NONMEM Users Guide - Part VI

PREDPP Guide

April 2017

by

Stuart L. Beal

Alison J. Boeckmann

Lewis B. Sheiner

NONMEM Project Group
University of California at San Francisco

ICON plc, Gaithersburg, Maryland

Preface to 4th Edition

The appearance of this 4th edition of the PREDPP Guide corresponds to the release of PREDPP 7.4.0 and NONMEM 7.4.0. This is the first new edition since PREDPP Version III and NONMEM IV in 1992.

Significant changes since the previous version are marked with bars.

Versions of PREDPP are now numbered to match those of NONMEM. In the following, new features are arranged by the versions in which they were introduced.

PREDPP IV (NONMEM V)

ADVAN11, ADVAN12, TRANS6

New ADVAN routines. These implement the analytic solution for the Three Compartment Linear Model. New TRANS routine. See Chapter VII, Sections C.11, C.12.

$MODEL

A user-written MODEL subroutine is no longer requrired. NM-TRAN can generate the desired subroutine. See Chapter VI, Section B.

Endogenous drug

A record with AMT=0, RATE=0, SS>0, and II=0 allows steady-state amounts to be computed by PREDPP based on endogenous input given by the DES routine. See Chapter I, Section I.

New value of SS data item: SS=3

With Steady State routines SS6 and SS9 and SS13, the existing state vector (compartment amounts and eta derivatives) may be used as the initial estimate in the computation of the steady-state amounts. See Chapter V, Section F.2.

Models with no Basic PK parameters

With ADVAN routines 6, 8, 9, and 13, MODEL may now define models with no basic PK parameters. See Chapter IV, Section B.

ADVAN9 as an Algebraic Equation Solver

ADVAN9 may be used when the system consists of only algebraic equations (i.e., no differential equations). See Chapter VI Section B.

Changes to DES and AES Routines

Argument DT

With subroutine DES, a new argument DT must be declared. The partial derivative of DADT(i) with respect to T must be stored in DT(i). See Chapter VI, Section C.

New Structure for DA, DP, DT: Full vs. Compact

See Appendix IV.

AES and DES Initialization at ICALL=1

AES and DES may return information to PREDPP at the initialization call. See Chapter VI, Sections C.B and E.B.

Display AES and DES Defined items

DES- and AES-defined items may be displayed in tables or scatterplots and are computed at the event time in the data record. See Chapter VI, Sections C.C and E.C.

Theta and Data record items in DES and AES

In DES and AES: Theta and EVTREC are usable directly. See Chapter VI, Sections C.A and E.A.

PK, ERROR, TRANS: NACTIV,IDXETA

A user written PK, TRANS, or ERROR routine may compute derivatives for all η's, or optionally, to reduce run time, only for active η's. See Chapter III, Section E.5.

ERROR: IDEF

At ICALL=1, a value may optionally be stored in IDEF to reduce run time. It tells if ERROR uses derivatives of state vector A (see below for similar feature of PK). See Chapter IV, Section D.A.

PK: Calling protocol

Substantial time savings are possible when the PK subroutine limits calls to once per individual record (CALLFL=1 or ONCE PER IR). This is especialy important with ADVAN5 and ADVAN7. See Chapter III, Section H.

INFN: PASSRC, NEWND

PASSRC contains the data record at ICALL values 0, 1 and 3. PASSRC changes in conjunction with calls to PASS, at which time transgeneration is permitted. NWIND contains the NEWIND value at ICALL 0, 1 and 3. NWIND changes value in conjunction with calls to PASS. See Chapter VI, Section A.

$BIND

This is an NM-TRAN feature useful for PREDPP. It specifies which values from which event records are used in PK subroutines at non-event times. See Chapter III, Section I.A.

PREDPP V (NONMEM VI)

MTIME

Model Event Time feature. See Chapter III and Appendix III.

A_0, A_0FLG

Compartment Initialization feature. See Chapter III, Section I.B.

I_SS feature

Initial Steady State feature. See Chapter III, Section I.B and Chapter V, Section F.3.

New value of CALL data item

CALL may be 10 to force a call to ADVAN9 See Chapter IV, Section V.J.

New values of CMT data items.

CMT data item values 100 and 1000 may be used to indicate the default compartment for output. See Chapter IV, Section V.H.

PK: State vector

Like ERROR, PK may use elements of the state vector (compartment amounts A) on the right. TSTATE (time of state vector A) may be used on the right. See Chapter III, Section I.A.

PK: IDEF

At ICALL=1, a value may optionally be stored in IDEF to reduce run time. It tells if PK uses derivatives of state vector A. See Chapter III, Section G.

PK, ERROR: New values of ICALL argument

New values 5 and 6 of ICALL indicate that NONMEM is computing raw data averages or expectations. PK and ERROR routines may have to provide code for these calls. See Chapter III, Section C and Chapter IV, Section B.2.

PREDPP7 (NONMEM 7)

ADVAN13

General Nonlinear Model With Stiff/Nonstiff Equations Using LSODA.
See Chapter VII, Section C.13.

ADVAN14

General Nonlinear Model With Stiff/Nonstiff Equations Using CVODES.

See Chapter VII, Section C.14.

ADVAN15
General Nonlinear Model with Equilibrium Compartments Using IDAS.
See Chapter VII, Section C.15.

TOL Changes to SUBROUTINE TOL. See Chapter VI, Section D.B.

New data items XVID XVID1 XVID2 XVID3 XVID4 XVID5
"Repeated Observation Records" feature exists to assist in specialized methodologies such as stochastic differential equations. See Chapter V, Section B.

Larger models
The default numbers of compartments and parameters are larger. Even larger values can be requested using options PC and PG of the $SIZES record. Other options such as MAXFCN affect PREDPP. For example, $SIZES MAXFCN=9000000 may be used with General Non-Linear models (ADVAN6, ADVAN8, ADVAN9, ADVAN13) to request more function evaluations than the default value. See Chapter VII, Section A.

Other changes to the PREDPP Guide itself are as follows.

Appendix III. MTIME Details

This replaces "Appendix III. One-Dimensional Format for IDEF of PK", which is obsolete.

Appendix IV. Compact Arrays in DES

This provides details about DES and the use of the Chain Rule.

All figures have been updated to the NONMEM 7.4 versions. In the figures showing NONMEM control streams (16, 20, 22, 25, 29, 31), certain records have been omitted for simplicity. E.g., LBW1 and BEST and others that are only used only with Bayesian methods. These are optional records, and their omission does not affect a NONMEM run in which the Bayesian method is not used. Note that the FCON control stream in Appendix II is complete.

Figure 17a is new.

Table of Contents

I. Overview of PREDPP

PRED is a NONMEM subroutine that needs to be supplied by the user. Its basic function is to compute model-based predictions for observations. A PRED for population pharmacokinetic data analysis can be very complicated and difficult to write, and the NONMEM Project Group has tried to help by writing PREDPP, a general version of this subroutine that is particularly suited for the analysis of population pharmacokinetic data. This version of PRED is informally called PREDPP -- for **PRED P**opulation **P**harmacokinetics, but the formal name of the subroutine is simply PRED, since this is the entry name used by the calling programs in NONMEM. PREDPP can also be used with NONMEM to analyze pharmacokinetic data from a single subject, because the required predictions are of a similar nature, and it can be used with NONMEM to *simulate* either population or single-subject pharmacokinetic data. As the reader shall see, PREDPP is a very powerful tool for computing pharmacokinetics.

There are a number of different basic kinetic models that can be invoked by PREDPP. The user selects the basic model by selecting one or more subroutines from a library of subroutines of PREDPP and linking the selected subroutines with the NONMEM and PREDPP subroutines that must always be present in a load module.† This process is described in detail in chapter VII. The library is called the PREDPP Library . For example: If the user is interested in a one compartment linear model, he chooses subroutine ADVAN1 from a list of PREDPP Library subroutines: ADVAN1, ADVAN2, ADVAN3, etc., which, though distinguishable by the different models they compute and by their informal names as just stated, actually have the same formal name, ADVAN, since this is the entry name used by the calling program in PREDPP. ADVAN stands for ADVANCE because it is the task of each ADVAN routine to advance the state vector of compartment amounts (and partial derivatives of the compartment amounts with respect to random variables, when appropriate) from each point in time to the next. For all the ADVAN subroutines, PREDPP does the housekeeping for the state vector. For example PREDPP initializes compartments to zero at the start of an individual record, resets compartments to zero when requested, applies bolus doses, turns compartments on and off, keeps track of event and non-event times such as infusion start and stop times, additional dose times, and model event times, and computes amounts in the the output compartment if it is on. Another possible name for ADVAN would have been SOLVER, because most ADVAN routines solve a set of differential equations, either analytically (ADVAN1-5, ADVAN7, ADVAN10, ADVAN11, ADVAN12) or by integration (ADVAN6, ADVAN8, ADVAN13, ADVAN14). ADVAN9 and ADVAN15 solve a set of simultaneous differential and algebraic equations.

The kinetic models that may be selected are:

- The one compartment linear model [ADVAN1]
- The one compartment linear model with first order absorption from a drug depot compartment [ADVAN2]
- The two compartment linear mammillary model [ADVAN3]
- The two compartment linear mammillary model with first order absorption from a drug depot compartment [ADVAN4]
- One of several general linear compartmental models [ADVAN5,7]
- The general nonlinear compartmental models [ADVAN6,8,13,14]
- The general nonlinear compartmental model with equilibrium compartments [ADVAN9,15]
- The one compartment model with Michaelis-Menten elimination [ADVAN10]
- The three compartment linear mammillary model [ADVAN11]
- The three compartment linear model mammillary model with first order absorption from a drug depot compartment [ADVAN12]

† The NONMEM load module is also referred to as the NONMEM executable.

As will be discussed momentarily, this list covers a good deal more than appears to be covered at first glance. First, though, the distinction between a general linear and a general nonlinear compartmental model is clarified. A general linear compartmental model describes linear kinetics with up to 999 compartments, and the computational algorithm employed is based on use of the matrix exponential. Linear systems with only real eigenvalues, or with possibly complex eigenvalues, are reliably handled. A general nonlinear model describes either linear or nonlinear kinetics with up to 999 compartments, and the computational algorithm employed is based on a differential equation solver.

Most often with population pharmacokinetic data analysis, a one or two compartment model suffices. Although these models can also be implemented with a general linear or nonlinear model, the specific implementations that have been written for PREDPP are based on the traditional sums of exponential forms, and are therefore computationally very much faster than are implementations with the general models.

With any of the models, a peripheral output compartment is always present. Associated with this compartment is a dimensionless pharmacokinetic parameter, F_o. Of the entire amount, A_o, of drug introduced into the system and eliminated from it, a fraction F_o of this amount goes into this output compartment. The output compartment may be turned on and off; while on, drug accumulates therein, and when turned off, the amount therein is reset to zero. So, for example, if the output compartment is regarded as a urine compartment (so F_o might be the ratio of renal to total clearance), the initiation and termination of a urine collection can be simulated. Alternatively, using general linear or nonlinear models, compartments from which no distribution can occur into any other compartments can be defined within the system itself. Such compartments may also be turned on and off. If a user-defined comparment is initally off, may be turned on and off, and may not receive a dose, it is called an output-type compartment.†

The kinetics, i.e. drug amounts, described by any of these models can be parameterized by either microconstants, i.e. rate constants, or certain reparameterizations of the microconstants, possibly involving other parameters such as volume of distribution. The value of a kinetic parameter can depend in turn on the values of physiological and other concomitant variables. For any individual, the values of these concomitant variables can change at discrete times, and so, therefore, can the value of the PK parameter.

Drug can be input into any of the compartments (except the output compartment) at any time. Bolus doses can be given. In the case of the one compartment model with first order absorption, for example, an IV bolus dose can be input into the central compartment, and another oral bolus dose can be input into the drug depot compartment at the same or different time as the first dose is input. Also, infusions of known amount and duration can be given. Also, a bolus dose of a known amount can be given whose appearance into a specified dose compartment, rather than being instantaneous, is actually governed by a zero-order process, the rate or time-duration of which may be regarded as a kinetic parameter whose value may be specified or estimated. So for example, a bolus dose in the form of a tablet may be placed in a compartment, but due to the dissolution process, its appearance in the compartment may be better described by a zero-order process.

If many doses are given over time, as may happen in a clinical setting, the specification of this in the data record structure can be tedious, and the computation time is affected since a state vector is updated at each time point that a dose is given. If, however, these doses are given in a regular cyclic fashion (e.g. 5 mg at 8 AM, 1 mg at 8 PM; daily) and over such a sufficiently long time that the system can be regarded as being at steady-state at the times the doses of some particular cycle, i.e. the "steady-state cycle", are given, then steady-state kinetics can be used to greatly mitigate the dose specification and computation time problems. Only the steady-state cycle of doses need be specified, and the state vector will only need to be updated at the times these doses are given. Steady-state kinetics may be invoked for this purpose with any of the PREDPP models (except with a general nonlinear compartmental model when the nonlinear kinetics imply that steady-state cannot be achieved).

---

† Output-type compartments have been part of PREDPP since the first version, but were not discussed in detail. See Chapter V Section H and VI Section B.

If several doses are given over time, but they do not lead to a steady-state, the specification of this in the data record structure might still be tedious. However, if these doses are the same type of dose, given with a constant interdose interval, then in this case there is still a convenient way to mitgate the specification problem.

All the kinetic models are augmented by the ability to allow any dose to appear in the system at some lag time (called an absorption lag time) after the time the dose is nominally given. All kinetic models are augmented by the ability to allow a fraction (called a bioavailability fraction) of any dose to appear in the system, rather than the nominal dose. Absorption lag times and bioavailability fractions may be specified or estimated.

