Plot engine.
Syntax	<i>Plot.GroupVars(Data set no.) = integer</i>
Example	<code>Plot.GroupVars(1) = 2</code>

GroupVarBreak() Property

Applies To	Table
Description	Specifies whether text output has a page break when the group variable values change.
Syntax	<i>Table.GroupVarBreak (Variable no.) = {True False}</i>

GroupVarBreak() (continued) Property

See also	GroupVar() , GroupVars
Example	<code>Table.GroupVarBreak(1) = True</code>

H**Halt** Method

Applies To	WinNonlin
Description	During runtime, Halt stops the script and presents a dialog asking “Do You Want to Continue?” with Yes/No buttons. Selecting No stops the run, leaving WNL and all windows open.
Syntax	<code>WinNonlin.Halt= string</code>
Remarks	Useful in debugging scripts. Similar to MsgBox .
Example	<code>WinNonlin.Halt="See the output"</code>
See also	MsgBox

Headers Property

Applies To	Data
Description	When Headers = “True,” the first row of a data file contains column names rather than data.
Syntax	<code>objectname.Headers = {True False}</code>
Remarks	Not used when loading PWO files. The default value is that used in the last Load operation.
See also	Missing , Delimiter , ConseqDelim

I**IDVar()** Property

Applies To	Table
Description	Defines the ID variables for a table. The number of ID variables is set by IDVars .
Syntax	<code>Table.IDVar(Variable no.) = string</code>
Remarks	Applies to table no. 2, 3, 5, 6, 7, 8, 10. When using table 9, see JoinIDVar(,) .
See also	CrossVar() , GroupVar() , RegVar()

IDVar() (continued) Property

Example	<pre>Table.IDVar(1) = "Subject" Table.IDVar(2) = "Time"</pre>
----------------	---

IDVars Property

Applies To	Table
Description	Sets the number of ID variables for the Table engine. IDVar() defines each variable.
Syntax	<code>Table.IDVars = integer</code>
Remarks	Applies to table no. 2, 3, 5, 6, 7, 8, 10. When using table 9, see JoinIDVars .
See also	CrossVars , GroupVars , RegVars
Example	<code>Table.IDVars = 2</code>

Include Method

Applies To	Data
Description	Includes a range of data.
Syntax	<code>objectname.Include</code>
See also	Exclude , Find , EntireRow , Tolerance , SearchArea , Operation , CaseSensitive
Example	<pre>MYDATA.SearchArea="conc" MYDATA.Find="4.3" MYDATA.Include</pre>

IncludeVar(,) Property

Applies To	Merge
Description	Tells the Merge engine which columns to include from each data set. Set IncludeVars() first.
Syntax	<code>Merge.IncludeVar(Data Set no., Variable no.) = string</code>
See also	IncludeVars()
Example	<pre>Merge.IncludeVar(1,1) = "AUC_PO" Merge.IncludeVar(1,2) = "Cmax_PO" Merge.IncludeVar(2,1) = "AUC_IV" Merge.IncludeVar(2,2) = "Cmax_IV"</pre>

IncludeVars() Property

Applies To	Merge
Description	Defines the number of variables to be included from each data set in a merge operation.
Syntax	<code>Merge.IncludeVars(Data Set no.) = integer</code>
See also	IncludeVar(,)
Example	<pre>Merge.IncludeVars(1) = 3 Merge.IncludeVars(2) = 2</pre>

InData Property

Applies To	Desc, Trans
Description	Defines the data source for the above engines. Must be set after NumData .
Syntax	<code>engine.InData = string</code>
Remarks	The value supplied must be a valid Data object in the script file.
See also	NumData , InData()
Example	<code>Trans.InData = MYDATA</code>

InData() Property

Applies To	Table, Merge, Plot
Description	Defines the data source for the above engines.
Syntax	<code>engine.InData() = string</code>
Remarks	The value supplied must be a valid Data object in the script file.
See also	NumData , InData
Example	<pre>Merge.InData(1) = MyOral Merge.InData(2) = MyIV where MyOral and MyIV are previously defined data objects</pre>

InitialDose Property

Applies To	Toolbox
-------------------	---------

InitialDose (continued) Property

Description	Sets a value for initial dose.
Syntax	<code>Toolbox.InitialDose=<i>number</i></code>
Example	<code>Toolbox.InitialDose=11</code>

InputLag Property

Applies To	Toolbox
Description	Selects whether WinNonlin will constrain the derivative of the estimated input rate to zero at the initial time (lag time).
Syntax	<code>Toolbox.InputLag={True False}</code>
Example	<code>Toolbox.InputLag=False</code>

Interval Property

Applies To	Desc
Description	Specifies the value of the confidence interval for descriptive statistics. The default is 95.
Syntax	<code>Desc.Interval = <i>number</i></code>
See also	Confidence, Percentiles
Example	<code>Desc.Interval = 95</code>

J**JoinCrossVar(,)** Property

Applies To	Table
Description	Defines a common cross variable when merging two data sets using table 9. JoinCrossVars sets the number of join cross variables.
Syntax	<code>Table.JoinCrossVar(<i>Data Set no.</i>, <i>Variable no.</i>) = <i>Variable Name</i></code>
Remarks	Both arguments are required even if there is only one data source and/or join cross variable.
See also	CrossVar(,) , JoinCrossVars , JoinGroupVar(,) , JoinIDVar(,)

JoinCrossVar(,) (continued)Property

Example	<pre>Table.JoinCrossVar(1,1) = "Form" Table.JoinCrossVar(1,1) = "Subject" Table.JoinCrossVar(2,1) = "Form" Table.JoinCrossVar(2,1) = "Subject"</pre>
----------------	--

JoinCrossVars Property

Applies To	Table
Description	Defines the number of join cross variables for table 9.
Syntax	Table.JoinCrossVars = <i>integer</i>
Remarks	The two data sets must have the same number of cross variables.
See also	CrossVars , JoinCrossVar(,) , JoinIDVars , JoinGroupVars
Example	Table.JoinCrossVars = 2

JoinGroupVar(,) Property

Applies To	Table
Description	Defines a common group variable when merging two data sets with table 9. JoinGroupVars sets the number of join group variables.
Syntax	Table.JoinGroupVar(<i>Data Set no.</i> , <i>Variable no.</i>) = <i>Variable Name</i>
Remarks	Both arguments are required even if there is only one data source and/or join group variable.
See also	GroupVar() , JoinGroupVars , JoinCrossVar(,) , JoinIDVar(,)
Example	<pre>Table.JoinGroupVar(1,1) = "Form" Table.JoinGroupVar(1,1) = "Trt"</pre>

JoinGroupVars Property

Applies To	Table
Description	Defines the number of join group variables when merging two data sets with table 9.
Syntax	Table.JoinGroupVars = <i>integer</i>
Remarks	The two data sets must have the same number of group variables.

JoinGroupVars (continued) Property

See also	GroupVars , JoinGroupVar(,) , JoinCrossVars , JoinIDVars
Example	<code>Table.JoinGroupVars = 1</code>

JoinIDVar(,) Property

Applies To	Table
Description	Defines a common ID variable when merging two data sets with table 9. JoinIDVars sets the number of join ID variables.
Syntax	<code>Table.JoinIDVar(Data Set no., Variable no.) = Variable Name</code>
Remarks	Both arguments are required even if there is only one data source and/or one join ID variable.
See also	IDVar(,) , JoinIDVars , JoinGroupVar(,) , JoinCrossVar(,)
Example	<code>Table.JoinIDVar(1,1) = "Subject"</code> <code>Table.JoinIDVar(1,2) = "PatNo"</code>

JoinIDVars Property

Applies To	Table
Description	Defines the number of join ID variables from each data set for table 9.
Syntax	<code>Table.JoinIDVars = integer</code>
Remarks	The two data sets must have the same number of ID variables.
See also	IDVars , JoinIDVar(,) , JoinGroupVars , JoinCrossVars
Example	<code>Table.JoinIDVars = 1</code>

K**Keo** Property

Applies To	Toolbox
Description	Sets the value for Keo. Must be from 0 to 1.
Syntax	<code>Toolbox.Keo=number</code>
Example	<code>Toolbox.Keo=.5</code>

L

LamzFile Method

Applies To	Model
Description	Loads an ASCII data file containing Lambda_z information for NCA models.
Syntax	Model.LamzFile = <i>string</i>
Remarks	See “ Special file formats ” on page 434 for details on this file.
See also	DoseUnit , PartFile
Example	Model.LamzFile = "c:\lamzdata.dat"

Legend() Property

Applies To	Plot
Description	Defines a string to appear in the legend of an overlay plot (instead of the data set name).
Syntax	Plot.Legend(<i>Data Set no.</i>) = <i>string</i>
Remarks	Applies only when creating overlaid plots.
Example	<pre>Plot.Legend(1) = "Plasma" Plot.Legend(2) = "Urine" Plot.Legend(1) = ""</pre>

LegendFont Property

Applies To	Graph
Description	Sets the font for the chart legend.
Syntax	ObjectName.LegendFont= <i>string</i>
See also	LegendFontSize
Example	MyGraph.LegendFont = "Arial"

LegendFontSize Property

Applies To	Graph
Description	Sets the font size for the chart legend.

LegendFontSize (continued) Property

Syntax	<code>ObjectName.LegendFontSize=number</code>
See also	LegendFont
Example	<code>MyGraph.LegendFontSize = 12</code>

LinkFile Method

Applies To	Model
Description	Load an ASCII data file containing model parameters information for PK/PD and IPR models.
Syntax	<code>Model.LinkFile = string</code>
Remarks	When using compiled, ASCII PK, or User-ASCII models, ParmFile reads in the model parameters and possibly lower/upper bounds. When using the pre-compiled PK/PD or PK/IPR models, LinkFile reads in the PK model parameters and ParmFile reads the PD or IPR parameters, and possibly lower/upper bounds. This method should not be used for NCA models. See “ Special file formats ” on page 434 for details of these ASCII files.
See also	DoseUnit , ParmFile
Example	<code>Model.LinkFile = "c:\linkdata.dat"</code>

Load Method

Applies To	Data, Graph, Model, Text, WinNonlin
Description	Loads a file into the system using the current file name.
Syntax	<code>objectname.Load</code>
Remarks	When using graph objects, the Load method works only for file type PCO. When used with the WinNonlin object, this method loads workspaces. When used with the Data Object associated with a PKS study, loading the Data Object as file type PKS with the file name pointing to a PKS file will load the study with or without a scenario. The scenario name is captured from the PKS file pointed to by the Data Object Filename property.
See also	Filename , FileType , Save
Example	<code>MyGraph.Load</code>

LoadTemplate Method

Applies To	Graph
-------------------	-------

LoadTemplate (continued)Method

Description	Loads a template on the current graph.
Syntax	<i>objectname</i> .LoadTemplate = <i>string</i>
See also	SaveTemplate
Example	MyGraph.LoadTemplate "c:\winnonln\temps\xy.gtp"

M**MaintenanceDose** Property

Applies To	Toolbox
Description	Sets a value for maintenance dose.
Syntax	Toolbox.MaintenanceDose= <i>number</i>
Example	Toolbox.MaintenanceDose=11

Missing Property

Applies To	Data
Description	Sets the alphanumeric string that represents missing data.
Syntax	<i>objectname</i> .Missing = <i>string</i>
See also	ConseqDelim , Delimiter , Headers
Example	MYDATA.Missing = .

MoveFirst MovePrevious MoveNext MoveLast Methods

Applies To	Graph
Description	In a group of graphs created with a sort variable, selects which graph is active or displayed.
Syntax	<i>objectname</i> .MoveFirst <i>objectname</i> .MovePrevious <i>objectname</i> .MoveNext <i>objectname</i> .MoveLast
See also	SortLevel
Example	MyGraph.MoveFirst

MsgBox Method

Applies To	WinNonlin
Description	Presents a message box during script runtime. Similar to Halt .
Syntax	WinNonlin.MsgBox = <i>string</i>
Remarks	The script will pause until the user clicks OK to clear the message box from the screen.
Example	WinNonlin.MsgBox = "Script complete"

MultiGraph Method

Applies To	Graph
Description	Changes the visible graph type in multigraphs.
Syntax	<i>objectname</i> .MultiGraph = <i>Graph Name</i> where available <i>Graph Names</i> are: <ul style="list-style-type: none"> Modeling: X vs. Observed Y and Predicted Y, Observed Y vs. Weighted Predicted Y, Weighted Predicted Y vs. Weighted Residual Y, X vs. Weighted Residual Y, Partial Derivatives NCA: X vs. Observed Y Predicted Y
Example	MyGraph.MultiGraph = "X vs. Observed Y and Predicted Y"

MultiGrid Method

Applies To	Data
Description	Selects which sub-grid (worksheet) in a multigrid (workbook) is active, or displayed.
Syntax	<i>objectname</i> .MultiGrid = <i>Grid Name</i> where <i>Grid Name</i> is the worksheet name, e.g., Final Parameters
Remarks	When sorting data, use this command to select the appropriate worksheet before specifying the SortVars, SortVar(), and Sort commands. If Multigrid is used on a data set that is currently showing the history, the data will be toggled into view for that object.

N**Npoints** Property

Applies To	Toolbox
-------------------	---------

Npoints (continued) Property

Description	Number of output data points. Value must be 2 or greater.
Syntax	<code>Toolbox.Npoints=<i>number</i></code>
Example	<code>Toolbox.Npoints=100</code>

NumData Property

Applies To	Table, Plot, Merge
Description	Indicates the number of data sets to be used with an engine.
Syntax	<code>engine.NumData = <i>integer</i></code>
Remarks	When used with Merge, NumData must be 2; with table or Plot, NumData must be 1 or 2.
See also	InData()
Example	<code>Table.NumData = 1</code>

NumTerms Property

Applies To	Toolbox
Description	Number of exponential terms in the unit impulse response.
Syntax	<code>Toolbox.NumTerms=<i>number</i></code>
Remarks	Range is 1 to 9
Example	<code>Toolbox.NumTerms=4</code>

O**Operation Property**

Applies To	Data
Description	Specifies search operator.
Syntax	<code>Objectname.Operation = {<> < > <= >= =}</code>
Remarks	Use with Exclude , Include , Remove , and Replace .

