

Using SAS/ETS® Software for Analysis of Pharmacokinetic Data

Gary S. Klonicki
SAS Institute Inc.

Abstract

The Econometrics and Time Series module of SAS/ETS® software contains a procedure that can be used to perform analysis on pharmacokinetic data. The MODEL procedure allows you to perform compartmental and noncompartmental analysis on the data. This paper shows how to use PROC MODEL to perform pharmacokinetic analysis.

Introduction

Pharmacokinetics is defined as "the study of the time course of drug absorption, distribution, metabolism, and excretion" (Gibaldi and Perrier 1982). In most cases some type of regression is performed on the time-concentration profile, which defines the concentration level of a given drug dosage in the blood at various sampling intervals. The analysis is either compartmental or noncompartmental.

Compartmental analysis is a modeling technique where different body organs and fluids such as the liver, blood, and urine are mathematically represented as compartments. Rates of drug flow from one compartment to another are expressed as partial differential equations. Often these are described as first order processes when the rate of drug flow is taken to be linearly proportional to the concentration of the drug in the compartment. The solution of such systems of differential equations leads to an expression relating the concentration of the drug in a given body fluid (usually blood plasma or serum) to the time elapsed since the drug was administered. If all of the flows of the drug in the compartmental model are first-order processes, then the concentration of the drug in plasma can be expressed as a weighted sum of exponentials in time since dosing. The parameters of this model can then be estimated using standard nonlinear regression techniques. Once this expression is defined, a number of pharmacokinetic parameters can be derived from the model, which characterize the availability of the drug in the body.

Noncompartmental analysis is a methodology where very few assumptions are made about the relationship between drug concentration and the time since the drug was administered. The

pharmacokinetic parameters of interest are calculated using simple arithmetic techniques. However, it is usual to assume that the final, or terminal elimination of the drug from the blood is a first order process. In this case, concentrations in the terminal portion of the concentration-time curve are described as a single negative exponential in time. Hence, the elimination rate can be estimated as the slope of the linear regression of log concentration on time in this latter part of the curve.

Many researchers use the SAS/STAT® module and the SAS® System DATA step to perform the analysis. However, SAS/ETS software contains the MODEL procedure, which may simplify compartmental and noncompartmental analysis. Through simple programming statements, PROC MODEL offers efficiency and flexibility to those performing pharmacokinetic analyses.

PROC MODEL Overview

The MODEL procedure can be used to perform the following tasks with respect to systems of nonlinear equations:

- define and save models
- analyze the structure
- estimate unknown parameters
- specify multiple starting values for parameters
- test parameters
- restrict parameters
- simulate models
- produce forecasts
- create output data sets.

Because single equations and linear equations are subsets of a nonlinear system, the MODEL procedure can handle a wide range of functional relationships, covering both noncompartmental and compartmental analysis. The programming aspect of PROC MODEL provides a wealth of options not available in other procedures.

The equation(s) and other variables you want to define are written using SAS programming statements. The procedure also supports most of

the programming features of the DATA step. Thus, PROC MODEL combines the strength of the DATA step with a procedure. In addition, PROC MODEL supports some lag and difference functions not supported in the DATA step, making some pharmacokinetic calculations easier.

The MODEL procedure is controlled by FIT and SOLVE statements. The FIT statement is used to estimate the model parameters, while the SOLVE statement is used to simulate or forecast the model. These statements can be used interactively, allowing several FIT and SOLVE statements to be invoked within one PROC MODEL invocation. Different input data sets can be employed by each FIT and SOLVE statement, and different output data sets can be created and used in subsequent FIT and SOLVE statements.

PROC MODEL supports two nonlinear minimization methods: GAUSS (the Gauss-Newton method) and MARQUARDT (the Marquardt-Levenberg method). You can control the number of iterations, the number of subiterations, convergence criteria, and starting values. You can also code the error structure to be used in the minimization process; that is, you can specify the objective function.

In terms of the nonlinear regression, you could use the NLIN procedure instead of PROC MODEL. However, an advantage of PROC MODEL is that you do not have to specify derivatives in the procedure; it calculates them for you. Another advantage of PROC MODEL is that it can handle simultaneous equation estimation, whereas PROC NLIN is designed for single equation estimation.

Other PROC MODEL Statements

In addition to FIT and SOLVE, other statements are used to define the model, specify starting values for parameters, invoke weighted least-squares, and output created variables to a data set. These are given below:

ENDOGENOUS *variable variable ...;*

defines the endogenous, or dependent, variable(s) to the procedure. In this paper the concentration variable is the endogenous variable.