All the kinetic models allow model event time parameters, which allow the modelling of discrete times at which some aspect of the system can be changed. Model event times may be specified or estimated.

All the kinetic models allow the use to supply initial values for the compartments, using the compartment initialization feature (See A_0, Chapter III, Section I.B).

With general non-linear models (ADVAN6, ADVAN8, ADVAN9, ADVAN13, ADVAN14, ADVAN15) the differential equations may include terms for endogenous drug production. The Initial Steady State feature of PREDPP may be used to compute steady state compartment amounts without using dose event records. (See I_SS, Chapter III, Section I.B and Chapter V, Section F.3). This feature may be used with the compartment initialization feature.

Not only must PRED compute model based predictions, it must also compute partial derivatives of the statistical model for the observations with respect to all random variables occurring in the model (see Guide I. See also the Introduction to NONMEM 7. Classical NONMEM methods require the partial derivatives, but new methods may not). This computation has proved to be a particularly difficult one for many users to program. Just as PREDPP serves the important purpose of greatly simplifying the programming the user must undertake to compute complicated population-type pharmacokinetic predictions, it also serves the important purpose of greatly simplifying the user-required programming of the derivative computation for population pharmacokinetic models.

The user must still be concerned with generating a small amount of code in the form of two user-supplied subroutines of PREDPP. The first routine, PK, functions essentially to compute the values of pharmacokinetic parameters, e.g. clearance, in terms of the values of covariables, e.g. age and weight, and the values of random interindividual effects accounting for random interindividual variability. The second routine, ERROR, functions essentially to specify the residual error structure, e.g. to specify that residual error is additive, or that it is multiplicative. However, PREDPP allows pharmacodynamics along with pharmacokinetics to be modeled, and it does so by allowing pharmacodynamic models to be specified in the ERROR routine. This may strike the user as a strange design, which it is. The design for PREDPP allows pharmacokinetic models to be specified conveniently, and only as an "afterthought" does it allow the specification of pharmacodynamic models. Detailed discussion of PK and ERROR is to be found in chapters III and IV. A few other user-supplied subroutines are at times required. Detailed discussion of these is to be found in chapter VI. In particular, use of a general nonlinear compartmental model requires the user to supply code for the involved differential equations, but as seen in section VI.C, this need not be a hard task. With NM-TRAN, which is a pre-processor to NONMEM and PREDPP, the task of creating PK and ERROR routines is simplified, and the user need not write the code to compute partial derivatives of the statistical model for the observations with respect to the random variables. $PK and $ERROR blocks describe the statistical model. NM-TRAN supplies the code to compute the partial derivatives.

II. Introduction to the Event Record

By default, NONMEM calls PRED with one data record after another, no data records being skipped, and no record being repeated. NONMEM can be requested to call PRED with a different sequence of records, using the Repetition feature of NONMEM. A reserved data item RPT_ may be used, or reserved variables RPTO, RPTON, PRDFL may be set in PK or ERROR (see chapter III Section I.B). One use of this feature is for models involving convolution. The discussion below is unaffected by this use of feature.

With NONMEM each observation (i.e. each value of the dependent variable of the regression model) is placed on a separate data record called an observation record , and the value placed on the record is called the dependent variable (DV) data item.

There are situations where it is convenient to place all the values of the concomitant variables (i.e. independent variables) needed to compute the prediction of an observation on the same data record as that one containing the observation. Then all data records are observation records. There are other situations, though, where it is convenient to organize the concomitant variables across different data records. One situation arises when one needs to avoid the NONMEM restriction on the number of data items per data record. (The maximum number of data items per data record is given by constant PD in resource/SIZES.f90. The default value is 50. A larger value of PD may be specified using the NM-TRAN $SIZES record.) Another situation arises with the following example. Suppose there is a chronology to events: at time 4 days a dose of 5 mg is given, at time 5 days a dose of 10 mg is given, and at 7 days a plasma concentration is observed, etc. Then data may be originally recorded in the laboratory or clinic so that all the data items associated with an event occuring at given time are contained on one record, with different records for events at different times. Moreover, these records may be time ordered. When NONMEM data records must be constructed, it may be more convenient to construct records that look like these. NONMEM allows this. A NONMEM data record need not be an observation record, i.e. it need not contain an actual observation. If there are nonobservation data records, each record must contain a missing dependent variable (MDV) data item, whose value is 0 if the record is an observation record, and 1 if the record is not an observation record. The DV data item is ignored when the MDV data item is 1. It need simply be a null value which serves as a place holder for the position where an observation would be placed on an observation record. With NONMEM 7.3, values of MDV may be 100 or 101. MDV=100 is the same as MDV=0 but tells NONMEM to ignore the record during the Estimation and Covariance Steps. MDV=101 is the same as MDV=1 but tells NONMEM to ignore the record during the Estimation and Covariance Steps.

PREDPP is concerned with time ordered events: dose events, observation events, and other events such as the beginning of a urine collection and a measured change in kidney function. This routine *expects* that there will be different data records associated with each of the different events, and that these data records are time ordered. Data records are also organized into individual records . In the case of population data an individual record is a group of contiguous data records all associated with the same individual and having the same identification (ID) data item. In this case the time ordering of data records expected by PREDPP refers to the time ordering of data records *within* an individual record. In the case of data from a single subject individual records do not correspond on a one-to-one basis to either animal or human subjects. An individual record is simply any group of contiguous data records containing only one observation record and having the same ID data item. (Although, the presence of a multivariate observation means the individual record containing it has several observation records, each containing one element of the observation.) In this case the time ordering of data records expected by PREDPP refers to the time ordering of data records *across all* records.

To summarize the foregoing, a typical NONMEM data set for PREDPP (the first 9 records of which) can be illustrated thusly:

ID	TIME	C1	C2	...	CN	DV	MDV
1	1.	.5	10		5		1
1	3.		9		10		1
1	6.		6			4.1	0
1	7.		7		10		1
1	10.		7			5.3	0
2	2.	.9	8		8		1
2	2.5		6				1
2	6.		7				1
2	7.		8			6.4	0

where ID, MDV, and DV label the ID, MDV, and DV data items, where TIME labels the time data items, and where C1, C2, ..., CN label concomitant variable data items. NONMEM itself, and not PREDPP, recognizes and responds to ID, DV, and MDV data items. With PREDPP there must always be time data items giving the times of the events. (An exception occurs with ADVAN9 when there are only algebraic equations to be solved, in which case the time data item is optional.) PREDPP, and not NONMEM, recognizes and responds to TIME data items. Note that some data items are left blank. In a NONMEM data set blanks are null values and will be read by the program as zeros, and displayed as such in tables and scatterplots. (In an NM-TRAN data set blanks serve as a place holder for a null value when a format is given. When a format is not given, a dot (' . ') or a zero (' 0 ') can be used as a place holder.) Neither NONMEM nor PREDPP supplies nonzero values for null values, or understands these null values as "missing values". In this illustration there are 2 individual records.

In addition to the time data item, each event record must have a data item indicating the type of event in question, and if, for example, the event is a dose-type event, the event record must have additional data items specifying certain dose type information. This is all discussed in Chapter V. Usually, all drug dosing information is specified in the event records, rather than in user-supplied code. In the illustration, the event identification data items are not shown, but the amount data items are labeled CN, though ordinarily, they might be labeled AMT; see section V.D.

In this illustration there are 9 data records corresponding to 5 different time ordered events within the first individual record and 4 different time ordered events within the second individual record. Three of these events are observation events: two within the first individual record, and one within the second individual record. At least four of these events are dose events (one cannot identify dose events simply by the presence of nonzero amount data items): three within the first individual record, and one within the second individual record. Typically, as in the illustration, with PREDPP each data record is also an <u>event record</u>, describing a separate event.

However, infrequently, several data records may be needed to describe an event.† One can group a number of data records associated with a given event into a single event record. This is accomplished by including in each data record a data item indicating whether or not the record and the succeeding record are both members of the same event record. This data item is called the <u>continuation data item</u> (CONT data item). Its value is 1 if the data record and the succeeding record are both members of the same event record, and 0 otherwise (or if the data record is the last data record). It is PREDPP that recognizes and responds to CONT data items, and the concept of an event record is peculiar to PREDPP, not NONMEM. If continuation data items are not used, each data record constitutes a separate event record.

A slightly different example of a NONMEM data set is this:

† The CONT data item described in this paragraph cannot be used with NM-TRAN abbreviated code.

```
ID      TIME      CONT     C1  ...  CN       DV       MDV
 1                  1      .5                          1
 1        1.        0                5                 1
 1        6.        0                       4.1        0
 1        7.        0               10                 1
 1       10.        0                       5.3        0
 2                  1      .9                          1
 2                  1                                  1
 2        6.        0                8                 1
 2        7.        0                       6.4        0
```

Here the CONT data items appear. There are 6 event records, 4 in the first individual record and 2 in the second individual record. Both the 1st and 3rd event records contain information needed to predict the second observation (at time 10 hr). At time 1 hr a dose of 5 mg is given, and at 7 hr a dose of 10 mg is given. A plasma concentration is measured at 6 hr. The time associated with each event must appear on the last data record of the corresponding event record; it need not appear elsewhere in the event record.

III. The PK and TRANS routines

A. Introduction

User-supplied code is not needed to define the relationship between the pharmacokinetic parameters and the drug amounts in the various compartments (except when using a general nonlinear compartmental model; see section VI.C). This relationship is referred to as the kinetic relationship , or the kinetics. As described in chapter I, these relationships are already coded into subroutines in the PREDPP Library. If, for example, a one-compartment linear model is to be used, ADVAN1 is chosen (see Chapter I). This subroutine computes drug amounts using, basically, the familiar monoexponential formula. However, user-supplied code for computing the values of the pharmacokinetic parameters themselves is needed. This code comprises a user-supplied subroutine called PK. This chapter is primarily concerned with a description of this routine.

To develop PK the user must first choose a set of pharmacokinetic parameters with which to describe the kinetics implemented by the chosen ADVAN routine. The kinetics can usually be described by several different sets of parameters. Having selected ADVAN1, for example, a user may choose to describe the kinetics in terms of the rate constant of elimination Ke, or he may prefer to describe them in terms of clearance and volume of distribution, Cl, and Vd. (In the first case Vd has to be modeled along with Ke if plasma concentrations are observed - but in order to scale drug amounts, not to compute them.) For each ADVAN, there exists a group of TRANS subroutines, TRANS1, TRANS2, TRANS3, etc., in the PREDPP Library. If Ke is chosen, the user chooses subroutine TRANS1 from the group for ADVAN1. If Cl and Vd are chosen, he chooses TRANS2. Each of these subroutines has the same formal name, TRANS, since this is the entry name that the calling program in PREDPP uses. The TRANS subroutine makes the translation between user-chosen pharmacokinetic parameters computed in PK and set of parameters used internally in the ADVAN subroutine. When, for example, an ADVAN is chosen which implements a linear kinetic model, the internal parameters are the rate constants, i.e. the microconstants, of the model. (With such an ADVAN, TRANS1 is a "dummy" translator that allows the user to compute the rate constants directly in PK.) The user could prefer to compute parameters for which no suitable translator is included in the Library, in which case he can either (i) include code in PK for the parameters he prefers, *followed by* code that performs the translation itself, and then use the dummy translator, or (ii) include code in PK for the parameters he prefers and then also supply his own TRANS subroutine. The requirements for supplying a user-written TRANS routine are addressed in section M.

Pharmacokinetic parameters are subject to interindividual variability, which must be taken into account by an appropriate statistical model. A more precise description of the PK routine is that it defines a statistical model for the PK parameters. A description of this model, and how it can be implemented by the PK routine, comprise the other sections of this chapter. Variability in the pharmacokinetic parameters that can be accounted for solely in terms of concomitant variables is addressed first in sections B and C. Unexplained variability that must be accounted for in terms of random individual effects is addressed second in sections D and E.

B. Modeling Typical Values of Pharmacokinetic Parameters

Values of pharmacokinetic (PK) parameters vary between individuals. This variability may be partially explained in terms of concomitant variables whose values vary between individuals. (The terms 'independent variable' and 'covariable' are also sometimes used.) The values of these variables may also vary within individuals over time; this particular situation is discussed in section B.2. Models for PK parameters that explain interindividual variability only in terms of concomitant variables are incomplete. Usually, there is evidence of variability in the PK parameters between individuals with the same set of values for the concomitant variables. This variability, unexplained by the concomitant variables, often appears as unexplained random variability, and may be modeled in terms of random individual effects, as discussed in section D below. As a result, the values of a PK parameter, between individuals with the same set, x, of values for the concomitant variables, will vary according to a probability distribution having a typical value (e.g. mean or geometric mean) depending on x. This value shall be called the typical value of the PK parameter for individuals whose concomitant values are those given by x. In section B.1 models for the relationship between the typical value of a PK parameter and the elements of x are discussed. (These models only involve the values of the concomitant variables.) An individual's specific value of a PK parameter, in contrast to the typical value (for individuals with the same concomitant values), is called the subject-specific value . Models for subject-specific values are described in section D.

When all the data are from a single subject, this subject is regarded in isolation from other subjects, and the data are not what are commonly referred to as "population data". However, as a matter of NONMEM terminological convention, in this case the subject-specific value is also referred to as the typical value. The models described in this section for the typical value can be used as models for the subject-specific value (except that when there is more than one concomitant variable, there may be identifiabilty problems). Moreover, sections D and E are not applicable.

The issue of time-varying concomitant variables is discussed in section B.2. However, several important general concepts are also discussed in that section: state time, state-time interval, and continuous and discrete action of PK parameters.