Operation (continued)Property

Example	<pre>MYDATA.SearchArea="CONC" MYDATA.Find="3" MYDATA.Operation="" MYDATA.Tolerance="0" MYDATA.CaseSensitive=False MYDATA.With="Missing" MYDATA.Replace</pre>
----------------	--

OutData Property

Applies To	ANOVA, Desc, PK, table, Merge
Description	Sets the name of the output data object generated by the Run method of any engine.
Syntax	<i>engine</i> .OutData = <i>string</i>
Remarks	<i>engine</i> .OutData = "" specifies a Null alias; no output will be generated.
Example	PK.OutData = "MYDATA"

OutGraph Property

Applies To	Plot, PK
Description	Creates a graph object during the Run of an engine.
Syntax	<i>engine</i> .OutGraph = <i>string</i>
Remarks	<i>engine</i> .OutGraph = "" specifies a Null alias; no output will be generated.
Example	Plot.OutGraph = "MyGraph"

OutText Property

Applies To	ANOVA, LinMix, Bioeq, PK, Desc Stats (Box and Whiskers plot)
Description	Sets the name of the text object, if any, created when Run is invoked.
Syntax	<i>engine</i> .OutText = <i>string</i>
Remarks	<i>engine</i> .OutText= "" specifies a Null alias; no output will be generated.
Example	PK.OutText = "MYTEXT"

PageFooter Property

Applies To	Table
Description	Sets a footer for a scripted table. Accepts a quote delimited string up to 255 characters.
Syntax	<i>Table.PageFooter = string</i>
Remarks	A footer specified using this command will override any footers specified in a table definition file, if both are used in a single run.
See also	PageHeader , DefinitionFile
Example	Table.PageFooter="This is my Footnote"

PageHeader Property

Applies To	Table
Description	Sets a title for a scripted table. Accepts a quote delimited string up to 255 characters.
Syntax	<i>Table.PageHeader = string</i>
Remarks	A title specified using this command will override any titles specified in a table definition file, if both are used in a single run.
See also	PageFooter , DefinitionFile
Example	Table.PageHeader="This is my Title"

ParmFile Method

Applies To	Model
Description	Loads an ASCII data file containing the model parameter and boundaries information.
Syntax	Model.ParmFile = <i>string</i>
Remarks	When using compiled or ASCII PK models or User-ASCII models, ParmFile reads in the initial values for model parameters, and, optionally, lower/upper bounds. However, when using SCI PK/PD or PK/IPR models, LinkFile reads in the PK model parameters and ParmFile reads the PD or IPR parameters, and optional lower/upper bounds. This method should not be used for NCA models. See " Special file formats " on page 434 for details on these files.
See also	DoseUnit , LinkFile
Example	Model.ParmFile = "c:\parmdata.dat"

PartFile Method

Applies To	Model
Description	Loads an ASCII file containing time ranges for the computation of partial areas in NCA.
Syntax	Model.PartFile = <i>string</i>
Remarks	See “Special file formats” on page 434 for the required format of these ASCII files.
See also	DoseUnit , LamzFile
Example	Model.PartFile = "c:\partdata.dat"

Paste Method

Applies To	Data
Description	Pastes a range of data selected with the Cut or Copy property to a data set.
Syntax	<i>Objectname</i> .Paste
Remarks	Use with the Cut or Copy method.
Example	<pre>MYDATA.Filename = <file name> MYDATA.StartCol=A MYDATA.StartRow=1 MYDATA.EndCol=J MYDATA.EndRow=10 MYDATA.Cut MYDATA.StartRow=5 MYDATA.StartCol=A MYDATA.Paste</pre>
Applicable Properties	StartCol , StartRow , EndCol , EndRow

Percentiles Property

Applies To	Desc
Description	When Percentiles = "True," percentile calculations will be included with descriptive statistics.
Syntax	Desc.Percentiles = {True False}
Remarks	The default is False.
See also	Confidence , Interval

Percentiles (continued) Property

Example	<code>Desc.Percentiles = True</code>
----------------	--------------------------------------

Print Method

Applies To	Data, Graph, Model, Text
Description	Prints the specified object.
Syntax	<i>objectname</i> .Print
Remarks	If used on a multigrid data object, only the current data grid will be printed. If the history of a data object is visible, the history will be printed rather than the data.
See also	PrintAll
Example	<code>MyData.Print</code>

PrintAll Method

Applies To	WinNonlin
Description	Prints all open windows or data grids, including all worksheets of open workbooks.
Syntax	WinNonlin.PrintAll
See also	Print , PrintData , PrintGraph , PrintModel , PrintText
Example	<code>WinNonlin.PrintAll</code>

PrintData Property

Applies To	WinNonlin
Description	PrintData = False excludes Data windows from printing with PrintAll . The default is True.
Syntax	WinNonlin.PrintData = {True False}
See also	PrintAll , PrintGraph , PrintModel , PrintText
Example	<code>WinNonlin.PrintData=False</code>

PrintGraph Property

Applies To	WinNonlin
-------------------	-----------

PrintGraph (continued)Property

Description	PrintGraph = False excludes Graph windows from printing with PrintAll . The default is True.
Syntax	WinNonlin.PrintGraph = {True False}
See also	PrintAll , PrintModel , PrintText , PrintData
Example	WinNonlin.PrintGraph=False

PrintModel Property

Applies To	WinNonlin
Description	PrintModel = False excludes the Model window from PrintAll . The default is True.
Syntax	WinNonlin.PrintModel = {True False}
See also	PrintAll , PrintText , PrintData , PrintGraph
Example	WinNonlin.PrintModel=False

PrintMulti Property

Applies To	Chart
Description	If True, multiple charts are printed per page.
Syntax	Chartobject.PrintMulti = {True False}
See also	PrintMultiAcross , PrintMultiDown , PrintAll

PrintMultiAcrossProperty

Applies To	Chart
Description	Sets the number of charts printed across the page.
Syntax	Chartobject.PrintMultiAcross = <i>number</i>
See also	PrintMulti , PrintMultiDown

PrintMultiDown Property

Applies To	Chart
-------------------	-------

PrintMultiDown (continued)Property

Description	Sets the number of charts printed down the page.
Syntax	Chartobject.PrintMultiDown = <i>number</i>
See also	PrintMulti , PrintMultiAcross

PrintText Property

Applies To	WinNonlin
Description	PrintText = False excludes Text windows from printing with PrintAll . The default is True.
Syntax	WinNonlin.PrintText = {True False}
See also	PrintAll , PrintModel , PrintData , PrintGraph
Example	WinNonlin.PrintText=False

Q**R****RegVar()** Property

Applies To	Table
Description	Defines a "regular" variable that composes the body of a table. RegVars sets the number of regular variables.
Syntax	Table.RegVar(<i>Variable no.</i>) = <i>string</i>
Remarks	Applies to all table templates. For table 8 only one regular variable may be specified. For tables 9 and 10, the regular variables are the variables from Data Set 2.
See also	CrossVar() , IDVar() , GroupVar()
Example	<pre>Table.RegVar(1) = "Tmax" Table.RegVar(2) = "Cmax" Table.RegVar(3) = "AUClast"</pre>

RegVars Property

Applies To	Table
-------------------	-------

RegVars (continued)Property

Description	Defines the number of regular variables for a table.
Syntax	<code>Table.RegVars = integer</code>
Remarks	Applies to all table templates. For table 8 only one regular variable may be specified. For tables 9 and 10, the regular variables are the variables from Data Set 2.
See also	RegVar() , CrossVars , IDVars , GroupVars
Example	<code>Table.RegVars = 3</code>

Remove Method

Applies To	Data
Description	Removes rows fitting search criteria from a workbook.
Syntax	<code>objectname.Remove</code>
See also	Find , Tolerance , SearchArea , Operation , CaseSensitive
Example	<pre>MYDATA.SearchArea="TIME" MYDATA.Find="3" MYDATA.Operation="" MYDATA.Remove</pre>

RemoveCol Method

Applies To	Data
Description	Removes a specified column from a workbook. Identify the column by column name or letter.
Syntax	<code>Objectname.RemoveCol</code>
See also	Column , CaseSensitive
Example	<pre>MYDATA.Column="a" MYDATA.RemoveCol</pre>

RemoveRow Method

Applies To	Data
Description	Removes a range of rows from a workbook. Identify the first and last row in the range by row number.

RemoveRow (continued)Method

Syntax	<i>objectname.RemoveRow</i>
Example	<code>MYDATA.StartRow=1</code> <code>MYDATA.EndRow=18</code> <code>MYDATA.RemoveRow</code>
Applicable Properties	StartRow , EndRow

RenameCol Method

Applies To	Data
Description	Renames a column. Identify the column to be renamed by column name or letter.
Syntax	<i>objectname.RenameCol</i>
See also	Column , With , CaseSensitive
Example	<code>MYDATA.Column="a"</code> <code>MYDATA.With="Subject"</code> <code>MYDATA.RenameCol</code>

Replace Method

Applies To	Data
Description	Replaces data specified by the Find property with data specified by the With property.
Syntax	<i>objectname.Replace</i>
See also	Find , With , Tolerance , SearchArea , Operation , CaseSensitive
Example	<code>MYDATA.SearchArea="CONC"</code> <code>MYDATA.Find="3"</code> <code>MYDATA.Operation=""</code> <code>MYDATA.Tolerance="0"</code> <code>MYDATA.CaseSensitive=False</code> <code>MYDATA.With="Missing"</code> <code>MYDATA.Replace</code>

Response Property

Applies To	Toolbox
Description	Specifies column for response data.
Syntax	Toolbox.Response= <i>column name</i>
Example	Toolbox.Response="Response"

Run Method

Applies To	ANOVA, LinMix, Bioeq, Desc, Plot, PK, Trans, table, Merge
Description	Invokes the specified engine with its currently defined properties.
Syntax	<i>engine</i> .Run
Remarks	Be sure to name all possible output objects prior to using the Run method.
Example	Desc.Run

RunWordExport Method

Applies To	PK
Description	Runs the PK analysis and launches Word (if not already running), creates a new document, and populates it with the PK output.
Syntax	PK.RunWordExport
Remarks	Optionally, use WordExportFilename to have Word automatically save the output file. RunWordExport requires Word to be installed on the same machine as WinNonlin.
See also	Run , WordExportFilename
Example	Set MyModel = Model MyModel.Filename = "bg1.pmo" MyModel.FileType = "PMO" MyModel.Load PK.WordExportFilename = "bg1.doc" PK.RunWordExport

S

SameErrCol() Property

Applies To	Plot
Description	Determines whether the same column is to be used for both up and down error bars.
Syntax	Plot.SameErrCol(<i>Data Set no.</i>) = {True False}
Remarks	The argument is required even if there is only one data source. The default is False. If True, then the column defined by UpErr() is used for the error data.
See also	UpErr() , DnErr()
Example	Plot.SameErrCol(1) = True

Save Method

Applies To	Data, Graph, Model, Text, WinNonlin
Description	Saves a file to disk using the current file name.
Syntax	<i>objectname</i> .Save
Remarks	When working with graph objects, if the FileType is anything other than PCO, Save is equivalent to export. <i>Saving Workspaces:</i> Saving a workspace requires that all WinNonlin objects be saved first. The Model, if one exists, must be saved as a PMO. All multigrids must be saved as PWO's. <i>Saving PKS:</i> Saving a PKS workspace requires that the save operation be performed on the study workbook data object as a file of type PKS. The PKS file contains all information necessary to save the workspace.
See also	Filename , FileType , Load
Example	MyGraph.Save

SaveAll Method

Applies To	Graph
Description	Saves each graph in a multipart graph to a file. File names use SaveAllPrefix plus a number.
Syntax	<i>objectname</i> .SaveAll
Example	MYOUTGRAPH.SaveAllPrefix="graf" MYoutgraph.filetype="BMP" MYOUTGRAPH.SaveAll

SaveAllPrefix Property

Applies To	Graph
Description	Specifies file name prefix for saving multipart graphs. Use with SaveAll .
Syntax	<i>objectname.SaveAllPrefix = string</i>
Remarks	This prefix must be between 1 to 4 characters in length.
Example	rema Save all plots as separate .WMF files MYOUTGRAPH.SaveAllPrefix="plot " MYOUTGRAPH.filetype="wmf " MYOUTGRAPH.SaveAll

SaveTemplate Method

Applies To	Graph
Description	Saves the current graph settings to a file, as a graph template.
Syntax	<i>objectname.SaveTemplate = string</i>
See also	LoadTemplate
Example	MyGraph.SaveTemplate "c:\winnonln\temps\xy.gtp"

SearchArea Property

Applies To	Data
Description	Specifies where the Exclude , Include , Remove or Replace should focus.
Syntax	<i>objectname.SearchArea = {column name "entire data set"}</i>
Remarks	It is not possible to search a "Current Selection" using the Scripting Language.
Example	MYDATA.SearchArea="CONC " MYDATA.Find="3 " MYDATA.Operation="=" MYDATA.Tolerance="0 " MYDATA.CaseSensitive=False MYDATA.With="Missing" MYDATA.Replace

Sequence Property

Applies To	Toolbox
Description	Specifies input column for sequence data.
Syntax	Toolbox.Sequence = <i>column name</i>
Example	Toolbox.Sequence="Sequence"

ShowUnits Property

Applies To	Plot
Description	When ShowUnits = True, any units defined by XUnits or YUnits will appear after the respective axis title. The default value is True.
Syntax	Plot.ShowUnits={True False}
See also	XUnits , YUnits
Example	Plot.ShowUnits=False

Smoothing Property

Applies To	Toolbox
Syntax	Toolbox.Smoothing={Automatic None User}
Example	Toolbox.Smoothing="Automatic"

Sort Method

Applies To	Data
Description	Sorts a data set using the specified sortkeys.
Syntax	<i>objectname</i> .Sort
See also	SortVar() , SortVars

SortLevel Method

Applies To	Graph
-------------------	-------

SortLevel (continued)Method

Description	Allows navigation among plots containing sort keys by going directly to a specified sort level.
Syntax	<i>objectname.SortLevel = string</i>
Remarks	If there is more than one sort key, SortLevel must contain a value for each key in order, separated by a single space.
See also	MoveFirst , MovePrevious , MoveNext , MoveLast
Example	<code>MyGraph.SortLevel = "SubjA Cap"</code>

SortVar() Property

Applies To	Data, Desc
Description	Defines the sort keys to be used by the specified engine. Must be set after SortVars .
Syntax	<i>Data.SortVar(Variable no.) = string</i>
See also	SortVar(,) , SortVars() , GroupVar()
Example	<code>Data.SortVar(1) = "Form"</code> <code>Data.SortVar(2) = "Time"</code>

SortVar(,) Property

Applies To	Plot, Merge
Description	Defines the sort keys to be used. Must be set after SortVars or SortVars() .
Syntax	<i>Engine.SortVar(Data Set no., Sort Var. no.) = string</i>
Remarks	For the Plot engine, both arguments are required even if there is only one data source and/or sort key. For the Desc engine, there must be a value even if there is only one sort key.
See also	SortVars() , GroupVar()

SortVars Property

Applies To	Data, Desc, Merge
Description	Defines the number of sort keys to be used by the specified object or engine.
Syntax	<i>engine.SortVars = integer</i> or <i>objectname.SortVars = integer</i>

SortVars (continued)Property

Remarks	When working with the Merge engine, even though there are two data sources, the number of SortVars must be the same for each. Therefore SortVars is used, and not SortVars().
See also	SortVars(), SortVar(), GroupVars()
Examples	<code>Desc.SortVars = 1</code>

SortVars() Property

Applies To	Plot
Description	Defines the number of sort keys to be used by the Plot engine.
Syntax	<code>Plot.SortVars(Data Set no.) = integer</code>
Remarks	There must be a value even if there is only one data source.
See also	SortVars , SortVar() , GroupVars()
Example	<code>Plot.SortVars(1) = 2</code>

SourceCol Property

Applies To	Trans
Description	Defines the source column for transformations.
Syntax	<code>Trans.SourceCol = string</code>
See also	DestCol
Example	<code>Trans.SourceCol = "Conc"</code>

Stacked Property

Applies To	Toolbox
Description	Sets whether treatment data are in separate columns or stacked in one column in the input.
Syntax	<code>Toolbox.stacked={True False}</code>
Remarks	True=stacked; False=separate
Example	<code>Toolbox.stacked=True</code>

StartCol Property

Applies To	Data
Description	Specifies the first column in a range.
Syntax	<i>Objectname.StartCol = {string number}</i>
Remarks	Use with RemoveRow , Cut and Copy or Font and FontSize . Set StartCol <= EndCol. StartCol can be a string identifying the column name (for columns already containing data) or a number identifying the column position. The latter is required to designate a new column.
See also	StartRow , EndRow , EndCol
Examples	<pre>MyData.StartRow = 1 MyData.StartCol = "Time" MyData.EndRow = 15 MyData.EndCol = "Time" MyData.Copy MyData.StartRow = 1 MyData.StartCol = 1 MyData.EndRow = 15 MyData.EndCol = 3 MyData.Copy</pre>

StartRow Property

Applies To	Data
Description	Specifies the first row in a range for RemoveRow , Cut and Copy .
Syntax	<i>objectname.StartRow = number</i>
Remarks	Set StartRow <= EndRow. Both are required, unless StartRow = -1. If StartRow is -1 and EndRow is not set, all rows will be selected.
Example	<pre>MYDATA.StartRow=1 MYDATA.EndRow=18 MYDATA.RemoveRow</pre>

Stats Property

Applies To	Table
-------------------	-------

Stats (continued)Property

Description	Sets whether to generate descriptive statistics in the Table engine.
Syntax	Table.Stats = {True False}
Remarks	Applies to table no. 1, 3, 4, 6, 7, 8, 9, 10. Causes all of the available statistics to be printed.
Example	Table.Stats = True

Subject Property

Applies To	Toolbox
Description	Specifies input data set column for subject identifiers.
Syntax	Toolbox.Subject = <i>column name</i>
Example	Toolbox.Subject="Subject"

SummaryVar() Property

Applies To	Desc
Description	Specifies a variable to for which to calculate descriptive statistics. Must follow SummaryVars .
Syntax	Desc.SummaryVar(<i>Data Set no.</i>) = <i>string</i>
See also	SummaryVars
Example	Desc.SummaryVar(1) = "Conc"

SummaryVars Property

Applies To	Desc
Description	Sets the number of summary variables for descriptive statistics.
Syntax	Desc.SummaryVars = <i>integer</i>
See also	SummaryVar()
Example	Desc.SummaryVars = 1

T

Tab Property

Applies To	Data
Description	Activates the Data or History tab of a data object.
Syntax	<i>objectname.Tab = {Data History}</i>
Remarks	If MultiGrid is used on a data set that is currently showing the history, the .TAB value will automatically be set to Data—that is, the data will be toggled into view for that object.