EXOGENOUS

-variable variable ...;

defines the exogenous, or independent, variable(s) to the procedure. In this paper the time variable is the exogenous variable.

PARMS

variable [value] variable [value] ...;

defines the parameters of the model. If a number is placed after the parameter name it is used as a starting value for regression. Default starting values are .0001.

WEIGHT

variable;

defines the variable in the input data set to use as a weight in weighted least-squares.

OUTVARS

variable variable ...;

defines the user-created variables to be output to a given data set.

ID

variable variable ...;

defines the variables in the input data set to be output to a given data set.

BY

variables;

defines the variables in the input data set that group observations for separate analyses.

PROC MODEL supports a wide range of options and statements, and this paper covers only a few of them. For more information on all the options and statements available in the procedure, see the *SAS/ETS User's Guide, Version 6, First Edition*.

Equation Variables

An equation variable is one of several special variables used by PROC MODEL. These variables

can be used in programming statements to create new variables or determine the objective function to be minimized. Two equation variables used in this paper are defined below:

- **RESID.name** is used to define the objective function in the minimization process. Unless otherwise specified, this is the residual of the equation estimated; thus, you are minimizing the error sum of squares.
- **PRED.name** is the predicted value of the endogenous variable.

You can make assignments to these variables or use them to create new variables using programming statements. Some of the examples below use these equation variables in PROC MODEL programming statements.

PROC MODEL Examples

The best way to illustrate the versatile programming features of PROC MODEL is to examine several examples. Comments are included in the code to describe sections of the program. Both noncompartmental and compartmental analyses are performed. The examples show how to perform a regression. In addition, programming statements are used to create variables and define the objective function.

Example One

In this example, noncompartmental analysis is performed on a time-concentration profile by patient identifier. The goal is to fit a linear regression to those data points that define the elimination rate of the drug and to define and calculate other variables from the data. The data must be sorted by time in order to perform the analysis correctly.

The SAS DATA step performs the following actions:

- reads time (TIME) and concentration (CONC) data, as well as the patient identifier (ID), the last time point with a concentration value for a given patient (LASTTIME), the last concentration value obtained for a given patient (LASTCONC), the highest concentration value recorded for a given patient (CMAX), the time where the highest concentration value occurred for a given patient (TMAX), and a weight variable (WEIGHT). CONC represents the concentration of the drug in the plasma sample. TIME represents the plasma sample time, in hours, after the drug was taken. That is, TIME=1.5 means that sample was taken 1.5 hours after the individual was given the drug.
- log transforms the concentration variable. Note: The first two concentration values are zero, and a log transformation will generate missing values. These zero values could have been omitted from the SAS DATA step. In pharmacokinetic analysis, you would like to keep zero values because they do provide information on the time-concentration profile, and may prove useful in comparing one patient with another, or in comparing the same patient with a different drug or dose.

The weight variable is input to determine which observations to use in the regression to estimate the elimination rate constant. See the Appendix for SAS DATA step code that finds *lasttime*, *lastconc*, *cmax*, and *tmax*.

The following SAS statements in PROC MODEL specify the regression model variables, parameters, and group processing:

- **ENDOGENOUS** specifies the endogenous variable, LCONC.
- **EXOGENOUS** specifies the exogenous variable, TIME.
- **PARMS** lists the parameters to be estimated.
- **BY** specifies processing by patient identifier.

Because the regression line to be fit is linear, starting values are not given. Default starting values are 0.0001. For nonlinear models, providing starting values can greatly improve the efficiency of the fitting algorithms and avoid discontinuities. After the variables and parameters are defined, the regression equation, a log transform of an exponential model, is specified.

Using programming statements, a predicted variable is created to transform the predicted concentration value from the regression log form back into the original units. Next IF-THEN/ELSE and DO loops creates the variables AUC (the area-under-the-curve and AUMC (the area-under-the-moment-curve), which enable you to perform separate calculations depending on the value of TMAX. These statements define the linear and logarithmic trapezoidal rules for calculating the area. Usually the linear method is employed up to TMAX, and the logarithmic method is used after TMAX, so these rules are coded. Additionally, other pharmacokinetic parameters are defined in the procedure, and in a subsequent DATA step.

The ZLAG function is used here instead of the LAG function. The ZLAG function returns a zero instead of a missing value if a lag does not exist. The LAG function returns a missing value, and for the area calculations you want to start the accumulation with '0' and not ''.