B.1 Time-Invariant Concomitant Variables

The simplest model for the typical value $\tilde{P}$ of a PK parameter P is

$$\tilde{P} = \theta_1 \tag{1}$$

Namely, $\tilde{P}$ is a constant, independent of x. Another model might be

$$\tilde{P} = \theta_1 WT \tag{2}$$

where WT is an individual's weight. Here θ_1 is a proportionality constant. Model (2) might be used for $\tilde{Vd}$. In both (1) and (2), θ_1 is a parameter which may be estimated by NONMEM.

In order to model $\tilde{P}$ it is helpful first, to model physiological variables in terms of x, and second, to model $\tilde{P}$ in terms of these physiological variables. For example, let SIZE be a measure of body size given by

$$SIZE = HT^{\theta_2} WT^{\theta_3} \tag{3}$$

where HT is an individual's height. Then, perhaps, let $\tilde{Vd}$ be given by

$$\tilde{Vd} = \theta_1 SIZE \tag{4}$$

(If the data are from a single subject, and HT and WT are not in fact time-varing, then θ_1, θ_2, and θ_2 are not all identifiable.) The physiological variable SIZE may be used also with models for the typical values of other PK parameters, e.g. metabolic clearance

$$\tilde{Cl}_{met} = \theta_4 SIZE \tag{5}$$

For another example, glomerular filtration rate may be modeled

$$GFR = WT(1.66 - 0.01\ AGE)/SCR \tag{6}$$

where AGE and SCR are an individual's age and serum creatinine measurement. Then the typical value of renal clearance may be given by

$$\tilde{Cl}_{ren} = \theta_5 GFR \tag{7}$$

The typical value of total clearance could be given by

$$\tilde{\mathrm{Cl}} = \tilde{\mathrm{Cl}}_{met} + \tilde{\mathrm{Cl}}_{ren} = \theta_4 \mathrm{SIZE} + \theta_5 \mathrm{GFR} \tag{8}$$

The model for $\tilde{P}$ in terms of physiological variables is often linear in the θ's, as these examples illustrate. The model for $\tilde{P}$ in terms of x is often nonlinear in the θ's, as indicated by (3) and (4) taken together.

B.2 Time-Varying Concomitant Variables

The discussion and examples in section B.1 apply when, for each individual, each concomitant variable has a single value. Essentially, the PK routine is called, and the typical value of a PK parameter is computed, using a model such as any of those described in section B.1. However, to a limited extent PREDPP also accommodates the case where the value x of the vector of concomitant variables varies within an individual over time. Again, the discussion and examples in section B.1 can apply, as is now described.

Note that a model for the typical value of a PK parameter simply produces different typical values as x varies. Similarly, as x varies, the subject-specific value of the parameter (for a fixed value of η; see the discussion in section D) also varies. The value x can vary from event record to event record (within an individual record), and if the typical/subject-specific value is computed with each event record, this allows the variation in the typical/subject-specific value, across the time domain during which observations are obtained, to be taken into account, at least to within the time-resolution given by the event times. To properly account for this variation, a fine degree of time-resolution may be required. Event records can be included in the individual record whose sole purpose is to give values x at times of greater resolution (see section V.B). If though, the concomitant variables are only measured at certain discrete times, interpolated values may need to be obtained for these "extra" event records. While the interpolation per se can be implemented within the NONMEM run (see section VI.A), the user must still include extra event records in the data set which contain the extra times. Also, PREDPP itself does not compute the interpolated values, rather this computation must be completely specified with user-supplied FORTRAN code.

The typical/subject-specific value can indeed be computed with each event record, or with a more limited set of event records if desired (see section H). It can even be computed with each event record *and* at certain additional times, allowing for just a bit more flexibility in obtaining interpolated values of the concomitant variables (see below and section H). How these computed values are used in the kinetic computations is outlined next.

The time domain is discretized at the event times, and at some other points as well. These times are called <u>state times</u>, and the time interval between two successive state times is called a <u>state-time interval</u>. The pharmacokinetic system, i.e. the state vector of compartment amounts, is advanced from one state time to the next, and the (typical and subject-specific) values of the PK parameters are assumed to be constant over each state-time interval (possibly different constants over each interval). As the system is advanced, the routine PK is called at various state times. When the system is advanced over the state-time interval (t_1, t_2), the PK routine will have already been called in order to obtain the typical/subject-specific values of all the PK parameters governing the kinetics over the interval. A more precise description is given next.

A state time may be an event time, but there are other discrete times to which the system must be advanced, which are not (formally) event times. For example, an infusion may terminate at some time t, but while an infusion termination is not signalled by an event record, the system state changes in a discontinuous way at t. If time t is also an event time, it is only coincidental. Another example of a nonevent state time occurs when an absorption lag time is computed with a dose; the time the dose actually enters the system is a state time. This state time - indeed, any nonevent state time when either a bolus dose actually enters the system or when an infusion actually begins - is called a <u>nonevent dose time</u>. Of course, if the lag time is computed to be 0, then just coincidentally, the nonevent dose time is an event time, i.e. the time the dose was given. With any state time t there are associated one, or possibly two, particular event records. The first record is the one with event time t if t is itself an event time, or it is the first record whose event time follows t if t is not an event time. It is called the argument record associated with t, for a reason described in section C. If t is a nonevent dose time, then the event record describing the dose is also associated with t.

Certain PK parameters such as clearance act continuously over state-time intervals in the sense that drug amount in the system varies over such an interval (t_1, t_2) according to a pharmacokinetic model which depends on values of these parameters at each instant in the interval. However, PREDPP assumes that the (typical and subject-specific) values of continuously acting PK parameters are constant over the interval, and it obtains these constant values from a call to PK where the argument record associated with t_2 is made available to the routine. The values of the concomitant variables on this argument record determine the constant values of the PK parameters holding over the interval (unless the PK routine is written in such a way as to make use of information made available to it from previous calls).

This is sometimes described as "LOCB Last Observation Feed Backwards" rather than "LOCF Last Observation Feed Forwards". Values computed by PK for the record with TIME=t_2 are used during the advance from t_1 to t_2. If the values on the record with t_1 were used, the values (other than TIME) recorded on the last record in the data set would never be seen by PK, and could not enter into the model. Linear interpolation could not be carried out properly. As it is, the first data record is always seen by PK (because there is always a call to PK with the first data record of the individual record), and all subsequent data records are seen prior to the advance to those records.

Other PK parameters such as a bioavailability fraction (see section F.2) act discretely at state times in the sense that drug amounts in the system vary from one state time to the next according to a pharmacokinetic model that depends on values of these parameters only at these times, although values of particular parameters are only needed at certain state times. In the case of a bioavailability fraction, for example (see section F.2), the model depends on the value of this parameter only at state times when doses enter (or start to enter) the system. For a nonevent dose time t, PREDPP normally obtains the values of these parameters from a call to PK with the argument record associated with t, and if requested, the event record describing the dose is also made available with this call. Information from one or both records may be needed to compute the values of a PK parameter such as bioavailability. For any other state time t, including all event times, PREDPP obtains the values from a call to PK with the argument record associated with t.

As concomitant values change across time, so does the information on event records, and then so does the output of the PK routine, i.e. the values of the kinetic parameters.

C. Implementing Models for Typical Values in PK

PK is a required user-supplied subroutine. Its first several statements, i.e. its preface , must be

```
SUBROUTINE PK(ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,IRGG,GG,NETAS)
USE SIZES,     ONLY: DPSIZE,ISIZE
USE PRDIMS,    ONLY: GPKD
IMPLICIT REAL(KIND=DPSIZE) (A-Z)
REAL(KIND=DPSIZE) :: EVTREC
INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS,IRGG,NETAS
DIMENSION :: IDEF(7,*),THETA(*),EVTREC(IREV,*),INDXS(*),GG(IRGG,GPKD+1,*)
```

This is the NONMEM 7 version. The preface was different with earlier versions of NONMEM. Global variable GPKD is needed because GG is sized according to the number of etas in the problem. GG may be declared as a 2 dimensional array `GG(IRGG,*)` when the Laplacian method is not used. For simplicity, it will be used that way in the examples below, although, when GG is declared 3 dimensional, the third subscript should be understood to be ",1"; E.g., GG(M,1) should be understood to be GG(M,1,1). However, when the Laplacian method might be used later with the given data set, it is a good idea to develop a PK code that allows this.

When PK is called by PREDPP, it is passed values for the vector θ in THETA. It is also passed a complete event record in EVTREC. Specifically, EVTREC(I,J) contains the Jth data item of the Ith data record of the event record. This record is the argument record defined in the previous section. Its name refers to the fact that it is passed to PK as a subroutine argument, EVTREC. (As mentioned in section B.2, there are circumstances where a dose record, different from the argument record, may also be needed by the PK routine. A description of how PK has access to this record is given in section I.) PK is also passed the total number N of data records comprising the event record. Typically N=1, and so the first subscript of EVTREC will always be 1; however, see chapter II. With NM-TRAN, the CONT data item cannot be used and N is 1.

With these arguments the typical values of the PK parameters may be computed. E.g. Let EVTREC(1,1) and EVTREC(1,2) be height and weight, respectively. If $\tilde{V}$d is given by (4) (of the previous section), then one might use the code

```
SIZE = EVTREC(1,1)**THETA(2)*EVTREC(1,2)**THETA(3)
TVVD = THETA(1)*SIZE
```

This typical value of Vd will apply over any state-time interval (t_1, t_2) where t_2 is a state time with which the argument record is associated. When using the first-order method of estimation, this typical value must be communicated to PREDPP, as must the typical values of all the PK parameters; the way to do this is discussed shortly. (When conditional estimates are used, or simulation with population data is implemented, subject-specific values must be communicated instead; see section E.)

The one-dimensional array, INDXS, functions in a way similar to that of a larger array of the same name, described in Guide I, section C.4.1.† In fact, INDXS is comprised of elements 12-50 of the larger array. The user places integers into that array, using the NONMEM control record INDEX (NM-TRAN control record $INDEX). These integers are then available to PREDPP and therefore to PK. The code.

```
I11 = INDXS(1)
I12 = INDXS(2)
I13 = INDXS(3)
SIZE = EVTREC(I11,I12)**THETA(2)*EVTREC(I11,I13)**THETA(3)
TVVD = THETA(1)*SIZE
```

has the same effect as has the previous code when INDXS(1), INDXS(2), and INDXS(3) are 1, 1, and 2, respectively. However, this code, unlike the previous code, frees the user from having to decide at the time PK is coded how the data items are going to be organized in the event record. PREDPP itself makes use of certain integers it requires be placed in elements 1-11 of the larger INDXS array (see section V.A), but it insures that INDXS(1), ..., INDXS(39), *as made available to PK*, refer to elements 12-50 of the larger array. So, the values 1, 1 and 2 of the example actually would be placed in elements 12-14 of that array.

With every translator routine, TRANS, there is associated a particular list of basic PK parameters whose values must be computed by PK, and a numbering of these parameters; see section VII.C. The parameters are numbered sequentially beginning with the number 1, but numbers may be skipped, e.g. 1,3,4,7. When the first-order method of estimation is used, the typical value of the Mth parameter should be placed in GG(M,1). So when, say, volume of distribution is numbered 2, before exiting, PK should execute code like this:

† The INDXS array cannot be used with NM-TRAN abbreviated code.

```
GG(2,1) = TVVD
```

The argument ICALL functions similarly to the ICALL argument described in Guide I, section C.4.2. It has 5 possible values when PK is called.†

The value 1 signals to PK that the routine is being called for the first time in the NONMEM problem. At such a time PK must store certain information in array IDEF, but optionally, store certain information in GG. Here we discuss the matter concerning GG; the use of IDEF is discussed in sections G and H.

The value 2 signals to PK that the routine is being called in a regular fashion for data analytic purposes and that values of PK parameters are to be stored in the first column of GG. These can be typical values, as is described in this section, or they can be subject-specific values (see sections D and E). For data analytic purposes, however, it is not sufficient to compute values of PK parameters. Certain partial derivatives are also needed; see sections D and E.

The value 4 signals to PK that the routine is being called in a regular fashion for data simulation purposes. If the data are population data, (simulated) subject-specific values of PK parameters are to be stored in the first column of GG; see section E.2. If however, the data are all from a single subject, so that the subject's specific values are synonomous with the typical values, then at ICALL=4 typical values are stored in this column.

The value 5 signals to PK that the routine is being called in a regular fashion when expectations are being computed; multiple calls occur. Expectation blocks are described in the help Guide VIII. No eta derivatives need be computed.

If there is abbreviated code in the $PK block that tests for ICALL=0, ICALL=1, or ICALL=3, this code is moved by NM-TRAN to the INFN routine as if it had been coded explicitly as part of an $INFN block. Such code is called $PK-INFN code. The initialization code described in the next paragraph is generated in FSUBS by NM-TRAN regardless of the presence of $PK-INFN code.

At ICALL=1, 0's and 1's should be stored in the first column of GG. Usually, a 0 should be stored in GG(M,1), indicating that the user acknowledges that when ICALL=2 (or 4), the typical (or subject-specific) value of the Mth PK parameter will be placed in GG(M,1). When ICALL=1, the value passed to PK in GG(M,1) is 0; so if the user stores nothing in GG(M,1), he is achieving the same effect. If, though, a 1 is stored in GG(M,1), the user is specifying that when ICALL=2 (or 4), the (natural based) logarithm of the typical (or subject-specific) value of the Mth PK parameter will be placed in GG(M,1).† PREDPP will exponentiate this logarithm so to obtain the typical (or subject-specific) value of the PK parameter. If this option is chosen, then at ICALL=2 the code for GG(2,1) might look like this:

```
ATVVD = LOG(THETA(1)*SIZE)
GG(2,1) = ATVVD
```

which would be appropriate for model (4) and which would have the same effect as the above code, except that it would execute more slowly (because an extra logarithm and exponentiation are involved). Alternatively, the code for GG(2,1) might look like this:

```
ATVVD = LOG(THETA(1))+THETA(2)*LOG(EVTREC(2,1))
                                 +THETA(3)*LOG(EVTREC(2,2))
GG(2,1) = ATVVD
```

which would also have the same effect as the above code, except that it would execute about as fast

† For a complete list of ICALL values and called routines, see Chapter VI.