TableNum Property

Applies To	Table
Description	Identifies a table template number.
Syntax	<i>Table.TableNum = integer</i>
Remarks	Tables may be defined in a script file either by using scripting language commands, including TableNum, or using a table definition file.
See also	DefinitionFile
Example	<i>Table.TableNum = 2</i>

Tau Property

Applies To	Toolbox
Description	Specifies value for Tau.
Syntax	<i>Toolbox.Tau = number</i>
Example	<i>Toolbox.Tau=1</i>

TerminalLower Property

Applies To	Toolbox
Description	Lower range for terminal phase.
Syntax	<i>Toolbox.TerminalLower = number</i>
Example	<i>Toolbox.TerminalLower=1</i>

TerminalPhase Property

Applies To	Toolbox
Description	Sets whether to include a user-defined terminal phase with Nonparametric Superposition.
Syntax	<code>Toolbox.TerminalPhase = { 1 0 }</code>
Remarks	1=True; 0=False. If True, use with TerminalLower and TerminalUpper .
Example	<code>Toolbox.TerminalPhase=1</code>

TerminalUpper Property

Applies To	Toolbox
Description	Upper range for terminal phase.
Syntax	<code>Toolbox.TerminalUpper = <i>number</i></code>
Example	<code>Toolbox.TerminalUpper=5</code>

TileHorizontal Property

Applies To	WinNonlin
Description	Arranges open objects to fill the WinNonlin window without overlapping, optimized for width.
Syntax	<code>WinNonlin.TileHorizontal</code>
See also	TileVertical and Cascade
Example	<code>WinNonlin.tilehorizontal</code>

TileVertical Property

Applies To	WinNonlin
Description	Arranges open objects to fill the WinNonlin window without overlapping, optimized for height.
Syntax	<code>WinNonlin.TileVertical</code>
See also	TileHorizontal and Cascade
Example	<code>WinNonlin.tilevertical</code>

TimeCol Property

Applies To	Toolbox
Description	Specifies input data column for Time.
Syntax	Toolbox.TimeCol = <i>column name</i>
Example	Toolbox.TimeCol="Time"

TimeRepeat Property

Applies To	Toolbox
Description	Sets repeat dosing interval to every <i>number</i> time units.
Syntax	Toolbox.TimeRepeat = <i>number</i>
Example	Toolbox.TimeRepeat=5

Title Property

Applies To	Plot
Description	Defines the string to be displayed in the title of the plot.
Syntax	Plot.Title = <i>string</i>
See also	XTitle , YTitle , Legend()
Example	Plot.Title = "My Plot Title"

TitleFont Property

Applies To	Graph
Description	Sets the font for the chart title.
Syntax	ObjectName.TitleFont = <i>string</i>
See also	TitleFontSize
Example	MyGraph.TitleFont = "Arial"

TitleFontSize Property

Applies To	Graph
Description	Sets the font size for the chart title.
Syntax	ObjectName.TitleFontSize = <i>number</i>
See also	TitleFont
Example	MyGraph.TitleFontSize=12

Tolerance Property

Applies To	Data
Description	Specifies tolerance range for numeric data searches.
Syntax	<i>objectname</i> .Tolerance = <i>number</i>
Remarks	Use with Exclude , Include , Remove and Replace .
Example	MYDATA.SearchArea="CONC" MYDATA.Find="3 " MYDATA.Operation="=" MYDATA.Tolerance="0 " MYDATA.CaseSensitive=False MYDATA.With="Missing" MYDATA.Replace

TreatmentA Property

Applies To	Toolbox
Description	Specifies input data set column to be used for first treatment.
Syntax	Toolbox.TreatmentA = <i>column name</i>
Example	Toolbox.TreatmentA = "Treatment1"

TreatmentB Property

Applies To	Toolbox
Description	Specifies input data set column to be used for second treatment.
Syntax	Toolbox.TreatmentB = <i>column name</i>

TreatmentB (continued)Property

Example	<code>Toolbox.TreatmentB="Treatment2"</code>
----------------	--

U**Uniform Property**

Applies To	Graph
Description	Specifies that plots created with a sort variable should have uniform axes across sort levels.
Syntax	<i>Objectname</i> .Uniform = {True False}
Example	<code>MYOUTGRAPH.UNIFORM = True</code>

Unit Property

Applies To	Data Object
Description	Assigns a unit to a column. Used in conjunction with Column .
Example	<code>MyData.Column = "Concentration"</code> <code>MyData.Unit = "ng/ml"</code>

Units Property

Applies To	PK engine
Description	Assigns a units file to provide preferred parameter units for non-compartmental analysis. The units file can be created in a text editor or using the Save button on the Units tab of the Model Options dialog. See also "Special file formats" on page 434.
Example	<code>Set MYMODEL=Model</code> <code>MYMODEL.Filename="MODEL.LIB"</code> <code>MYMODEL.FileType="ASCII"</code> <code>MYMODEL.Load</code> <code>MYMODEL.DoseFile="DOSES.DAT"</code> <code>MYMODEL.LamzFile="LAMBDАЗ.DAT"</code> <code>Pk.OutData=OUTDATA1</code> <code>Pk.OutText=OUTTEXT1</code> <code>Pk.Units="units.txt"</code> <code>Pk.Run</code>

Unload Method

Applies To	Plot
Description	Specifies that plots created with a sort variable should have uniform axes across levels.
Syntax	<i>Objectname</i> .Uniform = {True False}
Remarks	Unload should be applied to any object that will not be referenced any longer in a script. Using Unload on the WinNonlin object causes the application to close.
See also	Load
Example	<code>MyModel.Unload</code>

UpErr() Property

Applies To	Plot
Description	Defines the column containing up error data for a plot with error bars.
Syntax	<code>Plot.UpErr(Data Set no.) = string</code>
Remarks	The argument is required even if there is only one data source. If <code>SameErrCol() = "True,"</code> then <code>UpErr()</code> is used for both up and down error.
See also	DnErr() , SameErrCol()
Examples	<code>Plot.UpErr(1) = "Maximum"</code> <code>Plot.UpErr(1) = "SE"</code>

V**W****Weight** Property

Applies To	Desc
Description	Sets the weight column for weighted descriptive statistics.
Syntax	<code>Desc.Weight = string</code>
Remarks	If a weight variable is not specified, uniform weighting will be used.
Example	<code>Desc.Weight = "Weight"</code>

WindowState Method

Applies To	Data, Graph, Model, Text, WinNonlin
Description	Sets the window state of the specified object. Using this property on the WinNonlin object allows a script to run with the application minimized.
Syntax	<i>objectname</i> .WindowState = {Minimized Maximized Normal}
Example	WinNonlin.WindowState = Minimized

With Property

Applies To	Data
Description	Sets a replacement string for use with RenameCol and Replace .
Syntax	<i>Objectname</i> .With = <i>string</i>
Example	MYDATA.Column="a" MYDATA.With="Subject" MYDATA.RenameCol

WordExportFilename Property

Applies To	PK
Description	Sets the file name for the Word export method.
Syntax	PK.WordExportFilename = <i>string</i>
Remarks	Use only with RunWordExport . If WordExportFilename is not set, RunWordExport will create, but not save, the output document. Word must be installed on the same machine as WNL.
Example	Set MyModel = Model MyModel.Filename = "bg1.pmo" MyModel.FileType = "PMO" MyModel.Load PK.WordExportFilename = "bg1.doc" PK.RunWordExport

WorkingDir Property

Applies To	WinNonlin
-------------------	-----------

WorkingDir (continued)Property

Description	Sets the directory in which WinNonlin will load and save files.
Syntax	<code>WorkingDir = <i>string</i></code>
Remarks	This is particularly useful for making script files “portable,” as it removes the need to specify absolute paths for data files in the script—i.e., file name alone will suffice, if the files are located in the working directory.
Example	<code>WorkingDir = "C:\Newdir"</code>

X**XLabelsFont** Property

Applies To	Graph
Description	Sets the font for the X axis labels.
Syntax	<code>ObjectName.XLabelsFont = <i>string</i></code>
See also	XLabelsFontSize
Example	<code>MyGraph.XLabelsFont = "Arial"</code>

XLabelsFontSize Property

Applies To	Graph
Description	Sets the font size for the X axis labels.
Syntax	<code>ObjectName.XLabelsFontSize = <i>number</i></code>
See also	XLabelsFont
Example	<code>MyGraph.XLabelsFontSize = 12</code>

XTitle Property

Applies To	Plot
Description	Defines the string to be displayed in the x-axis title.
Syntax	<code>Plot.XTitle = <i>string</i></code>
See also	Title , YTitle , Legend()

XTitle (continued) Property

Example	<code>Plot.XTitle = "Time"</code>
----------------	-----------------------------------

XTitleFont Property

Applies To	Graph
Description	Sets the font for the X axis title.
Syntax	<code>ObjectName.XTitleFont = <i>string</i></code>
See also	XTitleFontSize
Example	<code>MyGraph.XTitleFont = "Arial"</code>

XTitleFontSize Property

Applies To	Graph
Description	Sets the font size for the X axis title.
Syntax	<code>ObjectName.XTitleFontSize = <i>number</i></code>
See also	XTitleFont
Example	<code>MyGraph.XTitleFontSize = 12</code>

XUnits Property

Applies To	Graph
Description	Defines the units to be displayed in the x-axis title (in parentheses after the title).
Syntax	<code>ObjectName.Xunits = <i>string</i></code>
See also	YUnits
Example	<code>MyGraph.XUnits = `hr`</code>

XUnits Property

Applies To	Plot
Description	Defines the units to be displayed in the x-axis title (in parentheses after the title).

XUnits (continued) Property

Syntax	Plot.XUnits= <i>string</i>
See also	ShowUnits , YUnits
Example	Plot.XUnits = "hr"

XVar() Property

Applies To	Plot
Description	Defines the X variable column for a plot.
Syntax	Plot.XVar(<i>Data Set no.</i>) = <i>string</i>
Remarks	The argument is required even if there is only one data source.
Example	Plot.XVar(1) = "Time"

Y**YLabelsFont** Property

Applies To	Graph
Description	Sets the font for the Y axis labels.
Syntax	ObjectName.YLabelsFont = <i>string</i>
See also	YLabelsFontSize
Example	MyGraph.YLabelsFont = "Arial"

YLabelsFontSize Property

Applies To	Graph
Description	Sets the font size for the Y axis labels.
Syntax	ObjectName.YLabelsFontSize = <i>number</i>
See also	YLabelsFont
Example	MyGraph.YLabelsFontSize = 12

YTitle Property

Applies To	Plot
Description	Defines the string to be displayed in the y-axis title.
Syntax	<code>Plot.YTitle = <i>string</i></code>
See also	Title , XTitle , Legend()
Example	<code>Plot.YTitle = "Concentration"</code>

YTitleFont Property

Applies To	Graph
Description	Sets the font for the Y axis title.
Syntax	<code>ObjectName.YTitleFont = <i>string</i></code>
See also	YTitleFontSize
Example	<code>MyGraph.YTitleFont = "Arial"</code>

YTitleFontSize Property

Applies To	Graph
Description	Sets the font size for the Y axis title.
Syntax	<code>ObjectName.YTitleFontSize = <i>number</i></code>
See also	YTitleFont
Example	<code>MyGraph.YTitleFontSize = 12</code>

YUnits Property

Applies To	Graph
Description	Defines the units to be displayed in the y-axis title of the plot (in parenthesis after the title).
Syntax	<code>ObjectName.YUnits = <i>string</i></code>
See also	XUnits
Example	<code>MyGraph.YUnits = "mg/ml"</code>

YUnits Property

Applies To	Plot
Description	Defines the units to be displayed in the y-axis title of the plot (in parenthesis after the title).
Syntax	Plot.YUnits = <i>string</i>
See also	ShowUnits , XUnits
Example	Plot.YUnits = "mg/ml"

YVar Property

Applies To	Plot
Description	Defines the Y variable column for a plot.
Syntax	Plot.YVar(<i>Data Set no.</i>) = <i>string</i>
Remarks	The argument is required even if there is only one data source.
Example	Plot.YVar(1) = "Concentration"

Warnings and Error Messages

Messages by category

There are six sources of error messages in WinNonlin. Each has a unique set of error message numbers, as follows:

Source	Message numbers
Microsoft Visual Basic & OCX's ^a	1-9999
Compartmental modeling	10000-10999
LinMix and Bioequivalence wizards	11000-11999
Table Wizard	12000-12999
Descriptive statistics	13000-13999
Noncompartmental analysis	14000-14999
IVIVC	15000-15999
Scripting language	19000-19999

a. See Microsoft’s documentation for information on error messages from Visual Basic and the Windows OCX's.

Compartmental modeling

See also “[Troubleshooting modeling problems](#)” on page 211.

- 10001

Invalid statement. Program terminated.
The preceding statement contains a syntax error (e.g. C = A+ *B).
- 10002

Unmatched parenthesis. Program terminated.
The preceding statement contains an extra left parenthesis.

- 10003 Improper statement found when evaluating model. Program terminated.
- This is an unusual error. If it occurs, please bring it to the attention of Pharsight customer support.
- 10004 Missing END statement. Program terminated.
- An “END” statement was missing for one or more of the following labeled groups of commands: [TRANSFORM](#), [COMMANDS](#), [TEMPORARY](#), [STARTING](#), [DIFFERENTIAL](#), [FNUMBER](#), [SECONDARY](#).
- 10005 Invalid number. Program terminated.
- The previous equation contains an illegal number (constant).
- 10006 Model too complex. Rerun with larger model size ([SIZE](#) command). Program terminated.
- The model was too complex for the default memory settings. Try the model again after increasing the model size (via the [SIZE](#) command, or the Model size option on the PK Settings tab, in the Model Options dialog box).
- 10007 Invalid variable name or character. Program terminated.
- The previous equation contains an invalid character. (i.e., not A-Z, 0-9, +, -, /, *, **, (), &)
- 10010 Model too complex. Rerun with larger model size ([SIZE](#) command). Program terminated.
- Workspace is not large enough to accommodate the model. The maximum number of variables and operators has been exceeded (2000). To help alleviate this problem, assign expressions to temporary variables (in the TEMP section) before using them to define the model.
- 10011 Unmatched parenthesis. Program terminated.
- There was an unmatched or missing right parenthesis, or too many arguments were encountered in a subscribed variable or function (e.g. [DZ](#)(1,1) or [ABS](#) (A,B)).
- 10012 Invalid function number. Program terminated.
- An invalid [FUNCTION](#) number was assigned. Function numbers must be integers, greater or equal to 1, and less than or equal to the number of functions in the [NFUNCTIONS](#) command.