The FIT statement specifies the regression and directs the procedure to fit the regression on the *lconc* equation. The slash (/) indicates that options follow. The OUT= option includes the results in the output data set, named OUTMOD. The OUTACTUAL option specifies that only the actual values of LCONC are to be included in the output data set, OUTMOD (by default the residuals are also included in the output data set).

The OUTVARS statement specifies the variables created within PROC MODEL are included in the output data set, OUTMOD.

The ID statement specifies the input variables to be included in the output data set, OUTMOD.

Finally, the WEIGHT statement specifies the weighting variable for the weighted least-squares regression. In the example, the weight is 0 or 1, which defines the time-concentration points to use in the regression.

```

/*-- input the data --*/

data sample;
input id time conc lasttime lastconc cmax tmax
weight @@;
lconc=log(conc);
cards;
1 0.0 0.00 8.0 0.54 3.50 4.0 0
1 0.5 0.00 8.0 0.54 3.50 4.0 0
1 1.0 1.15 8.0 0.54 3.50 4.0 0
1 1.5 1.98 8.0 0.54 3.50 4.0 0
1 2.0 2.82 8.0 0.54 3.50 4.0 0
1 3.0 3.43 8.0 0.54 3.50 4.0 0
1 4.0 3.50 8.0 0.54 3.50 4.0 0
1 5.0 2.79 8.0 0.54 3.50 4.0 1
1 6.0 1.64 8.0 0.54 3.50 4.0 1
1 7.0 0.93 8.0 0.54 3.50 4.0 1
1 8.0 0.54 8.0 0.54 3.50 4.0 1
2 0.0 0.00 8.0 0.57 3.53 3.0 0
2 0.5 0.00 8.0 0.57 3.53 3.0 0
2 1.0 1.20 8.0 0.57 3.53 3.0 0
2 1.5 2.01 8.0 0.57 3.53 3.0 0
2 2.0 2.92 8.0 0.57 3.53 3.0 0
2 3.0 3.53 8.0 0.57 3.53 3.0 0
2 4.0 3.51 8.0 0.57 3.53 3.0 0
2 5.0 2.81 8.0 0.57 3.53 3.0 1
2 6.0 1.69 8.0 0.57 3.53 3.0 1
2 7.0 1.00 8.0 0.57 3.53 3.0 1
2 8.0 0.57 8.0 0.57 3.53 3.0 1
;

/*-- print the data set SAMPLE --*/

proc print data=sample noobs;run;

/*-- PROC MODEL code to run regression --*/
/*-- and calculate pharmacokinetic and --*/
/*-- regression parameters --*/
/*-- BY patient identifier ID --*/

proc model data=sample ;
by id;
endogenous lconc;
exogenous time;
parms int k;
lconc = int - k*time;
pred = .;
if weight=1 or time=lasttime then
pred = exp(pred.lconc);
h=time*conc;

/*-- Linear trapezoidal rule --*/

if time le tmax then do ;
auc=zlag(auc)+
(time-zlag(time))*(conc+zlag(conc))/2;
aumc=zlag(aumc)+
(time-zlag(time))*(h+zlag(h))/2;
end ;

```

```

/*-- Log trapezoidal rule --*/
    if time gt tmax then do ;

/*-- Make correction to prevent zero --*/
/*-- divide under log trap method. --*/
/*-- This converts the calculation to a --*/
/*-- linear trap method. --*/

        if h=lag(h) or conc=lag(conc) then do ;
            auc=zlag(auc)+
            (time-lag(time))*(conc+lag(conc))/2;
            aumc=zlag(aumc)+
            (time-lag(time))*(h+lag(h))/2;
            end;
        else do ;

/*-- Log trapezoidal rule --*/
/*-- calculation --*/

            auc=zlag(auc)+
            (time-lag(time))*(lag(conc)-conc)/
            (log(lag(conc)/conc));
            aumc=zlag(aumc)+
            (time-lag(time))*(lag(h)-h)/(log(lag(h)/h));
            end;
        end;

/*-- Half-life calculation --*/

        k_hl=log(2)/k;

/*-- Estimate the equation --*/

        fit lconc /
            out=outmod(drop=_estype_ _type_ _weight_)
            outactual;

/*-- Specify created variables to output --*/

        outvars pred auc aumc k k_hl;

/*-- Specify input data set variables --*/
/*-- to output --*/

        id conc weight tmax cmax;
        weight weight;
    run;
    quit;

/*-- print pharmacokinetic results --*/

    proc print data=outmod noobs uniform;
    run;

```

Example Two

In this example, compartmental analysis is performed on a time-concentration profile. The goal is to fit a single nonlinear equation, which is bi-exponential and has four parameters, to the profile. In pharmacokinetic terms, the model to be fitted is that for IV bolus administration with two-phase elimination. Time and concentration data are entered in a SAS DATA step (data from Kaplan et al. 1972).