† The logarithms of PK parameters cannot be modelled in this way with NM-TRAN.

(because A**B is computed as EXP(B*LOG(A)).

The argument NETAS equals the total number of user-defined η variables. The user may possibly find this argument useful, particularly for implementing models for subject-specific values of PK parameters.

D. Modeling Subject-Specific Values of Pharmacokinetic Parameters

A model for subject-specific PK parameter values is needed for population data analysis and for the simulation of population data. Models for typical PK parameter values are discussed in section B, and their implementation in PREDPP is discussed in section C. If all the data come from the same subject, then the subject's specific value of a PK parameter is simply his typical value, the discussions in sections B and C suffice, and the discussion in this section D is not applicable.

The typical value is to be associated with the subpopulation of individuals sharing the same set, x, of values for the concomitant variables. Any given individual of this subpopulation, though, has his own specific value of the PK parameter. Unexplainable interindividual variability refers to differences that exist between these subject-specific values. In this section models for subject-specific PK parameter values are discussed. Such a model gives the relationship between (a) a subject's specific value of a PK parameter, and (b) the typical value for that (type of) subject and the random interindividual effects accounting for the difference between the subject's specific value and his typical value. Also, as will be seen, with such a model concomitant variables may have an effect on (a) other than through the typical value.

Clearly, by accounting for the difference between the subject's specific value and his typical value, across all subjects in the subpopulation, one also accounts for unexplainable interindividual variability. By doing so with random effects, this variability is modeled as arising randomly.

The simplest type model for an individual's specific value P^* of a PK parameter P is

$$P^* = \tilde{P} + \eta^* \tag{9}$$

where $\tilde{P}$ is the typical value of P, but more specifically, the mean P in the subpopulation of individuals whose concomitant values are those given by x, and where η^* is the realization (i.e. value) of a random variable η with mean 0 and variance ω^2. The variable η is a random effect accounting for the unexplained interindividual variability in P throughout the subpopulation; its realization η^* changes from individual to individual. We shall henceforth omit the asterisk from a PK parameter, P, when denoting a subject-specific value of P, and also henceforth omit the asterisk from a random variable such as η when denoting a subject-specific realization of the variable. Due to the context in which these symbols will be used little problem should result from this ambiguity in notation. Consequently, (9) may be rewritten

$$P = \tilde{P} + \eta \tag{10}$$

If $\tilde{P}$ is given in turn by (2), then we could write

$$P = \theta_1 WT + \eta \tag{11}$$

but for the purposes of what follows, it shall not be necessary to expand $\tilde{P}$ in terms of elements of x. However, we next describe how η may in turn be further modelled in terms of the elements of x, and so these elements thus may appear explicitly in the final model for P.

Actually, $P - \tilde{P}$ may not be entirely unexplainable. For example, it might be that there are two groups of individuals, identifiable by some dichotomous (0-1) valued concomitant variable, Z, say, and that metabolic clearance may vary more widely in one group than in the other, all other values of the concomitant variables being equal. In other words, for some random variable η_1 with mean 0 and variance ω_1^2,

$$\mathrm{Cl_{met}} - \tilde{\mathrm{Cl}}_{\mathrm{met}} = \eta_1$$

if Z = 0

$$= \theta_6 \eta_1$$

if Z = 1

Written differently,

$$\mathrm{Cl_{met}} = \tilde{\mathrm{Cl}}_{\mathrm{met}} + \theta_6^Z \eta_1 \tag{12}$$

So η in (10) has been expressed in terms of yet another random variable η_1. While η_1 has homogeneous variance, η does not; the variance of η is ${\omega_1}^2$ if Z=0 and $\theta_6^2 {\omega_1}^2$ if Z=1. Note that parameters like θ_6 may enter the model for $\mathrm{P} - \tilde{\mathrm{P}}$ and may be estimated.

For the purposes of using NONMEM, the user should become familiar with expressing the model for P in terms of random variables with means 0 and *homogeneous* variances. So for example, (12) is preferred to

$$\mathrm{Cl_{met}} = \tilde{\mathrm{Cl}}_{\mathrm{met}} + \eta$$

where η is the variable with inhomogeneous variance considered above.

Another simple model for P is

$$\mathrm{P} = \tilde{\mathrm{P}}(1 + \eta) \tag{13}$$

where the mean and variance of η are 0 and ω^2, respectively. Here ω is the coefficient of variation of P in the subpopulation. Instead of η having homogeneous variance ω^2, perhaps $\eta = \theta_6^Z \eta_1$, as above. In any case, under (13), $\mathrm{P} - \tilde{\mathrm{P}}$ again can depend on x, if only through $\tilde{\mathrm{P}}$.

The random variables (with homogeneous variance) occuring in a model for P may be regarded as having a population meaning beyond the particular subpopulation corresponding to x. They are independent of x. With every individual sampled from the larger population, there are associated with the individual (i) a particular set of values for the concomitant variables (some of which, like a dose, may be controlled by the investigator), and (ii) a particular set of realizations of the random variables. The variances of the random variables quantify random interindividual variability in P in the larger population, after the values of the concomitant variables are taken into account. We think of the random variables (as we do with the concomitant variables) as describing different population effects (although, unlike the concomitant variables, these effects are unobservable), and we think of their variances as a kind of population parameter. These variances may be estimated. The random effects confer the characteristics of a random variable to P itself. With model (10), the standard deviation of P is constant in the population if η has homogeneous variance. With model (13), the standard deviation of P in the population is proportional to $\tilde{\mathrm{P}}$.

The mean and variance of a random variable are suitable measures of centrality and dispersion, respectively, if the distribution of the variable is sufficiently Gaussian-like. Often the distribution of a PK parameter P (for fixed x) is significantly right-skewed in the population being sampled, and then the use of models like (10) and (13), and the quantification of random interindividual variability in terms of the variances of the involved η variables, are not very appropriate. A more appropriate model might be

$$\mathrm{P} = \tilde{\mathrm{P}} \, \exp \eta \tag{14}$$

where the mean and variance of η are 0 and ω^2, respectively. This model is, of course, equivalent to

$$\log\ P = \log\ \tilde{P} + \eta \tag{15}$$

If the distribution of η is Gaussian, then the distribution of P is lognormal. In any case, $\tilde{P}$ is the geometric mean of P, and exp ω is the geometric standard deviation of P. When ω^2 is sufficiently small, and η is Gaussian distributed, the distribution of P itself is Gaussian-like, and model (13) is not too bad an approximation to model (14). When ω^2 is sufficiently small, the mean and coefficient of variation of P are approximately $\tilde{P}$ and ω, respectively.

If metabolic clearance and renal clearance are modeled by

$$Cl_{met} = \tilde{Cl}_{met}\ \exp\ \eta_{met} \tag{16}$$

$$Cl_{ren} = \tilde{Cl}_{ren}\ \exp\ \eta_{ren}\ , \tag{17}$$

then total clearance might be given by

$$Cl = \tilde{Cl}_{met}\ \exp\ \eta_{met} + \tilde{Cl}_{ren}\ \exp\ \eta_{ren}\ . \tag{18}$$

This illustrates that a PK parameter might be modeled in terms of more than one η type variable. Also note that (18) cannot be written equivalently in terms of additive η's, as in (15), since the logarithm does not distribute over a sum.

In examples (10), (13), and (14), $\tilde{P}$ is obtainable from the model for the subject-specific value of P by setting η to its mean value, 0 (the typical value of η). By analogy, a typical value for total clearance can be obtained from (18) by setting both η_{met} and η_{ren} to 0, yielding

$$\tilde{Cl} = \tilde{Cl}_{met} + \tilde{Cl}_{ren} \tag{19}$$

(see (8) of section B.1). However, this typical value is neither a mean nor geometric mean. A model for a subject-specific value of a PK parameter has been described in this section as being dependent on a model for a typical value. In general though, a model for a typical value can always be obtained from a model for a subject-specific value in the way just illustrated. In fact, when NONMEM/PREDPP needs a typical value, but a model for subject-specific values has been coded (see section E), the program will obtain the typical value in this way.

The reader should recognize that the η variables discussed above are the same type of η variables discussed in Guide I. Two such random effects can correlate across individuals, and examples of this and the way one can communicate this to NONMEM and obtain estimates of covariability are described in that document.

Conditional estimates of the η's used in the model for a parameter P are obtained by searching for those values for the η's that minimize a certain objective function. Values are tried which vary somewhat independently of $\tilde{P}$. So it is possible that values of P result that are outside the meaningful range of the parameter and at which meaningful kinetic predictions are not computable. For example, if P is given by (13), large enough negative values of η may be tried which produce negative values of P, whereas P could be the volume of distibution, for which negative values are meaningless. For this reason, and because of the possiblility that the distibution of P might be significantly right-skewed, a model like (14) is often preferable when conditional estimates are computed. (However, it may not be actually necessary to use (14), and the more so P is symmetrically distributed, the less of a problem it is to use (13).)

Estimates of the η's do not result from using the first-order estimation method. The only value of an η variable used with this method is 0. As long as $\tilde{P}$ is a meaningful value of P, the kinetic predictions are computable. Therefore, from this point of view neither (13) nor (14) is preferable when the first-order method is used. Indeed, with first-order estimation models (13) and (14) cannot be distinguished; see discussion below. Conceptually though, η varies between $-\infty$ and ∞, even if the value 0 is the only value used in the computation. So, strictly speaking, model (13) can at best only be an approximate statistical model for P.

In fact, PREDPP checks that computed values of *certain* PK parameters are meaningful, e.g. that certain rate constants are positive, and if a value is not meaningful, PREDPP avoids the computation of kinetic predictions with this value and returns a PRED error-recovery code to NONMEM so that NONMEM understands that the "guilty" values of the η's cannot serve as estimates; see section K.1. (A check can be included in PK itself, and an immediate return to NONMEM with a PRED error-recovery code can be executed; see section K.2). Often, this allows a model such as (13) to be used when conditional estimates are computed; meaningful kinetic predictions can always be computed and meaningful estimates of the η's can be obtained. Nonetheless, when the distribution of P is significantly right-skewed in the population, use of model (14) can produce a better description of random interindividual variability in P, and this may not be detected when (13) is the only model tried and inherent problems with using (13) are masked.

When using a conditional estimation method, it is also possible for values of several parameters to result which are not meaningfully related. For example, suppose the kinetics are linear, one compartment with first-order absorption (ADVAN2), and that the elimination and absorption rate constants and the volume of distribution are given by

$$\begin{aligned} \mathrm{ke} &= \tilde{\mathrm{ke}}\ \exp \eta_1 \\ \mathrm{ka} &= \tilde{\mathrm{ka}}\ \exp \eta_2 \\ \mathrm{V} &= \tilde{\mathrm{V}}\ \exp \eta_3. \end{aligned} \tag{20}$$

(Here V is needed as a scaling parameter (see section F), not for the computation of compartment amounts.) Then values of η_1 and η_2 may be tried which produce values ke $\geq$ ka, whereas for the drug in question, suppose only ke < ka is meaningful. With these values of η_1 and η_2 meaningful kinetic predictions can be computed, but only if the roles of ke and ka are reversed in the kinetic model. However, reversing their roles entails reversing the roles of $\tilde{\mathrm{ke}}$ and $\tilde{\mathrm{ka}}$, and also of η_1 and η_2, and therefore, also of ω_1^2 and ω_2^2 (as well as changing the meanings of $\tilde{\mathrm{V}}$, η_3, and ω_3^2). The quantities $\tilde{\mathrm{ke}}$, $\tilde{\mathrm{ka}}$, $\tilde{\mathrm{V}}$, ω_1^2, ω_2^2, ω_3^2 are *population* quantities, applying to all individuals (with given x), and fixed in value for the purpose of estimating the η's. Changing their meanings, so that the parameter values of ke, ka, and V are meaningful for one individual, entails changing their meanings as they apply to all individuals. Under such a reinterpretation of these population quantities, and with their given values, it is now possible that values of the η's for yet another individual might be tried which give rise to nonmeaningful values ke, ka, and V for him. So a problem remains. The well-known parameter "flip-flop" phenomenon is not handled as easily in population PK data analysis as it is in single-subject PK data analysis.

When ADVAN2 is used, the user can check in PK whether ke $\geq$ ka, and if so, can force PREDPP to avoid the computation of kinetic predictions and return a PRED error-recovery code to NONMEM, so that NONMEM understands that the "guilty" values of the η's cannot serve as estimates (see section K.2). However again, a better solution is to try another type of model involving η's, e.g.

$$\begin{aligned} \mathrm{ke} &= \tilde{\mathrm{ke}}\ \exp \eta_1 \\ \mathrm{d} &= \tilde{\mathrm{d}}\ \exp \eta_2 \\ \mathrm{ka} &= \mathrm{ke}\ +\ \mathrm{d} \\ \mathrm{V} &= \tilde{\mathrm{V}}\ \exp \eta_3. \end{aligned} \tag{21}$$

where constraints on θ's are used to ensure that $\tilde{\mathrm{d}} \geq 0$. This model explicitly recognizes that ka > ke in the population. Therefore, it also implies that ke and ka cannot be statistically independent (even if η_1 and η_2 are assumed to be independent). Model (20), with or without the assumption that η_1 and η_2 are independent, is at best only an approximate statistical model for ke and ka.