-
- 10013 Invalid group identifier. Program terminated.
An invalid label was assigned to a group of commands (e.g. STRT in place of START).
- 10014 Missing argument in function or array. Program terminated.
An argument was missing for a function (e.g., MAX(A) instead of MAX(A,B))
- 10015 Model section does not exist. Check to see if NSEC > 0 but no secondary parameters are defined, or if the **FUNCTION** block(s) is missing. Program terminated.
A labeled section does not exist. For example, an NSEC command was specified, but the secondary parameters were never defined.
- 10017 Equation too complicated. Program terminated.
At least one of the equations was too complicated to evaluate. Split it up into two or more parts using temporary variables.
- 10018 No IF corresponding to ENDIF. Program terminated.
An ENDIF statement was encountered without a matching IF statement.
- 10019 Missing ENDIF statement. Program terminated.
An ENDIF statement is missing.
- 10020 Undefined (GOTO) label. Program terminated.
A label was referenced (e.g. with a GOTO) which was never defined. Either the label is missing entirely, or is missing the ending colon ":".
- 10021 Invalid (GOTO) label name. Program terminated.
The label name is invalid. Change the label to a valid name.
- 10022 Model too complex. Rerun with larger model size (SIZE command). Program terminated.
The maximum number of temporary variables and labels (100) has been exceeded.
- 10023 No matching IF.. THEN.. for an ELSE statement. Program terminated.

An ELSE statement was encountered without corresponding IF - THEN statements.

10024 IF statements nested too deeply. Program terminated.

IF statements can only be nested as deep as 10 levels.

10025 Cannot branch out of program section. Program terminated.

An attempt was made to branch from one block of statements (TEMP, FUNC, etc.) to a labeled section in another block of statements using a GOTO. This is not supported.

10026 Duplicate label name. Program terminated.

A duplicate label name was encountered. Change the label to a new name.

10027 Invalid array subscript. Check to see if subscript exceeded array bound. Program terminated.

An invalid array access was attempted. The model has gone outside allowable limits when accessing some variable which is an array (e.g. DTA (11)).

10028 Invalid statement encountered when evaluating model. Program terminated.

An invalid branching occurred during execution. Check to see that the model is specified correctly.

10029 Invalid statement encountered when evaluating model. Program terminated.

An invalid comparison occurred during execution. Check to see that the model is specified correctly.

10030 Operation attempted using A missing VALUE. Program terminated.

10031 Attempt to divide by zero. Program terminated.

10033 Invalid argument for LOGE(X). Program terminated.

The argument for loge(x) was invalid. Perhaps X was undefined, or 0 or a negative number.

10034 Invalid argument for LOG10(X). Program terminated.

The argument for log10(x) was invalid. Perhaps X was undefined, or 0 or a negative number.

-
- 10035 Invalid argument for SQRT(X). Program terminated.
The argument for SQRT(x) was invalid. Perhaps X was undefined or a negative number.
- 10036 Invalid argument for ATAN2(X,Y). Program terminated.
The argument for ATAN2(x,y) was invalid. Perhaps either X or Y was undefined or a negative number, or perhaps only one argument was given.
- 10037 Problem exceeds available resources. Program terminated.
Notify Pharsight technical support for assistance at support@pharsight.com.
- 10050 DO statement missing a NEXT statement. Program terminated.
A NEXT statement must end each DO loop. Check all DO statements for corresponding NEXT statements.
- 10051 Too many nested DO statements. Program terminated.
DO statements may only be nested 10 deep.
- 10052 Invalid DO statement. Program terminated.
Check the arguments and syntax of any DO statements.
- 10053 Maximum model size is 64. Program terminated.
Please check the model specification for problems.
- 10054 Maximum length of DO statement is 788 characters. Program terminated.
- 10055 Maximum number of DO statements is 999. Program terminated.
- 10056 NEXT statement missing corresponding DO. Program terminated.
A NEXT statement ends a DO loop. Check all NEXT statements for corresponding DO statements.
- 10057 String length exceeds 256. Program terminated.
No input record may exceed 256 characters in length.
- 10058 Invalid lag function argument. Program terminated.
The argument for lag(x) was invalid. X must be the column number of a defined variable.

- 10101 The number of data observations, NOBS, was not defined.
 The number of data observations to be input (NOBS) was not defined.
- 10102 A DATA command was not encountered.
 Missing DATA command.
- 10103 The number of parameters to be estimated, NPARM, was not given.
 No NPARAMETERS command was encountered.
- 10104 The initial estimates of the parameters, INIT, were not given.
 No INITIAL command was encountered.
- 10105 CONV = 0 is only supported for integrated functions (NDER=0).
 Turning off of convergence checks is only supported for integrated equations.
- 10106 The number of parameter values, NPARM, must be specified prior to INIT,
 LOWER, OR UPPER
 NPARAMETERS must precede INITIAL, LOWER and UPPER commands.
- 10107 Error converting units for next value. Using default units.
 The preferred units were inconsistent with the default units or were not
 acceptable units.
- 10108 The number of data observations, NOBS, must be specified prior to the DATA
 command.
 The number of observations to be input (NOBS), must precede DATA.
- 10109 Error encountered when reading initial parameter values.
 The program was unable to read NPARM numbers on the INITIAL command.
 Either fewer than NPARM initial estimates were given, or one of the numbers
 was not carefully defined (e.g. 1.2 - E-3).
- 10110 Error encountered when reading the number of NOBS for each function.
 The program was unable to read NFUN numbers on the NOBS command.
- 10111 The number of functions must be greater than 0.
 The specified number of functions must be between 1 and 10.

-
- 10112 The number of parameters to be estimated must be greater than 0. Program terminated.
The specified number of parameters must be between 1 and 10.
- 10113 The number of derivatives must be greater than 0. Program terminated.
If the system includes differential equations, the number of differential equations in the system must be between 1 and 10.
- 10114 The specified method for estimating the residual sum of squares must be 1, 2 or 3. Program defaulting to method 2, execution continuing.
The specified **METHOD** must be 1, 2 or 3. Method 1: Nelder-Mead, Method 2: Hartley's and Levenberg's modification of the Gauss-Newton Algorithm (default), Method 3: Hartley's Modification of the Gauss-Newton Algorithm.
- 10115 The maximum number of iterations must be greater than or equal to zero. Program defaulting to 50, execution continuing.
An invalid value was assigned to maximum number of **ITERATIONS**.
- 10116 The number of variables in the input file must be greater than 0.
A maximum of 10 variables (columns) can be read in from a data file.
- 10117 The column numbers corresponding to X and Y must be greater than 0 and must be less than or equal to NVAR.
The column numbers associated with X and Y (**XNUMBER**, **YNUMBER**) must be between 1 and 10.
- 10118 The number of rows in the plots must be between 10 and 50. Program defaulting to 50, execution continuing.
An invalid value was entered.
- 10119 The number of columns in the plots must be between 10 AND 100. Program defaulting to 100, execution continuing.
An invalid value was entered.
- 10120 The specified convergence criterion must be greater than or equal to 0. Program defaulting to 1.E-4, Execution continuing.
An invalid value was assigned to the **CONVERGENCE** command.

- 10121 The increment for estimating partial derivatives must be between 0 AND 0.1. Program defaulting to 0.001, execution continuing.
- The increment used for estimating the partial derivatives DINC must be between 0.0 and 0.1.
- 10122 The variable names for X and Y cannot be equal.
- X and Y cannot be assigned to the same column in the input data file.
- 10123 The variable number corresponding to weight must be greater than 0.
- If weights are to be read in from the data file, they must have an assigned column number greater than one.
- 10124 Error encountered when trying to read a number on the preceding command.
- An error was encountered when trying to read the (first) number from the preceding command. Either the program was expecting a number to be specified and none was given, or the number was not correctly defined (e.g. 1.2 - E-3).
- 10125 Error on data file
- An error was encountered when reading the data file. There are several ways this can happen: a. Fewer than NOBS observations were on the data file.
 b. Fewer than NVAR numbers were on one or more rows on the data file.
 c. One or more illegal numbers (e.g. 1.2 - E-3).
- 10126 Error encountered when reading in the upper or lower parameter limits. Execution continuing without limits on the parameter space.
- An error was encountered when reading the lower (or upper) parameter limits. Either fewer than **NPARAMETERS** numbers were on the card or one or more invalid numbers were specified (e.g. 1.2 - E-3).
- 10127 If lower (or upper) limits are specified then upper (or lower) limits must also be specified. Execution continuing without limits on the parameter space.
- If LOWER limits are specified, then UPPER limits must also be specified (and vice-versa).
- 10128 An improper call was made to the numerical integration routine. Check your input and model. Program terminated.
- An error was encountered when trying to numerically integrate a system of differential equations. This can usually be attributed to an improperly defined model. Check to see that the model was correctly specified.

- 10129 Too much relative accuracy was requested for this problem. Relative error and-or absolute error tolerances were reset. Check your model for possible singularities. Execution continuing.
- The program is having difficulty obtaining the necessary precision when integrating a system of differential equations. Check the model for possible singularities.
- 10130 The program is having to work very hard to obtain the desired accuracy. Make sure your model is correct. Execution continuing
- A very large number of function evaluations are having to be made to obtain the desired precision. Check to see that the model was correctly specified.
- 10131 The number of constants must be greater than 0.
- If **CONSTANTS** are used, the number of user input constants must be between 1 and 500.
- 10132 The number of constants must be specified before assigning them values.
- The NCON command must appear prior to the **CONSTANTS** command.
- 10133 An error was encountered when reading the values assigned to the constants.
- An error was encountered when trying to read the NCON values assigned to constants. Either fewer than NCON values were specified, or one of the values was not correctly defined (e.g. 1.2 - E-3).
- 10134 Parameter estimate for <parameterName> is at or near boundary.
- 10135 An initial parameter estimate is outside of the specified upper and lower limits. Parameter limits ignored - Execution continuing.
- One or more of the initial parameter estimates is outside the specified upper and lower limits.
- 10136 An error was encountered when reading constant names. Program terminated.
- 10137 The referenced library contains a model statement without a model number. Program terminated.
- Each **MODEL** statement in a WinNonlin library must also have a corresponding model number (e.g. **MODEL 7**).
- 10138 A convergence failure occurred during a singular value decomposition. Check the model for possible singularities. Program terminated.

Check to make sure that the model has been correctly defined.

- 10139 NCON was specified but the constants were not input.
An **NCONSTANTS** command was encountered but the constants were never defined (**CONSTANTS**).
- 10140 NSEC must be specified prior to the SNAME command.
The **NSECONDARY** command must precede the **SNAMES** command.
- 10142 NSEC must be greater than 0.
If secondary parameters are specified, the number of secondary parameters must be between 1 and 50.
- 10143 An error was encountered when reading parameter, secondary, data or unit names. Program continuing with default names.
An error was encountered reading the parameter or secondary parameter names. Make sure that the number of input names matches **NPARAMETERS** (or **NSECONDARY**) and that each name is enclosed in single quotes.
- 10144 The specified model number does not exist in the specified library.
- 10145 The specified title number must be 1, 2, 3, 4, OR 5.
The specified **TITLE** number must be 1, 2, 3, 4, or 5 (e.g. TITLE 3).
- 10146 The specified number of plot points must be between 10 and 20000.
NPOINTS reset to default.
The specified number of points to be included in the output plot file must be between 10 and 20000.
- 10147 Scale values could not be computed for the next plot. Plot deleted.
Axis scale values could not be computed for a plot. This could happen, for example, if all values to be plotted were identical.
- 10148 The number of constants (NCON) is inconsistent with the number of administered doses.
The number of constants (**NCONSTANTS**) is not consistent with the number of doses given.

-
- 10149 The WEIGHT command cannot be used when performing simulations. The WEIGHT command was changed to REWEIGHT.
- When performing simulations, since no observed data exist, it is inappropriate to use the **WEIGHT** command. **REWEIGHT** will be used.
- 10150 The parameter variance-covariance matrix is singular. Program terminated.
- The variance-covariance matrix for the model parameters is singular.
- 10151 The algorithm is unable to converge. Program terminated.
- The Gauss-Newton algorithm was unable to converge. Try using different initial estimates, or try rerunning the problem using the Nelder-Mead algorithm.
- 10152 The residual sum of squares is zero. Convergence assumed.
- 10153 One or more weights became negative. Weight set to missing.
- Check model definition statements to make sure **WT** is defined correctly.
- 10154 The specified library file does not exist. Program terminated.
- Check for proper specification of the model file.
- 10155 An error was encountered when reading beta time range. Program continuing with default estimation of time range.
- These are the values for selecting the Lambda Z time interval.
- 10156 An error was encountered when reading the missing value code. Program continuing with default missing value code.
- Make sure that the missing value code is eight characters or less in length, and surrounded by quotes.
- 10157 The value of FNUM is invalid. Check to make sure NVAR is correct. Program terminated.
- FNUMBER** must be the number of a column on the input data set.
- 10161 Inadequate system resources available for this problem.
- Notify Pharsight technical support for assistance at support@pharsight.com.
- 10162 Error encountered when reading link model PK constant values.
- An error occurred when reading the parameter values for the PK model.

- 10163 NOBS command ignored since FNUM specified.
The function variable specified by **FNUMBER** will be used to determine which observations belong to which functions.
- 10164 The NOBS values are inconsistent with the number of records in the data file.
The sum of the **NOBSERVATIONS** values is not equal to the number of observations on the data set. It is recommended to use a function variable for data with more than one function. See “**FNUMBER**” on page 566.
- 10165 The PNAME, SNAME, DNAME, CNAME, PUNIT and SUNIT values cannot contain embedded blanks.
- 10166 Internal parsing file error. Please report to technical support.
Contact Pharsight Technical Support for assistance: support@pharsight.com.
- 10167 Command encountered when reading model statements.
Check whether the model is missing an EOM statement or any commands in the model are not inside a block. See also “**Model structure**” on page 222.
- 10170 Improper use of END statement. Check to see if a compiled model was specified by user. Model statements were contained in the command file.
Use END statements only in user models. See “**User Models**” on page 219.
- 10171 The lower and upper bounds were equal for one or more model parameters. Please modify them and rerun your model.
- 10172 The number of observations must be greater than the number of parameters.
- 10173 Error degrees of freedom less than or equal to zero. Error variance and standard errors or parameters cannot be estimated.
- 10174 Infusion model should not be used if end time is less than or equal to start time. Use bolus model if start time equals end time.
- 10175 Models 15, 16, and 17 do not support bolus dose \leq zero, IV dose \leq zero, or infusion length \leq zero.
- 10201 Initial estimates cannot be determined for this model.
Initial estimates cannot be found for this model. If no bounds were used, try adding bounds so that the program can grid search for initial estimates.

- 10202 Curve stripping is only done for single dose data.
 Automatic estimation of initial estimates is available for single dose data only.
- 10203 An error occurred during curve stripping.
 Initial estimates could not be found for the model. If no bounds were used, try adding bounds so that the program can grid search for initial estimates.
- 10204 The requested number of exponentials could not be fit. Please try fitting a simpler model.
 Initial estimates for the number of exponential terms in the specified model could not be determined. Try fitting a less complex model, less compartments.
- 10205 Due to failure of the curve stripping method, a grid search will be performed.
 Initial estimates could not be determined for this model via curve stripping. A grid search will be used to obtain initial estimates.
- 10206 The initial estimates are outside of the lower or upper parameter limits. Execution continuing without limits on parameter space.
 The initial parameter estimates are outside of the **LOWER** and **UPPER** limits.
- 10998 Execution error. Contact Pharsight technical support at support@pharsight.com. Note that this may occur when extracting PWO's directly from zipped files. The files should be extracted first.