The DATA= option of the PROC MODEL statement enables you to specify the data set to use. In this case, the COMPART data set is specified. The ENDOGENOUS and EXOGENOUS statements specify the endogenous and exogenous variables. The PARMS statement specifies the parameters to be estimated and provides starting values because the model is nonlinear. Multiple starting values can be assigned in PROC MODEL using a START= option in the FIT statement, and the procedure evaluates each set of starting values and selects the set that minimizes the objective function. (See the appendix for an example of assigning multiple starting values to parameters.) After specifying the model variables and parameters, the compartmental model is specified. The FIT statement requests the fitting of the model. Finally, because PROC MODEL is an iterative procedure, you use a QUIT statement to end the session.

```

data compart;
input time conc @@;
cards;
0.25 177.3
0.50 175.1
0.75 140.8
1.00 145.2
1.50 110.0
2.00 110.0
3.00 95.9
4.00 84.9
6.00 67.3
12.0 30.8
24.0 13.2
48.0 2.7
;
proc model data=compart;
endogenous conc;
exogenous time;
parms a1 95.0 a2 120.0 theta1 1.5 theta2 0.12;
conc=(a1*exp(-theta1*time))+
      (a2*exp(-theta2*time));
fit conc;
run;
quit;

```

Example Three

Example Three is an extension of Example Two. In this example, it is anticipated that, over time, the variance of the distribution of drug concentration changes. This implies that the variance is not homoscedastic, that is, the variance is not constant at each time point. The nonconstant variance is a violation of a basic assumption of least-squares regression.

It is posited that the variance is directly proportional to predicted concentration. The goal here is to correct the previous example for heteroscedasticity by using weighted least squares,

where the weight is the inverse of predicted concentration. In regression terms, we want to minimize the *weighted* error sum-of-squares, which can be written as

$$\sum(1/\text{predicted concentration}) * (\text{residual} * \text{residual})$$

Since we are weighting by predicted concentration, the WEIGHT statement cannot be used in this example. The WEIGHT statement requires a variable that exists in the input data set, and predicted concentration is created during the regression process. This example shows you how to make a correction for the heteroscedastic variance by weighting with predicted concentration.

The MODEL procedure code defines the endogenous and exogenous variables in the data set and defines the parameters. Because this is a nonlinear regression, starting values are given, then the compartmental model is specified and estimated.

The **resid.conc** assignment in PROC MODEL is made to perform weighted least squares and correct for the heteroskedasticity described above. Note that since you make an assignment to **resid.conc**, not the residual squared, you divide by the square root of **pred.conc**. This approach will not work if **pred.conc** is negative.

```
proc model data=compart;
endogenous conc;
exogenous time;
parms a1 95.0 a2 120.0 theta1 1.5 theta2 0.12;
conc=(a1*exp(-theta1*time))+
(a2*exp(-theta2*time));
resid.conc=resid.conc/(sqrt(pred.conc));
fit conc;
run;
quit;
```

Simulating the Model

The MODEL procedure allows you to simulate, as well as estimate, a given model. To perform simulation, use the SOLVE statement. The advantage of PROC MODEL is that you can follow a FIT statement with a SOLVE statement and use the parameters estimated in the simulation.

Example Four

This is a continuation of Example Three. Here a simulation of the model is performed for data created in the DATA step. An option on the FIT statement, OUTEST=, outputs the parameter estimates to a data set. Then, a SOLVE statement

option, ESTDATA=, employs the estimates for the simulation. Another SOLVE statement option, DATA=, reads in a new data set for the simulation, and the OUT= option outputs the simulated concentration values to a data set. This flexibility allows you to change the input and output data sets at any point of the process--during estimation or simulation.

```
data simdata;
do i=0 to 50 by .25;
time=i;
conc=.;
output;
end;
run;
proc model data=compart;
endo conc; exo time;
parms a1 95.0 a2 120.0 theta1 1.5 theta2 0.12;
conc=(a1*exp(-theta1*time))+
(a2*exp(-theta2*time));
resid.conc=resid.conc/(sqrt(pred.conc));
fit conc/outest=est1;
run;
solve conc/estdata=est1 data=simdata out=outsim;
run;
quit;
proc print data=outsim;
var time conc;
run;
```

Conclusions

SAS/ETS software provides an alternative that may prove more flexible for many types of pharmacokinetic analysis.