Generally speaking, the PK routine specifies a subject-specific model for (all) the PK parameters. It does this in different ways, depending on whether PREDPP is being called for the purposes of data analysis, or data simulation, or both, and depending on the estimation method being used. For the purposes of data simulation, the specification uses the type of mathematical expressions for subject-specific values shown above.

For the purposes of data analysis, the specification can entail expressions for subject-specific values, such as those shown above, or instead, it can entail expressions for typical values. In either case, it also always entails expressions for a set of first partial derivatives of the model for the subject-specific values of the PK parameters with respect to the η's. For the purpose of data analysis using the Laplacian method, the specification further entails expressions for a set of second-partial derivatives. The matter of first-partial derivatives is addressed first.

The first-partial derivatives of the model for the subject-specific values of the PK parameters with respect to the η's, as functions of the η's, are called the <u>subject-specific first-partial derivatives</u> . For (12)-(14) and (18) for example, the first-partials are

$$\text{(12):} \quad \frac{\partial \mathrm{Cl}_{\mathrm{met}}}{\partial \eta_1} = \theta_6^{\mathrm{Z}} \tag{22}$$

$$\text{(13):} \quad \frac{\partial \mathrm{P}}{\partial \eta} = \tilde{\mathrm{P}} \tag{23}$$

$$\text{(14):} \quad \frac{\partial \mathrm{P}}{\partial \eta} = \tilde{\mathrm{P}} \ \exp \eta \tag{24}$$

$$\text{(18)} \quad \frac{\partial \mathrm{Cl}}{\partial \eta_{\mathrm{met}}} = \tilde{\mathrm{Cl}}_{\mathrm{met}} \ \exp \eta_{\mathrm{met}} \tag{25}$$

$$\frac{\partial \mathrm{Cl}}{\partial \eta_{\mathrm{ren}}} = \tilde{\mathrm{Cl}}_{\mathrm{ren}} \ \exp \eta_{\mathrm{ren}} \tag{26}$$

These types of expressions are used whenever conditional estimates are computed. They are also used when the first-order estimation method is used, but then the first-partials must be evaluated at all η's equal 0. These first-partial derivatives are called the <u>typical first-partial derivatives</u> . For the above examples these are

$$\text{(12):} \quad \frac{\partial \mathrm{Cl}_{\mathrm{met}}}{\partial \eta_1} = \theta_6^{\mathrm{Z}} \tag{27}$$

$$\text{(13):} \quad \frac{\partial \mathrm{P}}{\partial \eta} = \tilde{\mathrm{P}} \tag{28}$$

$$\text{(14):}\quad \frac{\partial \mathrm{P}}{\partial \eta} = \tilde{\mathrm{P}} \tag{29}$$

$$\text{(18)}\quad \frac{\partial \mathrm{Cl}}{\partial \eta_{\mathrm{met}}} = \tilde{\mathrm{C}}\mathrm{l}_{\mathrm{met}} \tag{30}$$

$$\frac{\partial \mathrm{Cl}}{\partial \eta_{\mathrm{ren}}} = \tilde{\mathrm{C}}\mathrm{l}_{\mathrm{ren}} \tag{31}$$

Note that the derivatives (28) and (29) are identical. With the first-order estimation method, the model for subject-specific values of the PK parameters is fully defined by specifying the typical values and the typical first-partial derivatives. Since the typical values are the same under models (13) and (14), and since the derivatives (28) and (29) are also the same, the first-order estimation method can never distinguish between models (13) and (14). That is, the same fit will result from using either model. In effect, an assumption is being made that the variance of η in (14) is small, and that the mean and coefficient of variation of P under model (14) are well approximated by $\tilde{\mathrm{P}}$ and ω, respectively. With the conditional estimation methods, however, the model for subject-specific values of the PK parameters is defined by specifying the subject-specific values themselves, along with subject-specific partial derivatives. Since expressions (13) and (14) differ for some values of η, the population conditional estimation methods can distinguish between models (13) and (14) when the data allow this.

It should be emphasized that the typical first-partial derivatives, despite their name and the fact that to obtain them all η's are set to zero, convey information about the model for subject-specific values. They are rates of change of PK parameters with respect to interindividual effects.

As noted in section C, the PK routine allows a model to be defined for log P, rather than for P. The derivatives of log P with respect to the involved η's, rather than the derivatives of P itself, may be specified. The subject-specific (and typical) first-partial derivative of log P from (15), for example, is

$$\text{(15):}\quad \frac{\partial \log \mathrm{P}}{\partial \eta} = 1 \tag{32}$$

PREDPP transforms $\dfrac{\partial \log \mathrm{P}}{\partial \eta}$ to $\tilde{\mathrm{P}}\,\dfrac{\partial \log \mathrm{P}}{\partial \eta}$ since it needs the latter.

Just as typical values can always be obtained from expressions for subject-specific values, so can typical first-partials.

Second-partial derivatives are needed when the Laplacian estimation method is used. The second-partial derivatives of the model for the subject-specific values of the PK parameters with respect to the η's, as functions of the η's, are called the <u>subject-specific</u> <u>second-partial</u> <u>derivatives</u> . These often are simply 0. For the above examples these are

$$\text{(12):}\quad \frac{\partial^2 \mathrm{Cl}_{\mathrm{met}}}{\partial \eta_1 \partial \eta_1} = 0 \tag{33}$$

$$\text{(13):}\quad \frac{\partial^2 \mathrm{P}}{\partial \eta \partial \eta} = 0 \tag{34}$$

$$(14):\quad \frac{\partial^2 \mathrm{P}}{\partial\eta\partial\eta} = \tilde{\mathrm{P}}\ \exp\ \eta \tag{35}$$

$$(18)\quad \frac{\partial^2 \mathrm{Cl}}{\partial\eta_{\mathrm{met}}\partial\eta_{\mathrm{met}}} = \tilde{\mathrm{Cl}}_{\mathrm{met}}\ \exp\ \eta_{\mathrm{met}} \tag{36}$$

$$\frac{\partial^2 \mathrm{Cl}}{\partial\eta_{\mathrm{ren}}\partial\eta_{\mathrm{ren}}} = \tilde{\mathrm{Cl}}_{\mathrm{ren}}\ \exp\ \eta_{\mathrm{ren}} \tag{37}$$

$$\frac{\partial^2 \mathrm{Cl}}{\partial\eta_{\mathrm{ren}}\partial\eta_{\mathrm{met}}} = 0 \tag{38}$$

The subject-specific second-partial derivative of log P from (15) is

$$(15):\quad \frac{\partial \log\ \mathrm{P}}{\partial\eta} = 0 \tag{39}$$

Second-partial derivatives of the model for the subject-specific values of the PK parameters with respect to the η's, evaluated at all η's equal to 0 (i.e. typical second-partial derivatives) may, of course, also be considered, but they are never needed in NONMEM computations.

E. Implementing Models for Subject-Specific Values in PK

For the purpose of data analysis with population data, models for the subject-specific values must be communicated to PREDPP. When the first-order estimation method is used, this involves communicating the typical values of the PK parameters (see section C), and also the typical first-partial derivatives, the implementation of which is discussed in section E.1. When a conditional estimation method is used, or posthoc estimates of η's are desired, this involves communicating subject-specific PK parameter values and subject-specific first-partial derivatives. Implementation of the former is discussed in section E.2, and implementation of the latter is discussed in section E.3. Also the simulation of population data uses subject-specific values of PK parameters. The Laplacian method uses subject-specific second-partial derivatives, and the implementation of these is discussed in section E.4.

The first-order method can also be used when subject-specific values and subject-specific first-partial derivatives are communicated. Implementation of this mode of communication is generally preferable for the development of new PK codes, for although one may *intend* to only use the first-order method, one might actually end up needing to compute conditional estimates (e.g. posthoc estimation of η's).

When all the data come from a single subject, both subject-specific values and derivatives are irrelevant, and this section is not applicable. For the purpose of reading this section the reader should be familiar with section C.

E.1 Typical First-Partial Derivatives

If the first-order estimation method is used, typical first-partial derivatives must be computed (see section D). The η's involved in the models for the subject-specific values of the PK parameters are numbered according to the enumeration of the initial estimates of their variances in NONMEM (or NM-TRAN) control records. The derivative of the Mth PK parameter with respect to η_K should be placed in GG(M,1+K). (The Mth PK parameter is defined in section C.) So if (total) clearance is the lst PK parameter and is given by (18), and if η_{met} and η_{ren} are the 4th and 5th η variables, respectively, then one needs code like

```
GG(1,1) = TVCLMT+TVCLRN
 ...
GG(1,5) = TVCLMT
GG(1,6) = TVCLRN
```

(see section D equations (19),(30),(31)).

All values GG(1,1+K), $K \neq 4, 5$, should be 0. However, since whenever PK is called, the GG array is initialized to zero immediately before the call, the user need not explicitly store zeros in elements of GG.

By storing a 1 in GG(M,1) at ICALL=1, the user specifies that when ICALL=2, the typical value of the logarithm of the Mth PK parameter will be placed in GG(M,1) (see section C).† This signal also means that the typical first derivative of the logarithm of the Mth PK parameter with respect to η_K will be placed in GG(M,K+1). To take an example, if P = Vd is the 2nd PK parameter, if log P is given by (15), and if η in (15) is the 1st η variable, then one needs code like

```
GG(2,1) = ATVVD
GG(2,2) = 1
```

See section C for examples of ATVVD. In this example when ICALL=1, one also needs GG(2,1)=1.

E.2 Subject-Specific Values

When ICALL=4, PK is being called during the Simulation Step, and then subject-specific values must be computed. When ICALL=2, PK is being called for the purpose of data analysis, and when conditional estimates are involved, then too, subject-specific values must be computed. When the first-order estimation method is used, it suffices to compute subject-specific values, since typical values can always be obtained from subject-specific computations (section D).

Subject-specific values are stored in the first column of the GG array, as are typical values when they are stored; see section C. However again, since typical values can always be obtained from subject-specific value computations, subject-specific values may be computed and stored in the first column whenever both types of values may be needed. As an example, when both simulation and data analysis using the first-order estimation method occur in the same run, subject-specific values should be computed and stored. Or, when a run involves posthoc estimation of η's, subject-specific values should be computed and stored. As a final example, when a run involves two problems, one using the first-order method, and another using a conditional method, subject-specific values should be computed and stored.

The subject-specific value of the Mth parameter is stored in GG(M,1). So if (total) clearance is the lst PK parameter and is given by (18), and if η_{met} and η_{ren} are the 4th and 5th η variables, respectively, then one needs code like

```
USE NMPRD_REAL,ONLY: ETA
 ...
CALL GETETA (ETA)
 ...
GG(1,1) = TVCLMT*EXP(ETA(4))+TVCLRN*EXP(ETA(5))
```

ETA is a one-dimensional array used to store values of $\eta_1, \eta_2, \cdots$, needed for the computation of subject-specific values of the PK parameters.

† The logarithms of PK parameters cannot be modelled in this way with NM-TRAN.

See Section I.A for a discussion of module NMPRD_REAL.
See Section K.2 for a discussion of GETETA and IQUIT. When ICALL=4, the values of η_1, η_2, $\cdots$ are obtained by a call to the NONMEM utility routine SIMETA. An example of the use of SIMETA is given in section L.1. When ICALL=2, these values are obtained by a call to the NONMEM utility routine GETETA.

If the NONMEM run is only for the purpose of simulation, a simple call to SIMETA at ICALL=4, preceding the first reference to ETA in an executable statement, suffices to obtain the η values. If the NONMEM run does not involve simulation, a simple call to GETETA at ICALL=2, preceding the first reference to ETA in an executable statement, suffices to obtain the η values, as in the above example. However, a run could entail calls to PK with values of ICALL=2 and 4. Or, the user might prefer that PK be coded to allow such a possiblity in a future run using the PK routine. In this case the following type of code can be written.

```
IF (ICALL.EQ.4) CALL SIMETA (ETA)
IF (ICALL.EQ.2) CALL GETETA (ETA)
 ...
GG(1,1) = TVCLMT*EXP(ETA(4))+TVCLRN*EXP(ETA(5))
```

Lastly, GETETA must always be initialized at ICALL=1. This involves simply calling GETETA at ICALL=1. So, the code actually might look like:

```
IF (ICALL.EQ.1) THEN
 ...
   CALL GETETA (ETA)
 ...
   RETURN
ENDIF
 ...
IF (ICALL.EQ.4) CALL SIMETA (ETA)
IF (ICALL.EQ.2) CALL GETETA (ETA)
 ...
GG(1,1) = TVCLMT*EXP(ETA(4))+TVCLRN*EXP(ETA(5))
```

The initialization call does not result in values of η's being stored in ETA. Only calls to GETETA at ICALL=2 or 4 result in η's being stored. Often initialization of GETETA is not the only task that is undertaken at ICALL=1; see sections G and H.

As stated earlier in this section, when the first-order method is used, and when the only values of PK parameters that are needed are typical values, expressions for subject-specific values may be coded instead. When the first-order method is used, GETETA stores zeros in ETA, and then the subject-specific values become the required typical values.