LinMix and Bioequivalence wizards

When errors occur using the LinMix or Bioequivalence wizards a message box will appear with the error number and a description of the error.

- 11001 Memory allocation error.
 Insufficient memory. If other applications are running, save your work in these applications, close them and try again. Closing other WinNonlin windows would also help.
- 11002-11032 Internal error during calculations. Contact Pharsight Technical Support.
- 11050, 11051 Internal parsing error. Contact Pharsight Technical Support.
- 11060 Corrupted data set.

11061	Input data not rectangular.
11062	Too many levels of a classification variable.
11063	Character variables in models must be class variables.
11064	Weight variable must be numeric.
11065	The residual variance is not modeled. Try random instead of repeated.
11066	There are no degrees of freedom for residual. Try the Satterthwaite option.
11067	Dependent variable must be numeric.
11068	There are no residual degrees of freedom. No statistical tests are possible.
11069	There is no residual variance. Try a different model.
11070	Error in Satterthwaite DF. Try using Residual DF option.
11071	Error in Variance Matrix. Model may be over-specified.
11075	Error opening file.
11089	Bioequivalence statistics could not be computed for some test formulations.
11090	Asymptotic covariance matrix not computed. Information matrix is deemed singular.
11091	Newton's algorithm converged with modified Hessian. Output is suspect.
11092	Failed to converge in allocated number of iterations. Output is suspect.
11093	Starting estimates supplied by user are invalid. Using method of moments.
11094	Negative final variance component. Consider omitting this VC structure. If this occurs when using the default bioequivalence model with Subj(Seq) as a random effect, it most likely indicates that the within-subject variance (residual) is greater than the between-subject variance. A more appropriate model would be to move Subj(Seq) out of the random model and into the fixed model, i.e., Seq+Subj(Seq)+Formulation+Period.
11095	Negative final variance parameter. Consider FA0 structure instead of UN.

11096	Negative final block term for CS. Consider omitting random model for this structure.
11097	Negative final variance parameter for TOEP structure.
11098	Negative final variance parameter for AR structure.
11099	Negative final standard deviation for ARH/CSH structure.
11101	More than 2 treatment formulations not allowed for population/individual bioequivalence.
11102	More than 10 periods not allowed for population/individual bioequivalence.
11103	More than 200 subjects not allowed for population/individual bioequivalence.
11104	More than 10 sequences not allowed for population/individual bioequivalence.
11105	Fewer than two treatment formulations. Program terminating.
11106	Fewer than two periods. Program terminating.
11107	Fewer than 2 sequences had complete design. Program terminating.
11108	Missing data for dependent variable causing an incomplete design.
11109	Ln-transform requested for data that is zero or negative causing an incomplete design.
11121	Subject <name> had incomplete design and was discarded.
11122	Sequence < > had less than 2 complete subjects and was discarded.
11201	<p>The column ColName contains alphanumeric value(s) that are greater than maxsize in length. We suggest creating a new variable with re-coded values that are less than or equal to this maximum length. Do you want to continue with shortened values?</p> <p>Alpha variables are limited to a length of 128 characters.</p>
11206	<p>There is an error on the following tab(s): <i>tab name(s)</i>.</p> <p>This error occurs when there is an error on one of the tabs. Possibly the model is missing, or the dependent variable is missing.</p>

11207	Failed to create model as specified in the model file.
11208	Unable to find any DATA command in command filename. This error occurs when the user loads a model command file from the tool bar and this file does not contain any valid DATA statement.
11210	Unable to find file command filename This error occurs when the specified command file does not exist.
11211	File data filename not found This error occurs when the data file referenced by the data statement in the command file is not found.
11212	Second delimiter in filename for DATA statement not found.
11214	Variable name too long for LinMix, variable will be temporarily truncated for use by LinMix. Maximum variable name length is 256 characters.
11215	There are greater than 256 variables in your data set. The LinMix module has a limit of using no more than 256 variables. Please decrease the number of variables on the data set then rerun the LinMix module.

Table Wizard

When errors occur with the Table Wizard a message box will appear with the error number and a description of the error.

12999	Memory allocation error Insufficient memory. If other applications are running, save your work in these applications, close them and try again. Closing other WinNonlin windows would also help.
-------	---

Descriptive statistics

When errors occur in the Descriptive Statistics module a message box will appear with the error number and a description of the error.

13000	Floating Point error
13999	Memory allocation error

Insufficient memory. If other applications are running, save your work in these applications, close them and try again. Closing other WinNonlin windows would also help.

Noncompartmental analysis

Error messages

- | | |
|-------|--|
| 14001 | Memory allocation error. Insufficient random access memory. If other applications are running, save work, close the applications and try again. Closing other WinNonlin windows may also help. |
| 14002 | Error opening file. Contact Pharsight Technical Support. |
| 14021 | All concentration values are zero. |
| 14022 | Only 1 non-zero concentration value found, and it was at dosing time. |
| 14024 | All volume values are zero. No grid output for this subject. |
| 14050 | Internal parsing error. Contact Pharsight Technical Support. |
| 14061 | No data in steady state dosing interval; program terminating. |
| 14062 | Duplicate time values on data file. Use Descriptive Statistics to compute mean data and perform NCA on the means, or select Sparse Sampling in NCA. |
| 14064 | Invalid urine data: lower time \geq upper time, or negative volume or concentration. |
| 14066 | Data error: same subject cannot have multiple samples at same time. |

Warnings

- | | |
|-------|--|
| 14501 | Incompatible units for partial AUCs and summary AUCs; using default units. |
| 14502 | Incompatible units for summary table AUMC values; using default units. |
| 14503 | Incompatible units for summary table Amount values; using default units. |

14504	Incompatible units for final parameter <default parameter name>; using default units.
14511	MRT parameters are adjusted for length of infusion.
14512	Adjusting for infusion has caused negative MRTlast value.
14513	Adjusting for infusion has caused negative MRTinf(obs) value.
14514	Adjusting for infusion has caused negative MRTinf(pred) value.
14515	Therapeutic window calculations cannot be done for negative data.
14520	Only one non-missing observation. All final parameters are set to missing.
14521	All concentration values are zero.
14522	Only 1 non-zero concentration value found, and it was at dosing time.
14524	All volume values are zero. All final parameters are set to missing.
14523	Less than two non-missing observations. No grid output for this subject.
14530	Lambda_z could not be estimated. No parameters could be extrapolated to infinity.
14531	AUC_TAU could not be estimated. Steady state PK parameters could not be estimated.
14532	The value at dosing time could not be determined by log-linear regression of the first two measurements, and was set to the first observed measurement. You may also try using a constant infusion model.
14533	No response at dosing time. Baseline is set to 0.
14534	Threshold cannot equal baseline; threshold will not be used.

IVIVC

15001	Memory allocation error. Insufficient random access memory. If other applications are running, save work, close the applications and try again. Closing other WinNonlin windows may also help.
-------	--

15002	Error opening file. Contact Pharsight Technical Support.
15003	Internal parsing error. Contact Pharsight Technical Support.
15004	Duplicate time values on data file. Use Descriptive Statistics to compute mean data to perform analysis on the means.
15010	LambdaZ could not be estimated by WNL. Cannot produce Wagner-Nelson table. (only for Wagner-Nelson)
15011	LambdaZ could not be estimated by WNL. Cannot continue with Loo-Riegelman. Try specifying AUCINF value. (Loo-Riegelman only.)
15101	Memory allocation error. Insufficient random access memory. If other applications are running, save work, close the applications and try again. Closing other WinNonlin windows may also help.
15102	Error opening one or more input files. Contact Pharsight Technical Support.
15103	Number of non-missing observations for a function is zero.
15104	Cannot have equal time values in input data set.
15105	Internal convolution error. Contact Pharsight Technical Support.

Scripting language

Errors that are generated when using the Scripting Language are written to the data log file. Please consult this log for guidance in resolving problems with script files. If no errors exist, this log file will not be created.

19999	Memory allocation error Insufficient memory. If other applications are running, save your work in these applications, close them and try again. Closing other WinNonlin windows would also help.
-------	---

General

20000 and above	Internal error. Contact Pharsight technical support at support@pharsight.com .
-----------------	---

References

Further reading on modeling and related analyses

See the following headings for an alphabetized listing of related references.

- “Bioequivalence” on page 661
- “In vitro-in vivo correlation and formulation science” on page 662
- “Pharmacokinetics” on page 663
- “Regression and modeling” on page 664
- “Sparse data analysis methods” on page 666
- “Statistics” on page 667

Bioequivalence

Anderson and Hauck (1983). A new procedure for testing equivalence in comparative bio-availability and other clinical trials. *Commun Stat Theory Methods*, 12:2663-92.

Chow and Liu (2000). *Design and Analysis of Bioavailability and Bioequivalence Studies*, 2nd ed. Marcel Dekker, Inc.

Clayton and Leslie (1981). The bioavailability of erythromycin stearate versus enteric-coated erythromycin base taken immediately before and after food. *Int Med Res* 9:470-7.

Hauschke, Steinijans, Diletti, Schall, Luus, Elze and Blume (1994). Presentation of the intra-subject coefficient of variation for sample size planning in bioequivalence studies. *Int J Clin Pharm Ther* 32(7): 376-378.

Hyslop, Hsuan and Holder (2000). A small sample on confidence interval approach to assess individual bioequivalence. *Statist Medicine* 19:2885-97.

Schuirmann (1987). A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. *J Pharmacokinet Biopharm* 15:657-680.

US FDA Guidance for Industry (August 1999). *Average, Population and Individual Approaches to Establishing Bioequivalence*.

US FDA Guidance for Industry (October 2000). *Bioavailability and Bioequivalence Studies for Orally Administered Drug Products—General Considerations*.

US FDA Guidance for Industry (January 2001). *Statistical Approaches to Establishing Bioequivalence*.

In vitro-in vivo correlation and formulation science

Charter and Gull (1987). *J Pharmacokinet Biopharm* 15, 645-55.

Cutler (1978). Numerical deconvolution by least squares: use of prescribed input functions. *J Pharmacokinet Biopharm* 6(3):227-41.

Cutler (1978). Numerical deconvolution by least squares: use of polynomials to represent the input function. *J Pharmacokinet Biopharm* 6(3):243-63.

Daubechies (1988). Orthonormal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics* 41(XLI):909-96.

Gabrielsson and Weiner (2001). *Pharmacokinetic and Pharmacodynamic Data Analysis: Concepts and Applications*, 3rd ed. Swedish Pharmaceutical Press, Stockholm.

Gibaldi and Perrier (1975). *Pharmacokinetics*. Marcel Dekker, Inc, New York.

Gillespie (1997). Modeling Strategies for In Vivo-In Vitro Correlation. In Amidon GL, Robinson JR and Williams RL, eds., *Scientific Foundation for Regulating Drug Product Quality*. AAPS Press, Alexandria, VA.

Haskel and Hanson (1981). *Math. Programs* 21, 98-118.

Iman and Conover (1979). *Technometrics*, 21,499-509.

Loo and Riegelman (1968). New method for calculating the intrinsic absorption rate of drugs. *J Pharmaceut Sci* 57:918.

Madden, Godfrey, Chappell MJ, Hovroka R and Bates RA (1996). Comparison of six deconvolution techniques. *J Pharmacokinet Biopharm* 24:282.

- Meyer (1997). IVIVC Examples. In Amidon GL, Robinson JR and Williams RL, eds., *Scientific Foundation for Regulating Drug Product Quality*. AAPS Press, Alexandria, VA.
- Polli (1997). Analysis of In Vitro-In Vivo Data. In Amidon GL, Robinson JR and Williams RL, eds., *Scientific Foundation for Regulating Drug Product Quality*. AAPS Press, Alexandria, VA.
- Treitel and Lines (1982). Linear inverse--theory and deconvolution. *Geophysics* 47(8):1153-9.
- Verotta (1990). Comments on two recent deconvolution methods. *J Pharmacokinet Biopharm* 18(5):483-99.

Pharmacokinetics

- Brown and Manno (1978). ESTRIP, A BASIC computer program for obtaining initial polyexponential parameter estimates. *J Pharm Sci* 67:1687-91.
- Chan and Gibaldi (1982). Estimation of statistical moments and steady-state volume of distribution for a drug given by intravenous infusion. *J Pharm Bioph* 10(5):551-8.
- Cheng and Jusko (1991). Noncompartmental determination of mean residence time and steady-state volume of distribution during multiple dosing, *J Pharm Sci* 80:202.
- Dayneka, Garg and Jusko (1993). Comparison of four basic models of indirect pharmacodynamic responses. *J Pharmacokin Biopharm* 21:457.
- Endrenyi, ed. (1981). *Kinetic Data Analysis: Design and Analysis of Enzyme and Pharmacokinetic Experiments*. Plenum Press, New York.
- Gabrielsson and Weiner (1997). *Pharmacokinetic and Pharmacodynamic Data Analysis: Concepts and Applications*, 2nd ed. Swedish Pharmaceutical Press, Stockholm.
- Gibaldi and Perrier (1982). *Pharmacokinetics*, 2nd ed. Marcel Dekker, New York.
- Gouyette (1983). Pharmacokinetics: Statistical moment calculations. *Arzneim-Forsch/Drug Res* 33 (1):173-6.
- Holford and Sheiner (1982). Kinetics of pharmacological response. *Pharmacol Ther* 16:143.
- Jusko (1990). Corticosteroid pharmacodynamics: models for a broad array of receptor-mediated pharmacologic effects. *J Clin Pharmacol* 30:303.
- Koup (1981). Direct linear plotting method for estimation of pharmacokinetic parameters. *J Pharm Sci* 70:1093-4.

Kowalski and Karim (1995). A semicompartamental modeling approach for pharmacodynamic data assessment. *J Pharmacokinet Biopharm* 23(3):307-22.

Metzler and Tong (1981). Computational problems of compartmental models with Michaelis-Menten-type elimination. *J Pharmaceutical Sciences* 70:733-7.

Nagashima, O'Reilly and Levy (1969). Kinetics of pharmacologic effects in man: The anticoagulant action of warfarin. *Clin Pharmacol Ther* 10:22.

Wagner (1975). *Fundamentals of Clinical Pharmacokinetics*. Drug Intelligence, Illinois.

Regression and modeling

Akaike (1978). Posterior probabilities for choosing a regression model. *Annals of the Institute of Mathematical Statistics* 30:A9-14.

Allen and Cady (1982). *Analyzing Experimental Data By Regression*. Lifetime Learning Publications, Belmont, CA.

Bard (1974). *Nonlinear Parameter Estimation*. Academic Press, New York.

Bates and Watts (1988). *Nonlinear Regression Analyses and Its Applications*. John Wiley & Sons, New York.

Beck and Arnold (1977). *Parameter Estimation in Engineering and Science*. John Wiley & Sons, New York.

Belsley, Kuh and Welsch (1980). *Regression Diagnostics*. John Wiley & Sons, New York.

Corbeil and Searle (1976). Restricted maximum likelihood (REML) estimation of variance components in the mixed models, *Technometrics*, 18:31-8.

Cornell (1962). A method for fitting linear combinations of exponentials. *Biometrics* 18:104-13.

Davies and Whitting (1972). A modified form of Levenberg's correction. Chapter 12 in *Numerical Methods for Non-linear Optimization*. Academic Press, New York.

