

Part 3 – User Written PML Model

Units: When using WNL-Phoenix Models, if units are used, the units are required to be entered properly for both concentration and dose. For example, if there are units assigned to observations, but units for dose are missing, then the results may be erroneous; **it is recommended to have dose units set to the same as the mass part of the concentration variable.**

Zero Order 1 Compartmental Model

1. Right-Click on 'Copy of Sheet1' > **Send To>WNL-PHX Modeling>Phoenix Model.**
2. View the Setup tab of the Phoenix Model object.
3. In the Structural tab, select 1 Compartmental Intravenous model with Micro Parameterization.
4. Select 'Main (Copy of Sheet 1)' listed in the Setup tab and map the following variables:
 - Time → Time
 - Cp → Cobs
5. Either map Dose from the previous WNL classic model's "**Dosing used**" or create an internal worksheet with a dose of 20000.
6. select the checkbox for 'Population ?' in the Structural tab.
7. Select Additive for the error model. (analogous to uniform weighting in classical WinNonlin → (Cpred + eps))

The screenshot shows the 'Setup' tab of the Phoenix Model window. The 'Main (Sheet1)' model is selected. The 'Mappings' table shows 'Time' mapped to 'Time' and 'Cp' mapped to 'Cobs'. The 'Structure' tab is active, showing 'Type: PK', 'Parameterization: Micro', 'Absorption: Intravenous', and 'Num Compartments: 1'. The 'Residual Error' section shows 'C' as 'CObs', 'CEps' as 'Additive', and 'Stdev' as '1'. The 'Statements' section contains the following code:

```
deriv(A1 = - Ke * A1)
dosepoint(A1)
C = A1 / V
error(CEps = 1)
observe(CObs = C + CEps)
```

8. Click straight to the button '**Edit as Textual Model >>**'. Click 'Yes' when the DME Confirmation message window appears.
9. Click on Model listed in the Setup tab and edit the Textual Model as **PML Code**, only the two lines in bold need be edited:

```

test(){

  # This is the PK model.
  deriv(A1 = -Ke * A1)
  C = A1 / V

  # This declares that dosing is to compartment A1,
  # and it is treated as zero-order of estimated duration.
  dosepoint(A1, duration = Tabs)

  # An alternative would be to instead model to find the zero-
  order rate

# dosepoint(A1, rate = Zrate)

  # this seems to be more stable, QC 11949, PHX Notification # 13.

  # This is an additive error model

  # Set to 10 as initial estimate of epsilon i.e. 10% of the
  observed conc
  error(CEps=10
  observe(CObs = C+CEps)

  # This is the parameter model.
  # You could incorporate random effects, because they are treated
  # as zero when doing individual modeling.
  stparm(V = (tvV))
  stparm(Ke = (tvKe))
  fixef(tvV = c(, 100, ))
  fixef(tvKe = c(, 1, ))
  fixef(Tabs = c(, 4, ))
}

```

Note on Epsilon initial estimate.

The conc data goes up to 87, so it is unlikely that eps could be as low as 1 (10% residual errors are a good rule of thumb, and it is usually better for the algorithm to guess 'high' on the initial eps rather than be too low, so all data points come in with reasonably similar initial weights.

10. Execute the Phoenix Model Object.
11. Review Output

PLOTS (PRED plots will be similar to IPRED plots since this is individual modeling)

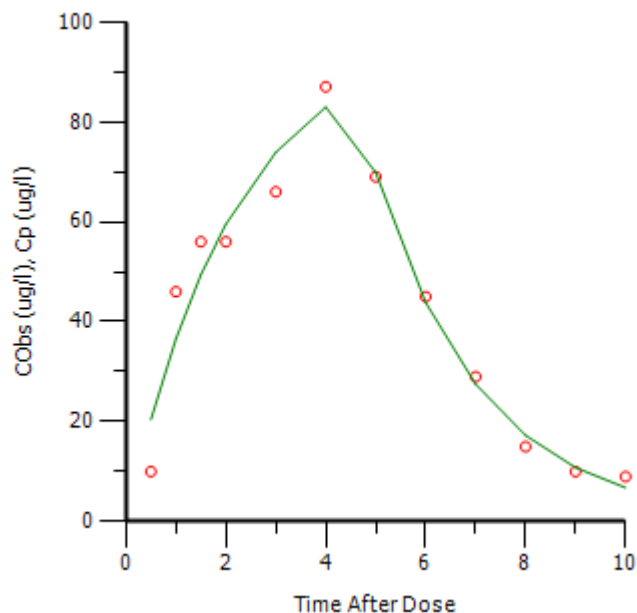
- DV vs. IPRED
- DV, PRED, IPRED vs IVAR
- DV, PRED, IPRED vs. IVAR Lattice
- DV, PRED, IPRED vs. TAD
- DV, PRED, IPRED vs. TAD Lattice
- IWRES vs. IPRED