By storing a 1 in GG(M,1) at ICALL=1, the user specifies that when ICALL=2 or 4, the subject-specific value of the logarithm of the Mth PK parameter will be placed in GG(M,1) (see section C).†

Something further about simulation: By default, as long as PK is being called with an event record from the same individual record, each time SIMETA is called, the values η_1, η_2, $\cdots$ stored in ETA remain the same; there is only one set of values obtained for the individual. However, the simulation can be done in such a way that the values change each time SIMETA is called (see Guide IV, section III.B.13). Then only the first time PK itself is called with an event record of a given individual record should PK

† The logarithms of PK parameters cannot be modelled in this way with NM-TRAN.

call SIMETA (see section H for a discussion about the sequence of calls to PK). This assures that there is only one set of values obtained for the individual, as in the default situation. Unlike that situation, though, during this first call to PK, multiple calls to SIMETA might occur. So for example, simulated values of η_1, obtained from multiple calls to SIMETA and such that $|\eta_1| \geq 2$, can be rejected until a value < 2 is obtained, i.e. the distribution on η_1 can be truncated. The code might look like this:

```
IF (ICALL.EQ.1) THEN
 ...
 CALL GETETA (ETA)
 ...
 RETURN
ENDIF
 ...
IF (ICALL.EQ.4) THEN
   IF (NEWIND.NE.2) THEN
5      CALL SIMETA (ETA)
      IF (ABS(ETA(1)).GE.2.)  GO TO 5
   ENDIF
ENDIF
IF (ICALL.EQ.2) CALL GETETA (ETA)
  ...
GG(1,1) = TVCLMT*EXP(ETA(4))+TVCLRN*EXP(ETA(5))
```

The variable NEWIND allows PK to know when it is being called for the first time with an event record of a given individual record (i.e. NEWIND not equal to 2); see section I.

E.3 Subject-Specific First-Partial Derivatives

For the purpose of data analysis, routine PK is called with ICALL=2, at which time derivatives must be computed. If a conditional estimation method is used or posthoc estimates of the η's are desired, subject-specific first-partial derivatives must be computed (see section D). If the Laplacian method is used, subject-specific second-partial derivatives must also be computed; see section E.4.

Note that, with NONMEM 7, NONMEM may not always require that first-partial derivatives be computed. A global integer variable IFIRSTEM is set by NONMEM to the value 1 or 0, according as the first-derivatives are needed or not. The usage of IFIRSTEM is similar to that of MSEC in Section E.4.†

The η's involved in the models for the subject-specific values of the PK parameters are numbered according to the enumeration of the initial estimates of their variances in NONMEM (or NM-TRAN) control records. The derivative of the Mth PK parameter with respect to η_K should be placed in GG(M,1+K). (The Mth PK parameter is defined in section C) So if (total) clearance is the lst PK parameter and is given by (18), and if η_{met} and η_{ren} are the 4th and 5th η variables, respectively, then one needs code like

† A different global variable, MFIRST was used in earlier releases of NONMEM. IFIRSTEM should be used with NONMEM 7.2 and higher. Note also that it is possible to cause PK and ERROR and other PREDPP subroutines to compute first-derivatives even when IFIRSTEM is 0; see FIRSTEM in the help Guide VIII.

```
IF (ICALL.EQ.1) THEN
 ...
   CALL GETETA (ETA)
 ...
   RETURN
ENDIF
 ...
CALL GETETA (ETA)
 ...
GG(1,1) = TVCLMT*EXP(ETA(4))+TVCLRN*EXP(ETA(5))
GG(1,5) = TVCLMT*EXP(ETA(4))
GG(1,6) = TVCLRN*EXP(ETA(5))
```

All values GG(1,1+K), K ≠ 4, 5, should be 0. However, since whenever PK is called, the GG array is initialized to zero immediately before the call, the user need not explicitly store zeros in elements of GG.

By storing a 1 in GG(M,1) at ICALL=1, the user specifies that when ICALL=2 or 4, the subject-specific value of the logarithm of the Mth PK parameter† will be placed in GG(M,1) (see section C). This signal also means that the subject-specific first derivative of the logarithm of the Mth PK parameter with respect to η_K will be placed in GG(M,K+1).

E.4 Subject-Specific Second-Partial Derivatives

If the Laplacian estimation method is used, subject-specific first *and* second-partial derivatives are required (see section D). The second-partial derivatives should be computed when ICALL=2. If one might use the Laplacian method, then it is a good idea to develop a PK code that accommodates this. If the Laplacian method is not used and the second-partial derivatives are computed, then they are ignored. See also the remarks below concerning MSEC.

When second-partial derivatives are computed, the GG argument is dimensioned differently from the way this is described in section C. Its dimension needs to be expressed thusly: GG(IRGG,GPKD+1,*) The subject-specific value of the Mth PK parameter should be placed in GG(M,1,1). The first-partial derivative of the Mth PK parameter with respect to η_K should be placed in GG(M,1+K,1). The second-partial derivative of the Mth PK parameter with respect to η_K and η_L should be placed in GG(M,1+K,1+L). The matrix of second-partial derivatives is symmetric, so it is only necessary to store second-partial derivatives for values K ≥ L. Consider the example where (total) clearance is the lst PK parameter and is given by (18), and η_{met} and η_{ren} are the 4th and 5th η variables, respectively. Then one needs code like

† The logarithms of PK parameters cannot be modelled in this way with NM-TRAN.

```
IF (ICALL.EQ.1) THEN
 ...
   CALL GETETA (ETA)
 ...
   RETURN
ENDIF
 ...
CALL GETETA (ETA)
 ...
GG(1,1,1) = TVCLMT*EXP(ETA(4))+TVCLRN*EXP(ETA(5))
GG(1,5,1) = TVCLMT*EXP(ETA(4))
GG(1,6,1) = TVCLRN*EXP(ETA(5))
GG(1,5,5) = GG(1,5,1)
GG(1,6,6) = GG(1,6,1)
```

All values GG(1,1+K,1+L), K $\neq$ 4, 5, should be 0. However, since whenever PK is called, the GG array is initialized to zero immediately before the call, the user need not explicitly store zeros in elements of GG.

In the above example, there are only two nonzero second-partial derivatives of clearance that must be explictly stored in GG. However, even these two are not actually needed with every call to PK. (Certainly, they are never needed unless the Laplacian method is being used.) In order to save computation time, information is provided in the NONMEM global variable MSEC as to whether second-partial derivatives are needed with a particular call to PK. This is particularly useful when there are nonzero second-partial derivatives of a number of PK parameters, and the total number of such derivatives is large. MSEC is set by NONMEM to the value 1 or 0, according as the second-derivatives are needed or not. Consequently, an alternative code to the above might be:

```
USE NMPRD_REAL,ONLY: ETA
 ...
IF (ICALL.EQ.1) THEN
 ...
   CALL GETETA (ETA)
 ...
   RETURN
ENDIF
 ...
CALL GETETA (ETA)
 ...
GG(1,1,1) = TVCLMT*EXP(ETA(4))+TVCLRN*EXP(ETA(5))
GG(1,5,1) = TVCLMT*EXP(ETA(4))
GG(1,6,1) = TVCLRN*EXP(ETA(5))
 ...
IF (MSEC.EQ.1) THEN
   GG(1,5,5) = GG(1,5,1)
   GG(1,6,6) = GG(1,6,1)
 ...
ENDIF
```

where all second-partials are computed and stored only when MSEC equals 1.

By storing a 1 in GG(M,1,1) at ICALL=1, the user specifies that when ICALL=2 or 4, the subject-specific value of the logarithm of the Mth PK parameter will be placed in GG(M,1,1) (see section C). This signal also means that the subject-specific first derivative of the logarithm of the Mth PK parameter with respect to η_K will be placed in GG(M,1+K,1) and that the subject-specific second-partial derivative of the logarithm of the Mth PK parameter with respect to η_K and η_L will be placed in GG(M,1+K,1+L).

E.5 Active Eta List for PREDPP

During a given call to PRED, NONMEM may not need drivatives for all η's. A PK, TRANS, or ERROR routine may compute derivatives for all η's or optionally, to reduce run time, only for active η's. NONMEM tells PRED how many etas are active (NACTIV) and a mapping such that M(k) is the index of the kth active eta. The declarations are:

```
USE PROCM_INT, ONLY: NACTIV,M=>IDXETA
```

PREDPP Library routines TRANS2, TRANS3, etc., use this feature, but without the alias M for IDXETA. This is a matter of style.

F. Modeling Values of Additional Pharmacokinetic Paramaters

As mentioned in section C., with every translator routine, TRANS, there is associated a different list of PK parameters. These parameters are called the basic PK parameters. They form a "minimal set" of PK parameters whose typical/subject-specific values and η-derivatives must be set in PK (see sections C and E). There are additional PK parameters whose use in a given problem are somewhat optional. In this section we describe them and give some examples for modeling them. As with the basic parameters, their typical/subject-specific values and η-derivatives are communicated to PREDPP in PK. The way to do this is described in section G.

F.1 Scaling Parameters

Associated with each observation is an observation compartment . This compartment is specified either explicitly in the event record containing the observation (section V.H), or by a default designation (see sections VI.B and VII.C). For each observation, NONMEM computes a prediction. The amount A in the observation compartment at the time of observation, divided by the value of a parameter S, is used as the prediction. The parameter S is called a scaling parameter . There is one such parameter associated with every compartment of the structural model (including the output compartment). In NM-TRAN abbreviated code, the scaling parameters have reserved names Sn or SC (where n is the compartment) or SC (for the central compartment).

Suppose the observation is a plasma concentration. Then the observation compartment should be taken to be the plasma compartment, and the S of that compartment should be taken to be the volume of distribution of that compartment. (Volume of distribution may or may not also be a basic PK parameter.) Suppose the observation is a urine concentration. Then the observation compartment should be taken to be the urine compartment, which in turn might be identified with the output compartment, and the S of that compartment should be taken to be the measured volume of urine. Whereas, as in earlier sections, volume of distribution is usually modeled in terms of θ's, η's and x, urine volume is usually a measured quantity and therefore simply some element of x. However, in principle each scaling parameter (or any of the PK parameters being described in section F) can be modeled in terms of θ's, η's, and x.

Scaling parameters are optional in the sense that scaling parameters associated with compartments never observed may be ignored. The values of scaling parameters that are not computed in PK are always understood to be 1 (see section G). Therefore, if, an amount, rather than a concentration, is measured, the computation of the scaling parameter may be ignored in this case also. If a scaling parameter is not ignored and is computed in PK to be nonpositive, PREDPP exits with a nonzero PRED error return code (see section K).

The scaling parameter for a given compartment acts discretely at times for which predictions of the scaled amount in this compartment are computed (see section B.2). If volume of distribution is a basic PK parameter, it acts continuously in that capacity. However, when, for example, the scaling parameter for the plasma compartment is set equal to the volume of distribution, the volume parameter acts discretely as it acts through the scaling parameter.

F.2 Bioavailability Fractions

Every dose is associated with a dose compartment , as specified either explicitly in the dose event record (see section V.H), or by a default designation (see sections VI.B and VII.C). This compartment is usually the compartment where the dose is physically input, although it need not be (see section F.3). If the dose is a bolus dose or a regular infusion, the dose amount must also be specified on the dose event record. If the amount is A, an amount F $\cdot$ A of drug actually appears in the dose compartment (either instantaneously at the time the dose enters the compartment - with a bolus dose, or over a period of time - with an infusion), where F is the value of the bioavailability fraction . There is one bioavailability fraction (parameter) associated with every possible dose compartment of the structural model (the output compartment is not a possible dose compartment). In NM-TRAN abbreviated code, the bioavailability fractions have reserved names Fn, where n is the compartment number. The bioavailability fraction for a given compartment acts discretely at the times doses enter (or start to enter) the system (see section B.2). For lagged doses, these are lagged times; see section F.6. Bioavailability fractions are optional in the sense that bioavailability fractions associated with compartments never used as dose compartments may be ignored. The values of bioavailability fractions that are not computed in PK are always understood to be 1 (see section G). Therefore, if, bioavailability cannot be estimated, the computation of the bioavailability fraction can be ignored in this case too, with the consequence that it is assumed that the drug is 100% available. If a bioavailability fraction is not ignored and is computed in PK to be negative, PREDPP exits with a nonzero PRED error return code (see section K).

If two different preparations are given into the same dose compartment, and the concomitant Z assumes the value 1 or 2 according to which preparation is being given with some particular dose, then a model for F might be

$$F = \theta_1 \ \exp \eta/(1+\exp \eta) \tag{40}$$

$$\text{if } Z = 1$$

$$= \theta_2 \ \exp \eta/(1+\exp \eta)$$

$$\text{if } Z = 2$$

With this model the typical value of F is θ_1 or θ_2 according to the preparation given. [The variable $\exp \eta/(1+\exp \eta)$ ranges from 0 to 1 and has typical value $\frac{1}{2}$; so if θ_1 and θ_2 are between 0 and 1, F is also.] On the other hand, the CV of (random) interindividual variability in F is approximately the same for both preparations (viz. $\frac{1}{2}\,\omega$). Under this model, if both preparations are given to some individuals, the bioavailabilities of the two preparations are perfectly correlated across these individuals (because with each such individual η is the same between preparations). With another model,

$$F = \theta_1 \ \exp \eta_1/(1+\exp \eta_1) \tag{41}$$

$$\text{if } Z = 1$$

$$= \theta_2 \ \exp \eta_2/(1+\exp \eta_2)$$

$$\text{if } Z = 2$$

the correlation depends on the degree to which η_1 and η_2 are correlated, which may be 0. The correlation between η_1 and η_2 can be estimated (provided the data allow this); see Guide I. With model (41) the CV of interindividual variability may differ between preparations ($\approx \frac{1}{2}\,\omega_1$ and $\frac{1}{2}\,\omega_2$).

F.3 Duration Parameters

There are two types of bolus doses that may be given. An instantaneous bolus dose of amount A is such that at the time the dose enters the system, the amount F · A of drug appears instantaneously in the dose compartment, where F is the value of the bioavailability fraction associated with the dose compartment. A zero-order bolus dose of amount A is such that its appearance in the dose compartment is described by a zero-order process over a finite time interval, such that the total amount appearing over this interval of time is F · A, where F is the value of the bioavailability fraction associated with the dose compartment. The appearance of drug in a depot compartment, resulting from the dissolution of a preparation placed therein, is an example of drug appearance that may be modeled by a zero-order process. The appearance of drug in the central compartment, resulting from absorption of a preparation placed in a depot compartment, is another example of drug appearance that may be modeled by a zero-order process, although this is often modeled by a first-order process. In this example, the dose compartment would need to be the central compartment, even though the dose was physically input into a depot. A zero-order bolus dose, just as an instantaneous bolus dose, may have a lag time, in which case the zero-order process starts at the lagged time; see section F.6.