DeLean, Munson and Rodbard (1978). Simultaneous analysis of families of sigmoidal curves: Application to bioassay, radioligand assay and physiological dose-response curves. *Am J Physiol* 235(2):E97-E102.

Draper and Smith (1981). *Applied Regression Analysis*, 2nd ed. John Wiley & Sons, NY.

- Fai and Cornelius (1996). Approximate f-tests of multiple degree of freedom hypotheses in generalized least squares analysis of unbalanced split-plot experiments. *J Stat Comp Sim* 554:363-78.
- Fletcher (1980). Practical methods of optimization, Vol. 1: *Unconstrained Optimization*. John Wiley & Sons, New York.
- Foss (1970). A method of exponential curve fitting by numerical integration. *Biometrics* 26:815-21.
- Giesbrecht and Burns (1985). Two-stage analysis based on a mixed model: Large sample asymptotic theory and small-sample simulation results. *Biometrics* 41:477-86.
- Gill, Murray and Wright (1981). *Practical Optimization*. Academic Press.
- Gomeni and Gomeni (1979). AUTOMOD: A Polyalgorithm for an integrated analysis of linear pharmacokinetic models. *Comput Biol Med* 9:39-48.
- Hartley (1961). The modified Gauss-Newton method for the fitting of nonlinear regression functions by least squares. *Technometrics* 3:269-80.
- Jennrich and Moore (1975). Maximum likelihood estimation by means of nonlinear least squares. *Amer Stat Assoc Proceedings Statistical Computing Section* 57-65.
- Kennedy and Gentle (1980). *Statistical Computing*. Marcel Dekker, New York.
- Koch (1972). The use of nonparametric methods in the statistical analysis of the two-period change-over design. *Biometrics* 577-83.
- Kowalski and Karim (1995). A semicompartamental modeling approach for pharmacodynamic data assessment, *J Pharmacokinet Biopharm* 23:307-22.
- Leferink and Maes (1978). STRIPACT, An interactive curve fit programme for pharmacokinetic analyses. *Arzneim Forsch* 29:1894-8.
- Longley (1967). *Journal of the American Statistical Association*, 69:819-41.
- Nelder and Mead (1965). A simplex method for function minimization. *Computing Journal* 7:308-13.
- Parsons (1968). Biological problems involving sums of exponential functions of time: A mathematical analysis that reduces experimental time. *Math Biosci* 2:123-8.
- PCNonlin. Scientific Consulting Inc., North Carolina, USA.
- Peck and Barrett (1979). Nonlinear least-squares regression programs for microcomputers. *J Pharmacokinet Biopharm* 7:537-41.

- Ratkowsky (1983). *Nonlinear Regression Modeling*. Marcel Dekker, New York.
- Satterthwaite (1946). An approximate distribution of estimates of variance components. *Biometrics Bulletin* 2:110-4.
- Schwarz (1978). Estimating the dimension of a model. *Annals of Statistics* 6:461-4.
- Sedman and Wagner (1976). CSTRIP, A FORTRAN IV Computer Program for Obtaining Polyexponential Parameter Estimates. *J Pharm Sci* 65:1001-10.
- Shampine, Watts and Davenport (1976). Solving nonstiff ordinary differential equations - the state of the art. *SIAM Review* 18:376-411.
- Sheiner, Stanski, Vozeh, Miller and Ham (1979). Simultaneous modeling of pharmacokinetics and pharmacodynamics: application to d-tubocurarine. *Clin Pharm Ther* 25:358-71.
- Smith and Nichols (1983). Improved resolution in the analysis of the multicomponent exponential signals. *Nuclear Instrum Meth* 205:479-83.
- Tong and Metzler (1980). Mathematical properties of compartment models with Michaelis-Menten-type elimination. *Mathematical Biosciences* 48:293-306.
- Wald (1943). Tests of statistical hypotheses concerning several parameters when the number of observations is large. *Transaction of the American Mathematical Society* 54.

Sparse data analysis methods

- Bailer (1988). Testing for the equality of area under the curves when using destructive measurement techniques. *J Pharmacokinet Biopharm* 16:303-9.
- Holder (2001). Comments on Nedelman and Jia's extension of Satterthwaite's approximation applied to pharmacokinetics. *J Biopharm Stat* 11(1-2):75-9.
- Mager and Goller (1998). Resampling methods in sparse sampling situations in preclinical pharmacokinetic studies. *J Pharm Sci* 87(3):372-8.
- Nedelman, Gibiansky and Lau (1995). Applying Bailer's method for AUC confidence intervals to sparse sampling. *Pharm Res* 12:124-8.
- Nedelman and Jia (1998). An extension of Satterthwaite's approximation applied to pharmacokinetics. *J Biopharm Stat* 8(2):317-28.
- Tang-Liu and Burke (1988). The effect of azone on ocular levobunolol absorption: Calculating the area under the curve and its standard error using tissue sampling compartments. *Pharm Res* 5:238-41.

Yeh (1990). Estimation and Significant Tests of Area Under the Curve Derived from Incomplete Blood Sampling. *ASA Proceedings of the Biopharmaceutical Section* 74-81.

Statistics

Allen (2001). Private correspondence.

Conover (1980). *Practical Nonparametric Statistics* 2nd ed. John Wiley & Sons, New York.

Dixon and Massey (1969). *Introduction to Statistical Analysis*, 3rd ed. McGraw-Hill Book Company, New York.

Koch (1972). The use of non-parametric methods in the statistical analysis of the two-period change-over design. *Biometrics* 577-84.

Kutner (1974). Hypothesis testing in linear models (Eisenhart Model 1). *The American Statistician*, 28(3):98-100.

Searle (1971). *Linear Models*. John Wiley & Sons, New York.

Steel and Torrie (1980). *Principles and Procedures of Statistics; A Biometrical Approach*, 2nd ed. McGraw-Hill Book Company, New York.

Winer (1971). *Statistical Principles in Experimental Design*, 2nd ed. McGraw-Hill Book Company, New York, et al.

Index

Symbols

- *.anv, 15, 354, 488
- *.bmp, 16
- *.cmd, 15, 491
- *.dep, 37
- *.dll, 195
- *.exp, 102, 103, 105
 - creating, 102
 - saving, 103, 106
- *.imp, 87
 - creating, 87
- *.jpeg, 16
- *.lib, 15, 195, 220
- *.lml, 15, 354, 399, 400, 497
 - saving, 364
- *.map, 79
- *.pco, 14
- *.pks, 459
- *.pmo, 15, 491
- *.psc, 15, 422
- *.pto, 15
- *.pwo, 14
- *.tdf, 148, 503
- *.txt, 195
- *.wdo, 15
- *.wgo, 15
- *.wmf, 16
 - placeable header information, 16
- *.wmo, 15

- *.wsc, 15, 422
- *.wsp, 19
- *.wto, 15
- *.xpt, 78

A

- A
 - defined, 487
- Absolute error bars, 120
- Accumulation index, 487
- Actual clock time
 - defined, 487
- Actual time
 - defined, 487
- AggregateVar, 583
- Akaike Information Criterion
 - AIC defined, 487
 - in LinMix, 367
 - in LinMix vs SAS PROC MIXED, 377
 - in PK text output, 215
 - in PK workbook output, 217
- Alias
 - PKS, 455
- Alignment, 48
 - center across cells, 48
 - fill, 48
 - table cells, 145
- Alpha, 488

- A, 487
 - half-life, 488
- AlphaTerms(), 583
- AND, 237
- Anderson-Hauck, 390
 - pval, 488
 - statistic, defined, 488
- ANOVA, 376
- APPEND, 584
- AppendFilename, 584
- AppendFileType, 584
- AppendSheet, 585
- ARRANGE, 585
- ASCII data files, 16
- ASCII models
 - block content, 223
 - comments, 224
 - comparison operators, 237
 - creating, 220
 - examples, 228
 - functions, 237
 - operators, 235
 - programming language, 235
 - reserved variable names, 234
 - statements, 235
 - structure, 222
 - variables, 233
 - weights, 245
- Aterms(), 585
- AUC, 215
 - AUCall, 488
 - AUCINF, 489
 - AUClast, 489
 - calculation formulas, 185
 - calculation methods, 179
 - compartmental modeling, 215
 - defined, 488
 - partial area calculations, 186
- Audit trail, 474
 - reason, 474
- AUMC
 - AUMCINF, 489

- AUMClast, 489
 - defined, 489
- AutoFileOptions, 586
- Automatic sorting, 117
- Autoregressive variance, 358
- AutoSort, 586
- Average bioequivalence, 379, 383
 - fixed effects, 400
 - recommended models, 383
 - setting up, 400
- Axes
 - log, 126
 - log base, 127
 - options, 122
 - uniform, 123

B

- B
 - defined, 489
- Balanced design, 489
- Banded no-diagonal factor analytic, 358
- Banded Toeplitz variance, 358
- Banded unstructured variance, 358
- Bar charts, 117
- Baseline
 - command, 561
 - NCA model 220, 172
- Beta
 - B, 489
 - defined, 489
 - half-life, 489
- Bibliography, 661
- Bioassay functions, 238
- Bioequivalence, 379, 489
 - average, 400
 - Bioequivalence Wizard, 399
 - classification variables, 401
 - dependent variables, 401
 - errors, 653
 - fixed effects (average), 400

- fixed effects (pop./individ.), 406
- general options, 405
- least squares means, 385
- linear model, 382
- missing data, 382
- options, 404, 406
- output, 407
- population/individual, 405
- power, 500
- random effects, 402
- regressor variables, 401
- repeated variance, 403
- sort variables, 401
- sums of squares, 385
- type, 399
- types, 489
- variable name limitations, 382
- variance structure, 402
- weight variables, 401

Bitmaps, 16

Blinding type, 458

BOLUS, 586

Bounds, 199

- defined, 490

Box and whiskers, 152, 490

BQL

- defined, 490

C

C

- defined, 490

C++, 219

Carry along data for like sort levels, 59

Carry along variables, 490

CarryData, 586

CASCADE, 586

CaseSensitive, 587

Cavg, 490

Cell references, 44

Cells

- alignment, 48
- cell references, 44
- center across cells, 48
- deleting, 43
- deleting formats, 49
- deleting values, 49
- filling adjacent, 46
- formatting, 46
- formulas, 44
- including or excluding, 60
- inserting, 43
- merge, 48

Center across cells, 48

Chart Wizard, 115

Charts, 115

- automatic sorting, 117
- axis options, 122
- axis titles, 122
- bar charts, 117
- box and whiskers, 490
- changing chart type, 124
- copy and paste, 130
- creating, 115
- error bars, 120
- grid lines, 127
- group variables, 119
- histograms, 118
- intervals, 123
- legend, 122
- log, 123, 126
- multiple X-Y variables, 117
- overlay, 117
- page setup, 25
- Pharsight Chart Objects, 15
- pink, 33
- save as *.wmf, 16
- saving to graphics formats, 16
- scatter plots, 117
- sort key title, 122
- sort variables, 119
- templates, 129
- title, 122

- uniform axes, 123
- units display default, 31
- use group headers, 122
- X-Y variables, 116
- CL, 491
 - CLR, 491
 - Clss, 491
 - Clss/F, 491
- Classical and Westlake Confidence Intervals, 387
- Classification variables
 - bioequivalence, 401
 - defined, 491
 - LinMix, 356
- Clock time, 491
- CloseAll, 587
- Cmax, 491
- Cmin, 491
- CMT, 83
- Coefficients, 369
- COLUMN, 587
- Columns
 - cell references, 44
 - deleting, 44
 - freezing, 56
 - including or excluding, 60
 - inserting, 43
 - names, 53
 - units, 53
- COMMAND, 223
- Command block, 225
- Command file, 491
- COMMANDS, 225
- Comments
 - comment character, 82
 - in ASCII models, 224
- Comparison operators, 237
- Compiling
 - Using C++, 240
- Compound, 458
 - defined, 491
- Compound symmetry variance, 358
- CON array, 234
- ConcCol, 588
- Concentration-effect models, 540
- COND, 214, 491
- Condition number, 491
- CONFIDENCE, 588
- Confidence intervals, 214
 - descriptive statistics, 152
 - planar, 500
 - univariate, 503
 - Westlake, 504
- Confidence limits
 - Univariate, 215
- Configuration
 - PKS, 455
- Connection string, 90
 - ODBC export, 102
 - ODBC import, 87
- CONSEQDELIM, 588
- Constants
 - defined, 491
- Constrained optimization, 7
- Contrasts
 - defined, 492
 - degrees of freedom, 353
 - LinMix, 350, 360
 - nonestimability, 352
- Control stream, 83
- Convergence criterion, 209, 377
 - defined, 492
 - LinMix, 373
- Convolution, 264, 268, 295
 - settings, 299
 - units, 301
 - variables, 297
- COPY, 589
- Copying
 - charts, 130
 - workbooks, 14
 - worksheet cell fill-down, 46
- Correlation matrix, 215
- Covariates, 356, 401

- defined, 492
- Crossed factors, 492
- Crossover design, 256
 - separate columns, 261
 - stacked in one column, 261
- CROSSVAR(), 589
- CROSSVARS, 589
- CSS, 217
- Cubic model, 551
- Cumul_Amt_Abs_V, 280
- Curve stripping, 167
 - defined, 492
 - exponential, 493
- Custom query builder, 97
- Customer support, 10
 - licensing, 10
 - user feedback, 11
- CUT, 590

D

- DATA, 571
- Data
 - dependencies, 33
 - file type limitations, 14
 - file types, 13
 - inclusions and exclusions, 60
 - merging, 57
 - reset selections, 62
 - sorting, 57
 - sparse sampling, 190, 518
 - toxicokinetic, 518
 - transformations, 62, 64
- Data variables
 - compartmental modeling, 198
 - noncompartmental analysis, 163
- Dates, 457
- DCP description, 465
- Decimal places
 - SAS Transport export, 79
 - tables, 146

- Deconvolution, 263, 264
 - Loo-Reigelman, 281
 - numeric, 262
 - references, 662
 - UIR, 271
 - units and UIR, 271
 - Wagner-Nelson, 280
- DefaultNCAParameterNames.map, 182
- Defaults
 - missing value indicator, 28
 - modeling output, 29
 - NCA calculation method, 29
 - page breaks in tables, 31
 - units, 31
 - workbooks, 28
- Defect reports, 11
- DefinitionFile, 590
- Degrees of freedom
 - LinMix contrasts, 353
 - LinMix versus PROC MIXED, 376, 377
 - Satterthwaite's approximation, 374
- DeleteFiles, 591
- Deleting
 - cell formats, 49
 - cell values, 49
 - cells, 43
 - columns, 44
 - rows, 44
 - worksheets, 42
- DELIMITER, 591
- DELTA, 591
- Dependencies, 33
 - *.dep, 37
 - origins, 36
 - refreshing output, 35
 - saving, 35
- Dependent variables
 - bioequivalence, 401
 - LinMix, 356
 - NONMEM export, 83
- Description

- PKS studies, 458
- Descriptive statistics, 151
 - KS_pval, 157
- DestArea, 592
- DestCol, 592
- Detach, 36
- Determining estimability, 337
- DF, 217
- DIFFERENTIAL, 223, 227
- Differential equations, 4
 - user models, 227
- Directories
 - options, 28
- Disable curve stripping, 167
- DNAMES, 226
- DnErr(), 592
- Do, 236
- Document manager, 471
- DoseFile, 434, 593
- DoseUnit, 593
- Dosing
 - ASCII models, 204
 - compiled PK models, 203
 - data variables, 204
 - dose normalization, 173, 203
 - file structure, 436
 - noncompartmental analysis, 173
 - NONMEM export, 84
 - PKS, 452, 476
 - regimen, 202
 - scenario-specific, 476
 - simulation, 417
 - Tau, 205
 - units, 173
 - worksheet data, 204
- Down variable, 120
- DTA, 232
- DTA array, 233
- DVID, 83
- DZ array, 234