The MODEL procedure is a powerful tool for performing noncompartmental and compartmental pharmacokinetic analysis. There are many other useful features of the procedure not discussed in this paper, such as storing and retrieving models, generating additional statistics of fit, and specifying multiple starting values for compartmental modeling. See the *SAS/ETS® User's Guide* for additional information on PROC MODEL.

References

Gibaldi, M., and Perrier, D. (1982), *Pharmacokinetics, Second Edition*, New York, NY: Marcel Dekker Inc.

Kaplan, S.A., Weinfeld, R.E., Abruzzo, C.W., and Lewis, M. (1972), "Pharmacokinetic profile of sulfisoxazole following intravenous, intramuscular, and oral administration in man," *Journal of Pharmaceutical Sciences*, 61, 773-778.

SAS Institute Inc. (1988), *SAS/ETS® User's Guide, Version 6, First Edition*, Cary, NC: SAS Institute Inc.

SAS, SAS/ETS, and SAS/STAT are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries.

Appendix

Calculating Cmax, Tmax, Lasttime, and Lastconc in the DATA Step

In Example One several pharmacokinetic parameters were input directly. You could use the SAS DATA step to find these parameters, as shown below. Martin Michael, SAS Institute Inc., provided this code.

```
data sample (keep=id time conc weight)
  parms (keep=time conc cmax tmax
          rename=(time=lasttime
conc=lastconc));
  infile cards eof=out;
  retain cmax tmax time conc;
  input id time conc weight;
  output sample;
  if conc>cmax then do;
    cmax=conc;
    tmax=time;
  end;
  return;
out:
  output parms;
cards;
1 0 0 0
1 0.5 0 0
1 1 1.15 0
1 1.5 1.98 0
1 2 2.82 0
1 8 0.54 1
run;

data sample;
  set sample;
  if _n_=1 then set parms;
run;

proc print;
run;
```

Specifying Multiple Starting Values

The MODEL procedure enables you to specify more than one starting value for any or all parameters. This can be done with the START= option in the FIT statement. The procedure performs a grid search on the parameter space and selects the combination that minimizes the objective function.

Returning to Example Two, multiple starting values can be written with the following code:

```
proc model data=compart;
  endogenous conc;
  exogenous time;
  parms a1 95.0 a2 120.0 theta1 1.5 theta2 0.12;
  conc=(a1*exp(-theta1*time))+
        (a2*exp(-theta2*time));
  fit conc start=(a1 90 95 100 a2 110 120 130 theta1
1 1.5 2 theta2 .10 .12 .14);
run;
quit;
```

Imposing Bounds on Parameters

The MODEL procedure does not have a BOUNDS statement to restrict the parameter values. However, programming statements can be employed to impose boundary conditions. Mark Little, SAS Institute Inc., provided this code.

```
/*-- implement parameter BOUNDS using --*/
/*-- penalty functions --*/

data nllka;
  input patient time conc @@;
  dose=51500;
  wt=1;
  cards;
2 .5 6.68 2 1 7.5 2 2 6.64 2 4 6.16 2 8 5.23 2 12
4.24 2 18 3.52
2 24 2.59 2 36 1.26 2 48 .77 2 60 .45 2 72 .21
;

proc model data=nllka ;
  endogenous conc;
  exogenous time;

  /*-- define regression function --*/
  conc=a*exp(-k*time)-a*exp(-ka*time);

  /*-- deny infeasible set so --*/
  /*-- we don't step out --*/
  if a le 0 or ka le 0 or ka ge 30 or k le 0
  then stop;

  /*-- add penalty functions to deflect --*/
  /*-- search from boundary --*/
  penalty = p1 / ( a ) /* a > 0 */
            + p2 / ( ka ) /* ka > 0 */
            + p3 / ( 30 - ka ) /* ka < 30 */
            + p4 / ( k ) ; /* k > 0 */

  resid.conc =
    sqrt( resid.conc ** 2 + scale * penalty );

  /*-- set initial parameter guess --*/
  parms a 5 ka 25 k .01;
```

```

/*-- set initial penalty weights --*/
control p1 1 p2 1 p3 1 p4 1 scale 1e-5;

/*-- get initial estimates and print --*/
/*-- results at each iteration --*/
fit conc / method=gauss iter=100 itprint;
run;

/*-- iterate fit shrinking penalty weights --*/
/*-- towards zero --*/
control scale 1e-6;
fit conc / method=gauss itprint;
run;

control scale 1e-8;
fit conc / method=gauss itprint;
run;
quit;

```