The difference between a regular infusion and a zero-order bolus dose is that the duration of a regular infusion is specified by information in the dose event record and computed by PREDPP itself, whereas the duration of a zero-order bolus dose is regarded as a parameter which may be modeled and computed by the PK routine. Of course, a model for the duration can be as simple as setting this parameter to some data item in the dose record that gives the duration of a regular infusion. Information in the dose record indicates that a dose is a zero-order bolus dose, rather than a regular bolus dose or an infusion; see sections V.E.

There is one duration parameter associated with every possible dose compartment of the structural model. In NM-TRAN abbreviated code, the duration parameters have reserved names Dn, where n is the compartment number. The duration parameter associated with a given compartment acts discretely at the times zero-order bolus doses start to enter the compartment (see section B.2). A zero-order bolus dose whose duration is modeled is called a duration- modeled zero-order bolus dose . Duration parameters are optional in the sense that duration parameters associated with compartments never receiving duration-modeled zero-order bolus doses may be ignored. The values of duration parameters that are not computed in PK are always understood to be 0 (see section G). If a duration parameter is not ignored and is computed in PK to be nonpositive, PREDPP exits with a nonzero PRED error return code (see section K).

Alternatively, the rate of a zero-order bolus dose may be modeled and computed by the PK routine; see next section. Some zero-order bolus doses may be duration-modeled, and others may be rate-modeled.

F.4 Rate Parameters for Zero-Order Bolus Doses

The zero-order rate of a zero-order bolus dose (see section F.3) may be modeled, instead of its duration. Information in the dose record indicates which is modeled, the duration or the rate; see section V.E. A zero-order bolus dose whose rate is modeled is called a rate-modeled zero-order bolus dose . There is one rate parameter associated with every possible dose compartment of the structural model. In NM-TRAN abbreviated code, the rate parameters have reserved names Rn, where n is the compartment number. These rate parameters are optional in the sense that rate parameters associated with compartments never receiving rate-modeled zero-order bolus doses (or rate-modeled steady-state infusions; see next section) may be ignored. The values of rate parameters that are not computed in PK are always understood to be 1 (see section G). If a rate parameter is not ignored and is computed in PK to be nonpositive, PREDPP exits with a nonzero PRED error return code (see section K).

Rate parameters act continuously. Therefore, PREDPP obtains the value of a rate parameter, holding over the state-interval (t_1, t_2), from a call to PK with the argument record associated with t_2, even if the dose event time occurs before t_2. Therefore, if there are state times (e.g. t_2) falling within the time interval over which a zero-order bolus dose appears in the system, there exists the possibility that the rate of drug input can change during the interval. For this to occur, the rate parameter would need to be modeled in terms of time varying concomitant values. As a result, a better description of the zero-order process where the rate is modeled might be a piecewise zero-order process. The release of drug from a sustained release capsule, designed to occur at different rates at different stages of release, might be modeled using a rate parameter. This model could be very simple, depending only on the manufacturer's design parameters and not necessarily on parameters whose values must be estimated.

The duration of a rate-modeled zero-order bolus dose entering a compartment may be regarded as a derived parameter, i.e. as a parameter computed by PREDPP from the primary additional parameters, in this case from the bioavailability fraction and rate parameter for the given compartment. This parameter, H, acts discretely at all state times t_2 such that there is an amount A_1 of drug remaining to be input into the compartment at the state time t_1 preceding t_2. At t_2 its value is $(F \cdot A_1)/r$, where F is the bioavailability fraction applied to the dose at the time it started to enter the system, and r is the value of the rate parameter at time t_2. If one wants H to be independent of F, r can be set to $F \cdot r'$, where r' is a nominal rate. For example, if one wants H to be a given value d (e.g. a regular infusion is given of known duration d), then one should set $r = F \cdot (A/d)$. If in fact the dose is a regular infusion, then A is the amount A_0 on the dose record, and PK can obtain this data item. In this case, though, it is simpler to implement a model for the duration parameter (of section F.3) than to implement the model for the rate parameter. If A cannot be obtained as a data item, then in general the value of H cannot be controlled. One exception occurs when H is constant all the while the dose enters the compartment. Then set $r = F \cdot (A_0/d)$, where d is the constant value (so $H = (A_1/A_0) \cdot d$).

Since two rate-modeled zero-order bolus doses into the same compartment share the same rate parameter, care should be taken that the intervals over which they appear in the system not overlap, or that if these do overlap, that the two possible values of the rate parameter be the same.

F.5 Rate Parameters for Steady-State Infusions

The rate of a (constant rate) steady-state infusion (see section V.F) may be modeled. Such an infusion is called a <u>rate-modeled steady-state infusion</u>. Information in the dose indicates that the infusion is rate-modeled; see section V.E. There is one <u>rate parameter</u> associated with every possible dose compartment of the structural model. The rate parameter that one uses for a given compartment is the same one used to determine the rate of zero-order bolus doses into the compartment. That is, rate-modeled steady-state infusions and zero-order bolus doses into the compartment share the same rate parameter. In NM-TRAN abbreviated code, the rate parameters have reserved names Rn, where n is the compartment number. Rate parameters are optional in the sense that rate parameters associated with compartments never receiving rate-modeled steady-state infusions or rate-modeled zero-order bolus doses may be ignored. The values of rate parameters that are not computed in PK are always understood to be 1 (see section G). If a rate parameter is not ignored and is computed in PK to be negative, PREDPP exits with a nonzero PRED error return code (see section K). Rate parameters may be 0. Steady-state doses with both amount and rate 0 are useful with general nonlinear models when the differential equations explicitly provide for endogenous drug production and there is no exogenous drug to be introduced.

Steady-state infusions are imagined as infusions which started long before time 0 and terminate at the event time on the dose event record. Rate parameters act continuously. In the case of a rate-modeled steady-state infusion terminating at time t, it should be imagined that the infusion rate is constant over the infinite interval from $-\infty$ to t and is obtained from a call to PK with the argument record associated with

t. Since a rate-modeled zero-order bolus dose and a rate-modeled steady-state infusion into the same compartment share the same rate parameter, care should be taken when two such doses occur, that the infusion does not terminate during the interval over which the bolus dose appears in the system, or that if it does, that the two values of the rate parameter are the same, or that some appropriate strategy is used.

One possible use of a rate-controlled steady-state infusion occurs when a patient has been on chronic drug therapy before entering a study, but one is uncertain about the dosing history. By modeling this history with a rate-controlled steady-state infusion (terminating at time 0) whose typical rate is, say, θ (but which depends also on an individual random effect), then all compartments are initialized at the outset with amounts commensurate with the assumed kinetics, but which will depend on a simple (and presumably) estimable parameter. For this model to make sense, one should be able to regard the differences in the pre-study drug histories between those patients whose histories are uncertain as being random.

Another possible use is to model the kinetics of a drug which is also present endogenously. As above, a rate-modeled steady-state infusion can be used to initialize the compartments to endongenous amounts. A very large amount of the compound administered thereafter (but with time data item also equal to 0), as a rate-modeled zero-order bolus dose (see section F.4), can maintain the "endogenous steady-state".

F.6 Absorption Lag Times

The time t on a dose record refers to the recorded time the dose was administered. In the case of a regular infusion, t is the time the infusion was initiated. (In the case of a steady-state infusion, t is the time the infusion *terminates*, but absorption lag times do not apply to steady-state infusions.) An <u>absorption lag time</u> is an increment of time L such that the time that the dose is regarded (by PREDPP) as entering (or starting to enter) the system is t+L. An absorption lag time is sometimes simply called the <u>lag time</u> . The time t+L is called the <u>lagged time</u> , and a dose with a positive lag time is called a <u>lagged dose</u> .

Absorption lag times act discretely at recorded dose times. That is, at such an event time t an absorption lag time L for the dose is computed by the PK routine. The lagged time t+L is not an event time; it is a nonevent dose time. If there actually is an event time coinciding with the lagged time, this is only coincidental. The bioavailability fraction and duration parameter act at the time t+L, when the dose actually enters (or starts to enter) the system. Normally, PK is called to compute these parameters with only the argument record associated with t+L. This record generally does not contain information that might be used in the computation. If PK requests calls at nonevent dose times (see section III.H), PK can compute these parameters using both the argument record associated with t+L and the the dose record describing the initiating dose.

When additional doses are specified on a dose event record (see section V.K), the absorption lag time acting at the time on the dose record applies to the dose and to all the additional doses. The lag time may exceed the (length of the) interdose interval. There is no restriction in PREDPP that absorption delay for a given dose event record must be completed before a new dose is entered in to the system. With a steady-state multiple dose (see section V.F) the absorption lag time applies not only to this dose, but also to all the preceding implied doses. With such a dose, the lag time should not exceed the interdose interval.

There is one absorption lag time (parameter) associated with every possible dose compartment of the structural model (the output compartment is not a possible dose compartment). In NM-TRAN abbreviated code, the absorption lag times have reserved names ALAGn, where n is the compartment number. Absorption lag times are optional in the sense that absorption lag times associated with compartments never used as dose compartments may be ignored. The values of absorption lag times that are not computed in PK are always understood to be 0 (see section G). If an absorption lag time parameter is not ignored and is computed in PK to be negative, PREDPP exits with a nonzero PRED error return code (see section K).

As with any PK parameter, a lag time may be modeled in as complicated a way as is desired; the model may include η's. However, data can often be insufficient to allow a lag time to be well-estimated, and even when a typical lag time can be estimated well enough, one may not be able to estimate the interindividual variance of the lag time. In this case either set the variance set to zero, or do not use an η.

F.7 The Output Fraction

With any of the kinetic models a (peripheral) output compartment is always present. Associated with this compartment is a PK parameter, the output fraction , denoted here by F_o. Of the entire amount, A_o, of drug introduced into the system by various dosage patterns and then eliminated from the system during a state-interval, a fraction of this amount, $F_o \cdot A_o$, goes into this output compartment. The output compartment may be turned on and off. While on, drug accumulates therein, and when turned off, the amount therein is reset to zero. So, for example, if the output compartment is regarded as a urine compartment, and F_o is the ratio of renal to total clearance, the initiation and termination of a urine collection can be simulated. In NM-TRAN abbreviated code, the output fraction has reserved name F0 or FO or Fm (where m is the compartment number of the output compartment.)

If the output compartment is never turned on, the output fraction can be ignored. If the value of the output fraction is not computed in PK, it is always understood to be 1 (see section G). Consequently, if the output fraction is ignored, then the amount in the output compartment would eventually equal 100% of all drug input into the system, provided the system does not retain any drug indefinitely, drug administration finally ceases, and the output compartment is always on. If the output fraction is not ignored and is computed in PK to be less than 0 or greater than 1, PREDPP exits with a nonzero PRED error return code (see section K). The output fraction acts continuously.

The example mentioned above might, more specifically, be

$$F_o = \frac{Cl_{ren}}{Cl} \tag{42}$$

where Cl_{ren} and Cl are given as in (17) and (18). Under this model, the typical value of F_o is

$$\tilde{F}_o = \frac{\tilde{C}l_{ren}}{\tilde{C}l} \tag{43}$$

and the subject-specific first-partial derivatives are

$$\frac{\partial F_o}{\partial \eta_{ren}} = F - F^2 \tag{44}$$

$$\frac{\partial F_o}{\partial \eta_{met}} = - F \frac{Cl_{met}}{Cl} \tag{45}$$

The use of F_o depends on the assumption that the rate of change of drug amount in the output compartment is linear in the other compartment amounts. Other than this linearity restriction, the system can be nonlinear.

F.8 The Time Scale parameter

In earlier sections it has been suggested that unexplained interindividual variablity in kinetic responses might be modeled in terms of random interindividual effects on familiar kinetic parameters. Alternatively, a population kinetic model can be entertained wherein at least some of the unexplained interindividual variability is attributable to what appears to be random differences between individuals' biological clocks. A simple low-dimensional description of unexplained interindividual variability, albeit somewhat empirical, results when all such variability is attributed to this source and to random interindividual differences in scaling parameters.

According to this idea, time itself is scaled differently between individuals. However, since some time intervals, such as an infusion time, are always measured on an external clock (e.g. the nurse's wristwatch), time is scaled only where it multiplies a rate constant (of a linear model). The scaling parameter is an additional PK parameter X which is modeled in PK. This parameter is called the time scale parameter (or sometimes, the X parameter). In NM-TRAN abbreviated code, the time scale parameter has reserved name XSCALE. There is a single time scale parameter that applies to all rate constants. The parameter acts continuously (and could therefore theoretically itself vary with time measured on an external clock). It can only be used with linear kinetic models. If it is not used, it can be ignored. If the value of the time scale parameter is not computed in PK, it is always understood to be 1 (see section G). If it is not ignored and is computed in PK to be nonpositive, PREDPP exits with a nonzero PRED error return code (see section K).

Random interindividual effects can be assumed to affect the time scale parameter and the scaling parameters (see section F.1). If x denotes time, and S is a scaling parameter, then (ignoring random intraindividual variability) an observation y can be written schematically as

$$Sy = A(Xx)$$

where A is drug amount as a function of time, and where, say,

$$S = \theta_1 \exp \eta_1$$

$$X = \theta_2 \exp \eta_2 .$$

Time x may be regarded as the abscissa value, the observation y may be regarded as the ordinate value, and then one sees that X scales the abscissa and S scales the ordinate. Random interindividual kinetic differences are being attributed at least in part to random interindividual differences in the abscissa and ordinate scales.