E

- E, 492
- E0, 493
- EC50
 - defined, 492
- ECOL, 593
- Eigenvalue
 - defined, 493
- Eigenvalues, 215, 368
- Else, 236
- Emax
 - defined, 493
- END, 223
- EndCol, 593
- EndRow, 594
- Engines, 427
 - defined, 493
 - scripting language, 422
- Enhancement requests, 11
- EntireRow, 594
- EQ operator, 237
- Error bars, 120
 - absolute, 120
 - relative, 120
- Error messages, 641
 - bioequivalence, 653
 - compartmental modeling, 641
 - descriptive statistics, 656
 - IVIVC, 658
 - LinMix, 653
 - noncompartmental analysis, 657
 - scripting, 659
 - tables, 656
- Errors, 212
- ErrType, 595
- Estimates
 - fixed effects, 368
 - LinMix, 353, 362
- Examples
 - Examples Guide, 2
- Excel

- Excel 4.0 files, 15
- files, 16
 - saving workbooks to Excel, 16
- EXCLUDE, 595
- Exclude profiles with insufficient data, 176
- Exclusions, 60
 - based on criteria, 60
 - by selection, 60
 - from Lambda Z calculations, 168, 288
 - reset selections, 62
- Exponential curve stripping, 493
- Export definition, 102, 103
- ExportAll, 595
- Exporting
 - NONMEM, 80
 - ODBC, 102
 - SAS Transport files, 78
 - SigmaPlot, 85
 - S-PLUS, 86
 - Word export, 76
 - Word export options, 77, 78
- EXTRA, 596

F

- F, 233
- Fabs
 - defined, 493
- Fdiss
 - defined, 493
- File types
 - WNL and WNM compatibility, 17
- FileMask, 597
- Files
 - ASCII, 16
 - data file limitations, 14
 - data files, 13
 - PKS files, 445
 - SAS Transport, 78
- FileType, 597

- Fill-down, 46
- Final parameters
 - confidence intervals, 214
- FIND, 598
- Find, 50
 - find next, 51
 - replace, 51
- Fit
 - Akaike Information Criterion, 217
 - Schwarz Bayesian Criterion, 217
- Fixed effects, 340
 - average bioequivalence, 400
 - defined, 493
 - LinMix, 340, 356
 - parameter estimates, 368
 - pop./individ. bioequivalence, 406
- Fixed parameters, 200
- FNUMBER, 233
- FONT, 598
- FontSize, 598
- FootnoteFont, 599
- FootnoteFontSize, 599
- Formatting
 - tables, 146
 - worksheet cells, 46
- Formulas, 44
 - operators, 45
- Formulation
 - defined, 493
- FORTTRAN, 219
 - user models, 239
- FUNCTION, 223, 227, 229, 599
 - multiple FUNC blocks, 225
- Function variable, 225, 493
- Functions, 237

G

- Gamma
 - defined, 494
 - half-life, 494

- Gauss-Newton algorithm, 6, 494
 - Hartley modification, 5, 9
 - Levenberg Hartley modification, 8
 - Levenberg-type modification, 5
- GE operator, 237
- Geometric mean
 - defined, 494
- Go to, 52
- GOTO, 235
- Green
 - scenario icon, 467
- Group variables, 494
 - charts, 119
- GroupVar(,), 601
- GroupVar(), 600
- GroupVarBreak(), 601
- GroupVars, 601
- GroupVars(), 601
- GT operator, 237

H

- HALT, 602
- Harmonic mean, 494
- HEADERS, 602
- Hessian eigenvalues, 368
- Heterogeneous
 - autoregressive variance, 358
 - compound symmetry, 358
- Hidden windows, 18
- Hill model, 552
- Histograms, 118
 - intervals, 123
- History
 - "cannot be processed", 38
 - items logged, 38
 - worksheets, 37

I

- IConsumer interface, 111

- ID variables, 495
- IDVar(), 602
- IDVars, 603
- If Then Else, 235
- Importing
 - ODBC, 87
- INCLUDE, 603
- Include placeable header information, 16
- IncludeVar(,), 603
- IncludeVars(), 604
- Including
 - based on criteria, 60
 - selected cells, 60
- Inclusions, 60
 - based on criteria, 60
 - by selection, 60
 - reset selections, 62
- Increment for partial derivatives, 209, 495
- InData, 604
- InData(), 604
- Independent variable, 495
- Indication, 458
- Indirect response models, 543, 545
 - defined, 495
 - parameter estimates and bounds, 546
 - parameters, 546
- Individual bioequivalence, 380
- INITIAL, 572
- Initial parameter estimates, 199
- InitialDose, 604
- InputLag, 605
- Inserting
 - cells, 43
 - columns, 43
 - rows, 43
 - worksheets, 42
- Interaction, 495
- Intercept, 495
- Interpolation
 - linear trapezoidal, 179, 294
 - linear trapezoidal with linear/log, 179, 294

- linear up/log down, 180, 294
- linear/log trapezoidal, 180, 294
- INTERVAL, 605
- Intervals, 123
- IPR, 495
- IQueryBuilder interface, 110
- Iterations, 210
 - history, 369
 - procedure, 5
- Iterative reweighting, 244, 495
- IVIVC, 279
 - convolution, 295
 - correlation models, 313
 - dissolution data and models, 308
 - dissolution models, 551
 - error messages, 658
 - fixed parameters, 200
 - fraction absorbed, 309
 - Levy plots, 302
 - loading projects, 307
 - Loo-Reigelman, 281
 - minimizing the wizard, 306
 - model validation, 313
 - options, 318
 - output options, 289
 - predicting PK, 316
 - project status, 306
 - saving projects, 307
 - toolkit, 279
 - toolkit introduced, 2
 - unit impulse response fit, 309
 - use raw data, 309, 318
 - Wagner-Nelson, 280
 - wizard, 304
- IVIVC In vivo data options, 311

J

- Join variables, 496
 - merge data sets, 57
 - tables, 143

- JoinCrossVar(,), 605
- JoinCrossVars, 606
- JoinGroupVar(,), 606
- JoinGroupVars, 606
- JoinIDVar(,), 607
- JoinIDVars, 607
- Joint Contrasts, 351
- JPEG files, 16

K

- K half-life, 496
- K0, 496
- K01, 496
- K10, 496
 - half-life, 496
- KE0, 607
- KM, 496
- Kolmogorov-Smirnov normality test, 157
- KS_pval, 157

L

- LABEL, 235
- Lag time
 - defined, 496
- Lambda z, 164
 - defined, 496
 - not estimable, 166
 - range file structure, 435
- LamzFile, 434, 608
- LE operator, 237
- Least squares means, 4, 385
 - defined, 496
 - least squares fitting, 496
- LinMix, 348, 363
- Legend(), 608
- LegendFont, 608
- LegendFontSize, 608
- Legends, 122
- Levy plots, 302

- Libraries, 480
 - add or update, 481
 - loading from PKS, 481
 - status, 482
- Licensing
 - customer support, 10
 - expiration warning, 31
- Linear interpolation rule, 186
- Linear kinetics, 497
- Linear models, 3
- Linear regression, 3, 550
 - defined, 497
 - minimization algorithm, 208
- Linear trapezoidal rule, 179, 185, 294
 - with linear/log interpolation, 179, 294
- Linear up/log down, 180, 294
- Linear/log trapezoidal rule, 180, 294
- Link models, 540
 - simultaneous, 549
- LinkFile, 434, 609
- LinMix, 327
 - Akaike's criterion, 367
 - classification variables, 356
 - coefficients, 369
 - command file, 497
 - compared to ANOVA, 376
 - compared to PROC MIXED, 376
 - computations, 370
 - contrasts, 350, 360
 - convergence criterion, 373
 - crossed factors, 492
 - dependent variables, 356
 - errors, 653
 - estimability, 337
 - estimates, 353, 362
 - fixed effects, 340, 356
 - fixed effects estimates, 368
 - general options, 364
 - Hessian eigenvalues, 368
 - hypothesis tests, 366
 - initial variance parameters, 364
 - iteration history, 369
 - least squares means, 348, 363
 - limits and constraints, 354
 - linear mixed effects model, 330
 - LinMix Wizard, 354
 - Newton's algorithm, 372
 - non-estimable functions, 338
 - output, 365
 - output parameterization, 340
 - parameters, 369
 - partial tests, 366
 - predicated values, 368
 - random effects, 345, 359
 - regressor variables, 356
 - REML, 370
 - repeated effects, 343, 360
 - residuals, 368
 - Satterthwaite's approximation, 374
 - Schwarz's criterion, 367
 - sequential tests, 366
 - singularity tolerance, 341
 - sort variables, 356
 - text output, 365
 - transformation of the response, 341
 - variable name limitations, 355
 - variance structure, 342, 357
 - weight variables, 356
 - workbook output, 365
 - X matrix, 332
- LLOQ, 463
 - defined, 497
 - use when less than LLOQ, 74
- LOAD, 609
- Loading
 - ODBC import query, 96
- LoadTemplate, 609
- Locked, 483
- Logarithmic
 - axes, 123
 - interpolation rule, 186
 - trapezoidal rule, 186
- Logarithms
 - transformations, 62

Loo-Reigelman, 281
 output options, 289
 weighting, 290
 workbook output, 290
LOWER, 214, 572
LT operator, 237

M

Macro rate constants, 497
 stripping dose, 502
MaintenanceDose, 610
Makoid-Banakar model, 554
Marking valid/invalid data, 35
Mathematical operators, 235
Maximum likelihood estimates, 4
mdllib_md1602, 553
mdllib_md1603, 553
mdllib_md1604, 554
Mean
 defined, 497
Mean square, 210
Mean square variance, 497
Merge
 cells, 48
Merging, 57
 carry along for like sort levels, 59
Method
 scripting language, 422
Methods
 defined, 497
Michaelis-Menten, 546
 defined, 497
 equation, 3
 Mm.lib, 547
 Vmax, 503
Micro rate constants, 498
Microsoft Visual C++, 241
Microsoft Word
 export, 76
Minimization algorithm, 208

MISSING, 610
Missing values
 default indicator, 28
 in NCA parameter estimates, 184
Mm.lib, 547
MODEL, 223
Model 220, 519
 baseline, 169
 partial areas, 169
 slopes data ranges, 164
 threshold, 169
Model options, 175, 205
 convergence criterion, 209
 exclude profiles with insufficient data, 176
 increment for partial derivatives, 209
 iterations, 210
 mean square, 210
 minimization, 208
 model size, 210
 number of predicted values, 209
 output, 206
 output intermediate calculations, 176
 PK settings, 208
 propagate final estimates, 200
 title, 208
 title for NCA text output, 178
 transpose final parameter table, 180, 208
 units, 210
 weighting for NCA, 177
Model parameters, 199
 bounds, 199
 initial estimates, 201
 ParmFile, 200
Model properties
 compartmental models, 198
 dosing, 202
Model size, 210
Model variables
 compartmental modeling, 198
 noncompartmental analysis, 163

Modeling, 195

- ASCII model number, 223
- convergence criterion, 209
- data variables, 198
- default output, 29
- dosing, 202
- error messages, 641
- errors, 212
- initial parameter estimates, 199
- iterations, 210
- linear models, 3
- mean square, 210
- minimization algorithm, 208
- model options, 205
- model settings, 198
- model size, 210
- NONMEM export, 80
- notation and conventions, 508
- number of predicted values, 209
- options, 29
- output, 212
- output dependencies, 33
- output options, 206
- PK settings, 208
- refreshing output, 35
- scaling of weights, 207
- S-PLUS export, 86
- start, 211
- stop, 211
- troubleshooting, 211
- units, 210
- weights, 207

Models

- concentration-effect, 540
- creating ASCII models, 220
- cubic, 551
- double Weibull, 553
- Hill, 552
- indirect response, 495, 543
- linear, 550
- link, 540
- loading, 195

- Makoid-Banakar, 554
- Michaelis-Menten, 546
- noncompartmental, 162
- noncompartmental analysis, 509
- pharmacokinetic, 523
- Pharsight Model Objects, 15
- sigmoidal/dissolution, 551
- simultaneous link, 549
- types, 195
- user models, 219
- Weibull, 553

Modified likelihood, 377

MRT, 498

- MRTINF, 498

- MRTlast, 498

Multiple linear regression, 3

Multiple profiles in script files, 432

N

N

- modeling output, 498

Name objects, 476

Naming

- columns, 53

Native format, 475

NCA

- model 220 baseline, 172

- model 220 threshold, 172

NCONSTANTS, 225

NDERIVATIVES, 225

Nelder-Mead simplex algorithm, 5, 8

- defined, 498

Nested factors, 498

Nested variables, 356

Newton's algorithm, 372

Next, 236

NFUNCTIONS, 225

Nmiss, 498

No Intercept, 332

No Scaling of Weights, 208

- Nobs, 498
 - NOBSERVATIONS, 571
 - No-diagonal factor analytic, 358
 - Nominal time
 - defined, 498
 - Noncompartmental analysis, 161
 - AUC formulas, 185
 - chart output, 176
 - computational rules, 183
 - data exclusions, 184
 - data pre-treatment, 183
 - default calculation method, 29, 30
 - default parameter names file, 182
 - defined, 498
 - disable curve stripping, 167
 - dosing, 173
 - drug effect data, 519
 - errors, 657
 - exclude profiles with insufficient data, 176
 - include parameter in output, 182
 - Lambda z, 164
 - Lambda z file structure, 435
 - missing parameter values, 184
 - model selection, 162
 - model variables, 163
 - models, 509
 - output, 175
 - output intermediate calculations, 176
 - output options, 175
 - output text, 176
 - page breaks in text output, 176
 - parameter calculations, 510
 - parameter names, 181
 - partial area computations, 186
 - partial areas, 169
 - partial areas file structure, 443
 - precision of output, 181
 - slope data ranges, 164
 - sparse sampling data, 162
 - therapeutic response window, 170
 - therapeutic window calculations, 188
 - units, 180
 - weighting, 177
 - workbook output, 176
 - Nonlinear kinetics
 - defined, 499
 - Nonlinear models, 3, 499
 - Nonlinear regression, 3, 5
 - methods, 8
 - NONMEM export, 80
 - CMT, 83
 - comment character, 82
 - control stream, 83
 - dependent variables, 83
 - dosing, 84
 - DVID, 83
 - occasions, 85
 - other data items, 83
 - problem, 82
 - Non-numeric data
 - troubleshooting, 16
 - Nonparametric superposition, 247
 - Normality test, 157
 - NPARAMETERS, 225, 572
 - NSECONDARY, 225
 - Number format, 47
 - Number of predicted values, 209
- O**
- Object
 - defined, 499
 - Objects
 - Scripting Language, 422, 425
 - Occasions, 85
 - ODBC
 - defined, 499
 - ODBC export, 102
 - building an export definition, 104
 - connection string, 90, 102
 - creating an export, 102
 - definition, 103

- export definition, 102
- loading a saved connection, 92
- saving export settings, 105
- ODBC import, 87
 - building a query, 93
 - connection string, 87, 90
 - custom query builder, 97
 - custom query builder example, 111
 - import settings file, 87
 - loading a saved connection, 92
 - save password, 89
 - saving, 89
 - software developer's kit, 107
 - Watson DMLIMS, 97
- Operators, 237
- Options, 28
 - bioequivalence, 404, 405, 406
 - directories, 28
 - license expiration warning, 31
 - LinMix, 364
 - models, 29
 - NCA model, 175
 - NCA output, 175
 - tables, 31
 - WinNonlin options, 28
 - Word export, 77, 78
 - workbooks, 28
- OR, 237
- Origins, 36
- Other data items
 - NONMEM export, 83
- Output
 - compartmental models, 212
 - pink, 33
 - refreshing, 35
 - unlocking, 36
- Output intermediate calculations, 176
- Overlay chart, 117