OUTPUT DATA

- Overall
- Residuals
- Theta
- Theta Covariance

Table statements for predicted curves

12. You may notice that the default PML code gives a 'jagged' point to point curve, this is because plot generated by the PML code only uses the original time points, unlike WinNonlin classic models which predicts over a smooth grid of time automatically. This is because WinNonlin is designed to support only individual modelling whilst Phoenix may also perform NLME i.e. Population analyses. In these analyses individual prediction plots of thousands of points per profile could be prohibitive in terms of performance.



13. To generate a smoother prediction curve in Phoenix you should ask for a table with fine grid time points under **Run Options**, of **seq(0,4,0.1)**, **seq(4.5,12,0.5)**

Population?		General	Parameters	Input Options	Initial Estimates	Run Options	Model Text	Plots	no warnings
Method: Naive-pooled	Stderr: Central Diff	Run Mode		Table01 x <input checked="" type="checkbox"/> SmoothTime					
N Iter: 1000	Confidence Interval %	<input checked="" type="radio"/> Simple		Add Table					
<input checked="" type="checkbox"/> Sort Input?	Alpha: 95	<input type="radio"/> Pred. Check		Structural Parameter Estimates					
Max ODE: matrix exponent		<input type="radio"/> Simulation		Times: seq(0,12,0.01)					
				When covr set:					
				When dose: A1					
				When observe: CObs					
				Variables: C					

The syntax follow S-plus conventions and the **seq** function

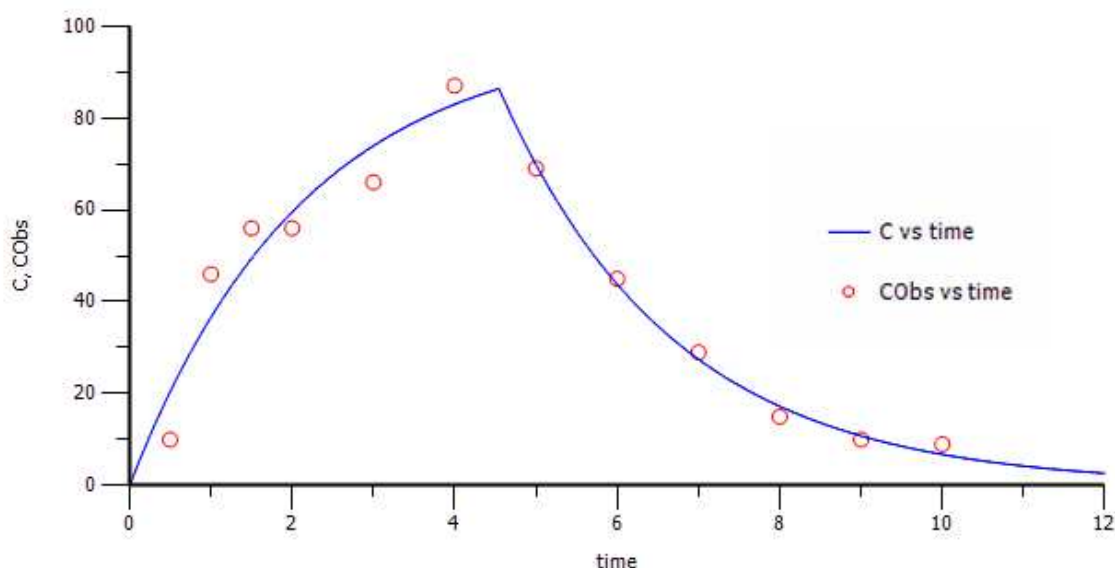
Seq(from=, to=, every)

It also accepts a vector of times concatenated with the **c** function e.g:

c(seq(0.5,2,0.01),seq(3,24,0.05)) or **c(1,2,3,4,5)**

Note that this **Table** is only needed when fitting, with simulation where the WinNonlin Classic input of from to and Npoints still applies.