P

- P array, 234
- Page breaks
 - NCA output, 176
 - table defaults, 31
- Page setup, 21
- Parameters
 - file structure, 438
 - file structure for link models, 440
 - fixed, 200
 - LinMix output, 369
 - names for NCA, 181
 - ParmFile, 434
 - preferred default names, 182
 - preferred names, 182
- ParmFile, 200, 434
- PartFile, 434
- Partial areas, 169
 - computation of, 186
- Partial derivatives, 4, 7
 - increment, 209, 495
- Partial tests, 366, 499
- Password
 - PKS, 456
 - save ODBC password, 89
- Percentiles
 - descriptive statistics, 152
- Performance
 - PKS, 471
- Pharmacodynamic models, 537
- Pharmacokinetic models, 4, 8, 523
- Pharsight, 9
 - contact information, 10
 - customer feedback, 11
 - products, 9
 - scientific and consulting services, 9
 - technical support, 10
 - training courses, 10
- Pharsight Chart Objects, 15
- Pharsight Model Objects, 15
- Pharsight Text Objects, 15

- Pharsight Workbook Objects, 15
 - Pink output windows, 33
 - PK settings, 208
 - PK text output, 213
 - PK/PD link models, 540
 - defined, 500
 - simultaneous, 549
 - PKS, 1
 - accessing, 453
 - add columns, 454
 - add or update library, 481
 - alias, 455
 - append/update study, 454
 - configuration, 455
 - create study, 453
 - creating scenarios, 469
 - DCP description, 465
 - dosing, 452, 454, 476
 - file format, 445
 - green icon, 467
 - information, 454, 483
 - libraries, 454, 480
 - library and share status, 482
 - load, 453
 - load study or scenario, 467
 - loading libraries, 481
 - logging on, 455
 - non-WinNonlin documents, 470
 - other documents, 454
 - password, 456
 - performance, 471
 - red icon, 467
 - save, 453
 - save as, 453
 - scenario search, 468
 - scenarios, 451
 - scenario-specific dosing, 476
 - server, 455
 - sessions, 454
 - share, 454, 479
 - status, 454
 - studies, 450
 - study data requirements, 456
 - study metadata, 451, 457
 - time, 451
 - user name, 456
 - using with WinNonlin, 449
 - PLANAR, 214
 - Planar confidence interval, 500
 - PNames, 226, 231
 - Population bioequivalence, 380
 - Population/individual bioequivalence, 405
 - data requirements, 405
 - fixed effects, 406
 - Portfolio
 - PKS studies, 458
 - Power, 390
 - Precision
 - NCA output, 181
 - SAS Transport export, 79
 - Pre-clinical data, 190, 518
 - Preferred units, 180
 - Printing, 21, 25
 - page setup, 21
 - print all, 27
 - print preview, 25
 - PrintMulti, 617
 - PrintMultiAcross, 617
 - PrintMultiDown, 617
 - Problem
 - NONMEM, 82
 - PROC MIXED, 377
 - Profile, 500
 - Programming language, 235
 - Project, 458
 - Propagate final estimates, 200
 - Properties
 - defined, 500
 - PUNIT, 226
- Q**
- Quantiles

- computation, 157
- Queries
 - building ODBC queries, 93
 - ODBC import
 - queries, 92
- QueryBuilderInterfaces.dll, 109

R

- Random effect
 - defined, 500
- Random effects
 - bioequivalence, 402
 - LinMix, 345, 359
- RANK, 213
- Rank, 500
- Rank transformations, 68
- Ratio tables, 410
- Read only, 483
- Reason, 474
- Red
 - scenario icon, 467
- References, 661
- Refresh, 35
- Regression methods, 8
 - Gauss-newton algorithm, 8, 9
 - Nelder-Mead simplex, 8
- Regressor variables
 - bioequivalence, 401
 - LinMix, 356
- Regressors
 - defined, 501
- RegVar(), 618
- RegVars, 618
- Relative actual end time, 501
- Relative actual time, 501
- Relative error bars, 120
- Relative nominal end time, 501
- Relative nominal time, 501
- Relative time, 501
- REMARK, 224

- REML, 377
 - LinMix, 370
 - Newton's algorithm, 372
- REMOVE, 619
- RemoveCol, 619
- RemoveRow, 619
- RenameCol, 620
- Repeated
 - bioequivalence, 403
- Repeated effects
 - LinMix, 343, 360
- REPLACE, 620
- Replace, 51, 66
- Residuals, 215
 - defined, 501
- RESPONSE, 621
- Restricted maximum likelihood
 - LinMix, 370
 - Newton's algorithm, 372
- Rich text files, 16
- Rows
 - cell references, 44
 - deleting, 44
 - freezing, 56
 - including or excluding, 60
 - inserting, 43
- Rule sets, 71
- RUN, 621
- RunWordExport, 621

S

- S, 215, 501
 - estimate of residual standard deviation, 217
- S array, 234
- SameErrCol(), 622
- Sample number, 462
- SAS
 - share, 479
- SAS Transport files

- data set name, 79
- export, 78
- Satterthwaite's approximation, 374
- SAVE, 622
- SaveAll, 622
- SaveAllPrefix, 623
- SaveTemplate, 623
- Scaling of weights, 207, 244
- Scatter plots, 117
- Scenarios, 451
 - adding columns, 470
 - adding non-WinNonlin documents, 470
 - creating, 469
 - defined, 502
 - search, 468
- Schwarz Bayesian Criterion
 - in LinMix, 367
 - in LinMix vs SAS PROC MIXED, 377
 - in PK text output, 215
 - in PK workbook output, 217
 - SBC defined, 501
- Scientific notation, 48
- Scripting
 - commands, 583
 - engines, defined, 493
 - error messages, 659
 - file paths (Filename property), 597
 - method, defined, 497
 - object, defined, 499
 - PKS, 444
 - property, defined, 500
 - script file (.PSC), 422
 - script file components, 430
- SD, 502
- SDK, 107
- SE, 502
- SearchArea, 623
- SECONDARY, 223, 228, 229
- Secondary parameters, 216
 - simulation, 419
 - standard error computation, 216
 - user models, 228
- Select Data for Modeling dialog, 211
- Semicompartmental modeling, 252
- SEQUENCE, 624
- Sequential tests, 365, 366
- Server
 - PKS, 455
- Setting preferred units, 210
- Share, 479
 - status, 482
- Show formulas, 28
- ShowUnits, 624
- SigmaPlot
 - export to, 85
 - version, 85
- Significant digits
 - NCA output, 181
 - SAS Transport export, 79
 - tables, 145, 146
- Simplex algorithm, 5, 214
- Simulation, 413
 - comparing dose schemes, 419
 - dosing, 417
 - model parameters, 416
 - secondary parameter estimates, 419
- Simultaneous PK/PD link models, 549
- Singular or near-singular equations, 6
- Singular value decomposition, 6
- Singularity tolerance, 341, 378
- Slopes
 - disable curve stripping, 167
- SMOOTHING, 624
- SNAMES, 226
- SORT, 57, 624
- Sort key title, 122
- Sort variables, 502
 - bioequivalence, 401
 - charts, 119
 - LinMix, 356
- Sorting
 - worksheet data, 57
- SortLevel, 624

SortVar(,), 625
SortVar(), 625
SortVars, 625
SortVars(), 626
Source changed, 35
SourceCol, 626
Sparse data
 computations, 190
 models, 518
 references, 666
 use sparse sampling, 162
S-PLUS
 export to, 86
 version, 86
SS/df (mean square), 497
SSR, 217
STACKED, 626
Standard error, 214
START, 223
StartCol, 627
STARTING, 227
StartRow, 627
STATS, 627
Status code transformations, 69
 LLOQ, 74
 rule sets, 71
Stripping dose
 defined, 502
Studies
 balanced design, 489
 defined (PKS), 502
 description, 458
 design, 458
 locked, 483
 name, 457
 PKS, 450
 start and end dates, 457
 type, 458
SUBJECT, 628
Summary statistics, 151
 tables, 144
Summary variables, 502

SummaryVar(), 628
SummaryVars, 628
Sums of squares, 385
SUNIT, 226

T

TAB, 629
Table Wizard, 131
 cell alignment, 145
 formatting, 146
 significant digits, 145
 summary statistics, 144
 templates, 134
TableNum, 629
Tables, 131
 cell alignment, 145
 column alignment, 146
 data only, 146
 decimal places, 146
 error messages, 656
 excluded data, 133
 font, 147
 formatting, 146
 join variables, 143
 missing data, 133
 options, 31
 page break defaults, 31
 page numbers, 147
 significant digits, 145, 146
 summary statistics, 144
 table definition files, 148
 templates, 134
 units, 147
 units display default, 31
TAU, 629
Tau, 205, 503
Technical support, 10
 licensing, 10
Templates
 charts, 129

- tables, 134
- TEMPORARY, 223, 226
- Temporary variables, 226
- TerminalLower, 629
- TerminalPhase, 630
- TerminalUpper, 630
- Tests of hypotheses, 366
- Text
 - page setup, 24
- Text windows
 - Pharsight Text Objects, 15
 - pink, 33
- Therapeutic response window, 170
 - calculations, 188
- Threshold
 - command, 579
 - NCA model 220, 172
- TI, 503
- TileHorizontal, 630
- TileVertical, 630
- Time
 - actual clock time, 487
 - actual time, 487
 - clock time, 491
 - nominal time, 498
 - relative actual end time, 501
 - relative actual time, 501
 - relative nominal end time, 501
 - relative nominal time, 501
 - relative, defined, 501
- TimeCol, 631
- TimeRepeat, 631
- TITLE, 631
- Title, 178, 208
- TitleFont, 631
- TitleFontSize, 632
- Toeplitz variance, 358
- TOLERANCE, 632
- Tolerance, 51
- Training courses, 10
- TRANSFORM, 223, 224, 246
- Transformations, 62

- action, 66
- equations, 67
- filter, 66
- LinMix response, 341
- multiple, 64
- rank, 64, 68
- replace, 66
- status codes, 69
- Transpose final parameter table, 29
 - compartmental modeling, 208
 - non-compartmental modeling, 180
- Trapezoidal rule, 215
 - default method for NCA, 29
- Treatment
 - PKS mapping, 466
- Troubleshooting, 211
 - loading tabular data, 16
- Trust region, 6
- Types of models, 195

U

- UIR, 271
 - automatic generation, 311
- ULOQ, 463
- Undo, 50
- UNIFORM, 633
- Uniform axes, 123
- UNIT, 633
- Unit impulse response
 - See also UIR., 311
- Units
 - assigning, 53
 - compartmental modeling, 210
 - converting, 55
 - convolution, 301
 - dose, 173
 - file, 434
 - mass, 54
 - noncompartmental analysis, 180
 - options, 31

- prefixes, 54
- summary statistics, 153
- supported unit types, 54
- time, 54
- volume, 54
- Univariate command, 214
- Univariate confidence interval, 503
- UNLOAD, 634
- Unlocking derived data, 36
- Unstructured variance, 358
- Up variable, 120
- Update Reason dialog, 475
- UpErr(), 634
- UPPER, 214, 572
- Use group headers, 122
- Use raw data, 309, 318
- Use sparse sampling, 162
- User models, 219
 - ASCII examples, 228
 - block content, 223
 - commands block, 225
 - creating an ASCII model, 220
 - differential equation block, 227
 - FORTTRAN, 239
 - function block, 227
 - libraries, 219
 - secondary parameters block, 228
 - starting values block, 227
 - structure, 222
 - temporary variables, 226
 - transform block, 224
 - WinNonlin variables, 233
- User name
 - PKS, 456

V

- V, 503
- Valid data, 35
- Validation
 - IVIVC, 302

- Variable names
 - Bioequivalence, 382
 - LinMix, 355
- Variables
 - carry along, 490
 - charts, 116
 - classification, 491
 - function variable, 493
 - group, 119, 494
 - ID, 495
 - independent, 495
 - join, 496
 - model variables, 198
 - model variables for NCA, 163
 - names, 234
 - regressors, 501
 - reserved names, 234
 - sort, 119, 502
 - summary variables, 502
 - temporary, 226
 - weight variable, 504
 - WinNonlin variables, 233
- Variance
 - autoregressive, 358
 - banded no-diagonal, 358
 - Banded Toeplitz, 358
 - banded unstructured, 358
 - components, 358
 - compound symmetry, 358
 - defined, 503
 - heterogeneous autoregressive, 358
 - heterogeneous compound symmetry, 358
 - mean square, 497
 - no-diagonal factor analytic, 358
 - Toeplitz, 358
 - unstructured, 358
- Variance Inflation Factor, 419
 - VIF defined, 503
- Variance structure, 342, 357, 382, 402
 - bioequivalence, 402
 - LinMix, 357

- types, 358
- Visual Basic
 - custom ODBC query example, 111
- Vmax, 503
- Vss, 504
- VZ, 504
- VZ/F, 504

W

- Wagner-Nelson, 280
 - output options, 289
 - weighting, 290
 - workbook output, 290
- Warnings and errors, 641
- Watson DMLIMS, 87, 97
- WCSS, 217
- Weibull model, 553
 - double, 553
- WEIGHT, 634
- Weight
 - variables (bioequivalence), 401
 - variables (LinMix), 356
 - variables, defined, 504
- Weighted least squares, 243
- Weighted summary statistics, 157
- Weighting, 243
 - compartmental modeling, 207
 - in ASCII models, 245
 - iterative reweighting, 244, 495
 - least squares, 177
 - Loo-Reigelman, 290
 - noncompartmental analysis, 177
 - scaling, 244
 - scaling of weights, 207
 - uniform, 177
 - Wagner-Nelson, 290
 - weighted least squares, 243
 - weights from data file, 244
- Westlake confidence interval, 504
- Windows

- hiding, 18
- Windows metafiles, 16
- WindowState, 635
- WinNonlin, 1
 - Enterprise, 1
 - file compatibility with WNM, 17
 - options, 28
 - Professional, 1
 - variables in user models, 233
 - WinNonlin object files, 15
- WinNonlin Script File, 422
- WinNonMix
 - file compatibility with WNL, 17
- WITH, 635
- Word
 - export, 76
 - export options, 77, 78
 - ExportAll command, 595
- WordExportFilename, 635
- Workbooks
 - copying, 14
 - copying with transformations, 64
 - dependencies, 33
 - missing values, 28
 - new, 14
 - operators, 45
 - options, 28
 - out of date, 35
 - page setup, 21
 - Pharsight Workbook Objects, 15
 - pink, 33
 - refreshing, 35
 - saving to Excel, 16
 - show formulas, 28
 - status codes, 69
 - troubleshooting, 16
 - unlocking, 36
- WorkingDir, 635
- Worksheets
 - cell references, 44
 - deleting, 42
 - fill-down, 46

find, 50
formulas, 44
freezing rows and columns, 56
go to, 52
history, 37
inserting, 42
merge cells, 48
merging, 57
naming, 42
replace, 51
scientific notation, 48
sorting, 57
transformations, 62, 64
units, 53
wrap text, 48
Workspaces, 19
 defined, 504
 loading, 20
 new, 20
 saving, 19
 saving dependencies, 35
Wrap text, 48
WRSS, 504
WSSR, 217
WT, 234

X

X, 234
X matrix, 332
XLabelsFont, 636
XLabelsFontSize, 636
XTitle, 636
XTitleFont, 637
XTitleFontSize, 637
XUnits, 637
XVar(), 638

Y

YLabelsFont, 638

YLabelsFontSize, 638
YTitle, 639
YTitleFont, 639
YTitleFontSize, 639
YUnits, 639
YVar, 640

Z

Z array, 234