14. Use this output to generate an additional the plot



OPTIONAL: Notice that with an additive error model, $\text{observe}(\text{CObs} = C + \text{CEps})$, the parameter estimates have better precision (CV%) compared to the results from the ASCII model. Try a different error model, such as the multiplicative error model, **$\text{observe}(\text{CObs} = C * (1 + \text{CEps}))$** , and compare results.

Residual Error:			
C	CObs	CEps	= Mixed <input type="checkbox"/> BQL?
mix Ratio:	1.5	error(CEps = 0.1)	
Stdev:	0.1	observe(CObs = C + CEps * (1 + C * 1.5))	

Parameter	Units	Estimate	StdError	CV%	UnivarCI_Lower	UnivarCI_Upper	PlanarCI_Lower	PlanarCI_Upper
VF		96.069022	10.98297	11.4323	71.223607	120.91444	57.809771	134.32827
TABS		4.5464788	0.23977094	5.27377	4.0040746	5.0888829	3.711235	5.3817225
KE		0.46674425	0.06319870	13.5403	0.32377762	0.60971087	0.24659111	0.68689738

ASCII Model

Parameter	Value	Stderr	CV%	CI Low	CI High
tvV	96.1416	9.542955	9.92593	77.05568	115.22751
tvKe	0.46632	0.054875	11.7677	0.356569	0.5760706
Tabs	4.5454	0.207590	4.56705	4.130218	4.9605819
stdev0	0.00538	0.001098	20.4128	0.003184	0.0075784

Additive Error

Parameter	Value	Stderr	CV%	CI Low	CI High
tvV	105.754	12.55826	11.8749	80.63747	130.87053
tvKe	0.43028	0.040348	9.37719	0.349586	0.5109799
Tabs	4.3317	0.374205	8.63877	3.583288	5.0801117
stdev0	0.19705	0.041758	21.1911	0.113540	0.2805756

**Multiplicative
Error**

Generally, the WNL Classic engine and the Phoenix Model Object Naïve Pooled engine will yield very similar results, that is, when the fits are good (standard errors are small or the confidence intervals around the estimates are narrow). They won't yield precisely the same values, but in our internal testing, they were typically within half of a percent

The PHX naive pooled results are true maximum likelihood estimators, whereas the WNL classic results are based on an iterated weighted least squares algorithm that usually comes close to a maximum likelihood solution when the fits are good, but may be significantly different for poor fits. For simple additive error models the results should be identical but will usually differ slightly when other error models are used but the fits are reasonably good.

Note this is only true if the parameters are not at a bound – if a final parameter value is at or near a bound, the results are no longer maximum likelihood or near maximum likelihood estimators)

If the fits are poor, then the Phoenix maximum likelihood parameters may differ considerably from the WNL Classic iterated weighted least squares parameters.

ASCII

MODEL

remark - define model-specific commands

COMMANDS

NFUNCTIONS 1

NPARAMETERS 3

PNames 'VF', 'Tabs', 'Ke'

PUNIT 'L', 'h', '1/h'

NCON 1

END

remark - define temporary variables

TEMPORARY

T=X

Dose=CON(1)

Finf=Dose/Tabs

END

remark - define algebraic functions

FUNCTION 1

PML

PML Code

```
test() {  
  
  # This is the PK model.  
  deriv(A1 = -Ke * A1)  
  C = A1 / V  
  
  # This declares that dosing is to compartment A1,  
  # and it is treated as zero-order of estimated duration.  
  dosepoint(A1, duration = Tabs)  
  
  # This is the error model.  
  error(CEps=1)  
  observe(CObs = C+CEps)  
  
  # This is the parameter model.  
  # You could incorporate random effects, because they are treated  
  # as zero when doing individual modeling.  
  stparm(V = (tvV))  
  stparm(Ke = (tvKe))  
  fixef(tvV = c(, 100, ))  
  fixef(tvKe = c(, 1, ))  
  fixef(Tabs = c(, 4, ))  
}
```