The X parameter does not scale the duration parameter D of a duration-modeled zero-order bolus dose. If this is desired, this must be done by setting D = XD′, where D′ is an unscaled duration parameter.

F.9 Model Event Time parameters

Model event times are additional PK parameters defined in the PK routine or $PK block. A model event time is not associated with any compartment, but, like an absorption lag time, defines a time to which the system is advanced. When the time is reached, PREDPP sets indicator variables and a call to PK is made. At this call (and/or subsequent to this call) PK or DES or AES or ERROR can use the indicator variables to change some aspect of the system, e.g., a term in a differential equation, or the rate of an infusion. This feature may be used with any ADVAN routine. If a model event time is also an event time, it is only coincidental. In NM-TRAN abbreviated code, the model event times have reserved names MTIME(i). Reserved variable MTDIFF may be set when MTIME variables are changed. Indicator variables have reserved names MNEXT(i) and MPAST(i), each corresponding to the MTIME(i) variable with the same index i. Reserved variable MNOW is also set by PREDPP when indicator variables are set. Details are in Appendix III.

G. Implementing Models for Additional Parameters

The argument ICALL of PK was described in section III.C. It functions similarly to the ICALL argument described in Guide I, section C.4.2. It has several possible values when PK is called. The value 1 signals to PK that the routine is being called for the first time in the NONMEM problem. At such a time PK must store certain values in array IDEF, telling PREDPP what, if any, additional PK parameters the user has chosen to model, and where their typical/subject-specific values and η-derivatives will be stored in the GG array. Usually, a model for at least one additional parameter, e.g. a scaling parameter, is given in PK. IDEF is a two-dimensional array†. To see how the two-dimensional formatted IDEF is declared in PK, see section C.

The first row of IDEF is also used to inform PREDPP what tasks PK will perform at later calls, and this is described first.

IDEF(1,1)=-9 (required)
IDEF(1,2) is the PK calling protocol (call-limiting element) (see Section H).

IDEF(1,3) describes whether PK performs compartment initialization, i.e., whether or not PK initializes elements of the initial state vector A_0(n) (See section I.B). Values are:
-1: PK may initialize A_0.
0: PK does not initialize A_0.
1: PK does initialize A_0.
The default used by PREDPP is IDEF(1,3)=-1. However, when compartment initialization is not implemented, then if IDEF(1,3) is set to 0, PREDPP can avoid some time-consuming processing. Indeed, when $PK abbreviated or verbatim code is supplied, and there is no reference to compartment initialization amounts A_0(n) in either the abbreviated or verbatim code, then NM-TRAN sets IDEF(1,3)=0.

IDEF(1,4) describes whether PK uses derivatives of compartment amounts (e.g. compartment amounts themselves are used as random variables in arithmetic statements in PK) (see Section I.A). Values are:
-1: PK may use derivatives of compartment amounts.
0: PK does not use derivatives of compartment amounts.
1: PK uses derivatives of compartment amounts.
The default used by PREDPP is IDEF(1,4)=-1. However, when derivatives of compartment amounts are not used, then if IDEF(1,4) is set to 0, PREDPP can avoid some time-consuming processing. Indeed, when $PK abbreviated or verbatim code is supplied, and there is no reference to A(n) (as a random variable in an arithmetic statement) in the abbreviated code (or to derivatives of A(n) in the verbatim code), then NM-TRAN sets IDEF(1,4)=0.

Just as typical/subject-specific values and η-derivatives for each of the basic PK parameters are stored in some row of the GG array, so are typical/subject-specific values and η-derivatives for each of the additional PK parameters. The rows can be assigned somewhat arbitrarily. If the output fraction is modeled, set IDEF(2,1) to the number of the row, called the <u>row index</u>, chosen for this fraction. If the time scale parameter is modeled, set IDEF(2,2) to the row index chosen for this parameter. If model event times (MTIME(i)) are modeled, set IDEF(2,3) to the row index of the lowest-numbered MTIME parameter. Set IDEF(2,4) to the row index of the hightest-numbered MTIME parameter. If the scaling parameter, bioavailability fraction, rate parameter, duration parameter, or absorption lag for the Ith compartment is modeled, set IDEF(3,I), IDEF(4,I), IDEF(5,I), IDEF(6,I), or IDEF(7,I), respectively, to the row index chosen for this parameter.

There is a number, M_B, that is the largest number of basic parameters permitted with the selected kinetic model. This number is either set in the selected ADVAN subroutine (see section VII.C) or set by the user via the MODEL subroutine (see section VI.B). A row index M assigned to an additional PK parameter must be a number greater than M_B, but no greater than PG, a constant in SIZES.f90 which is given by PARAMETER (PG=50+PCT), where PCT is the maximum number of model event time parameters given by PARAMETER (PCT=30) Both these parameters can be changed with the $SIZES record.

† Previous editions of this guide also described a one-dimensional format for IDEF. This format is obsolete and the descripiton is omitted.

Consider, for example, the one compartment linear model, with one basic PK parameter: rate constant of elimination. (This parameterization is implemented via TRANS1.) From section VII.C.1 it may be seen that $M_B = 2$. If the scaling parameter for compartment 1 is to be modeled, then one can set IDEF(3,1)=3. If the scaling parameter for compartment 2 is also to be modeled, then one can set IDEF(3,2)=4. Lastly, if the bioavailability fraction for compartment 2 is to be modeled, one can set IDEF(4,2)=5

The row indices of the additional PK parameters must be consecutive integers *beginning with* $M_B + 1$, with no integers skipped, as in the above example. However, one is not restricted to preserving an increasing monotonic relationship between the numbers of the compartments and their row indices, or between the numbers of the rows of IDEF itself and the row indices. So, in the above example one can just as well set IDEF(3,1)=4 and IDEF(3,2)=3, or set IDEF(3,1)=4, IDEF(3,2)=5, and IDEF(4,2)=3. Nor is one restricted from using a row index more than once. So, one can set IDEF(3,1)=IDEF(3,2)=3 (which specifies that the scaling parameters for compartments 1 and 2 are modeled and that their values are to be equal and stored in row 3 of GG), though usually there is no need to do this.

For each scaling parameter and bioavailability fraction, and for the output fraction and time scale parameter, PREDPP assumes that if its row index is not explicitly set at ICALL=1, then its typical/subject-specific value is always 1, and its η-derivatives are always 0. Therefore, for example, the row index of a bioavailability fraction associated with a plasma compartment receiving *only* intravenous bolus doses need not be explicitly set. (If this compartment were also a dose compartment for zero-order bolus doses with possibly less than 100% bioavailability, the row index would need to be set. In this case one would need to be careful that the fraction stored is less than 1 only when the fraction for the zero-order bolus dose is being obtained with the call to PK.) For each absorption time lag, PREDPP assumes that if its row index is not explicitly set at ICALL=1, then its typical/subject-specific value is always 0, and its η-derivatives are always 0. PREDPP assumes that if neither the row index of the duration parameter nor the row index of the rate parameter for a given compartment is explicitly set at ICALL=1, then the compartment never receives zero-order bolus doses.

The user can specify that the typical/subject-specific value and the η-derivatives of the logarithm of an additional PK parameter will be placed in GG, just as with the logarithm of a basic PK parameter (see sections C and E).†

H. PK Calling-Protocols

As the pharmacokinetic system is advanced, PK is called one or more times, each time with some argument record. The event records comprise these argument records, and are passed to PK in time order. The simulation and/or data analytic computations will normally be done correctly if routine PK is called with one event record after another (within an individual record), no event records being skipped, and no event record being repeated. This is the default. However, PREDPP can implement a few different protocols for calling PK. A protocol is specified by setting IDEF(1,2) to one of various values at ICALL=1 (for more about IDEF, see section G). For example, the PK routine can be called only with the first event record of the individual record and with every event record thereafter where the time data item differs from the time data item of the previous event record. If this more limited sequence of calls is desired, this can be accomplished by setting IDEF(1,2)=0. Note, though, that *in this case*, IDEF(1,2) must be explicitly set to 0 because 0 is not the default.

Often, none of the basic or additional PK parameters depend on concomitant variables whose values vary within an individual record, i.e. vary over time. In this situation the information output by PK, i.e. the GG array, is the same for each event record of an individual record (for fixed THETA and ETA). Considerable computation time can be saved; PREDPP need call PK *only once* per individual record, with the first event record only (for any given values of the THETA and ETA arrays). The values of continuously acting PK parameters computed with this call can be assumed to hold over all state-time

† The logarithms of PK parameters cannot be modelled in this way with NM-TRAN.

intervals for the individual record, and the values of discretely acting PK parameters can be assumed to hold at each state time for the record. The user can request this calling-protocol by setting IDEF(1,2)=1. This is illustrated in examples below; see sections L.1 and L.2.

When the data are from a single subject, PREDPP treats all event records in the entire data set as though they are associated with the same subject. (This is a consequence of the single-subject assumption; see section IV.A.) In particular, suppose that every call to PK with a different event record results in the same output from PK (for fixed THETA). Then only one call is necessary, a call with the first event record in the *entire data set*. The values of continuously acting PK parameters computed at this call can be assumed to hold over all state-time intervals for the entire data set, and the values of discretely acting PK parameters can be assumed to hold at all state times for the entire data set. Setting IDEF(1,2) to 1 has this effect. This is illustrated in examples below; see sections L.3 and L.4.

Even when IDEF(1,2)=0 or 1, a call to PK with any given event record can be forced with the use of the CALL data item (see section V.J).

The protocol where PK is called once with every event record (see above) can be specified by setting IDEF(1,2) to -1, or by not setting IDEF(1,2) at all, i.e. this protocol is the default. If it is desired that, in addition, the values of PK parameters at any nonevent dose time t be computed with access to both the argument record associated with t *and* the event record describing the dose (see section B.2), set IDEF(1,2)=-2. This has the effect that PK may be called repeatedly with the same event record (for if t is a nonevent dose time, and s is the subsequent event time, PK is called with the event record for s at both times t and s). The primary use of this protocol is so that the values of certain discretely acting PK parameters, the bioavailability fractions and duration parameters, can always be computed with access to useful dose-related concomitant information contained in the dose record, such as the type of preparation, even when the dose is an additional or lagged dose. Another use of this protocol is to cause PREDPP to call PK at the time specified by a model event time parameter; this protocol is required with such parameters.

When NM-TRAN abbreviated code is used, the reserved variable CALLFL may be used to specify the value for IDEF(1,2). E.g., to set IDEF(1,2)=0, use:

```
CALLFL=0
```

Alternately, A calling protocol phrase can be used instead of the CALLFL pseudo-statement, e.g.,

```
$PK (NEW EVENT TIME)
```

In the next section the way PK has access to DOSREC, the record describing the dose, is explained.

I. Global and Other Reserved Variables

PK can access variables other than though its argument list. With versions of NONMEM prior to NONMEM 7 FORTRAN COMMON blocks were used. With NONMEM 7, FORTRAN MODULES are used. The names of the COMMON's and MODULE's are not identical. Integer variables are in modules whose names ends with _INT. Real variables are in modules whose names ends with _REAL. Character variables are in modules whose names ends with _CHAR. The characters "RO" in a module name indicates "Read-Only", e.g., ROCM (NONMEM Read-Only) and PROCM (PREDPP Read-Only) and the variable should not be changed by PREDPP, i.e., should be used only on the right. Modules whose name starts with NMPR or NMPRD are for NONNEM-PRED communication. Variables may be used on the right or sometimes on the left. Variables in ROCM and NMPR and NMPRD modules are of interest to any PRED. Modules whose name starts with PR are for communication between PREDPP and its subroutines. This section lists some of the variables of interest for PREDPP. The module name immediately follows the USE clause. Variable name(s) are **bold** and follow the ONLY clause. Some variables listed below may be pointers to differently named variables in MODULES. Sometimes the aliasing feature "=>" is used so that a variable has a different name in the subroutine than in the module. The help Guide

VIII can be used to obtain the exact declarations and all variables of interest.

I.A Global Input Variables

USE NMPRD_INT , ONLY : **MSEC**=>ISECDER , **IFIRSTEM**

These variables are discussed in Sections E.3 and E.4, respectively.

USE PROCM_INT , ONLY : **NEWIND**=>PNEWIF

NEWIND is an integer variable acting like the NEWIND variable described in Guide I. NEWIND is an argument to PRED subroutines, but is communicated in a module to subroutines of PREDPP. It assumes one of 3 values: 0 if the event record is the first of the entire data set, 1 if the event record is the first event record of an individual record (other than the first individual record), and 2 if the event record is other than the first event record of an individual record. When the data are single-subject data, individual records are as described in chapter II.

USE PROCM_REAL , ONLY : **DOSTIM , DDOST , D2DOST**

When IDEF(1,2)=-2, two kinds of calls to PK may occur: at nonevent dose times (with additional and lagged doses) and at model event times (with model event time parameters). Nonevent dose times are discussed first.

The values of PK parameters at any nonevent dose time t are obtained with access to both the argument record associated with t *and* the event record describing the dose (see sections B.2 and H). At the call to PK, when the values of the PK parameters are obtained, the value of t itself can be found in DOSTIM. The event time when the dose is given is found in another module (DOSREC; see below).

The time t may depend on η's. The first-partial derivatives and the second-partial derivatives of t with respect to the η's are given in DDOST and D2DOST, respectively. DDOST(K) = ∂DOSTIM/$\partial\eta_K$, and DDOST(L,K) = ∂^2DOSTIM/$\partial\eta_L\eta_K$ for L $\geq$ K. *When PK is called at an event time, DOSTIM, DDOST and D2DOST contains 0's.*

USE PROCM_REAL , ONLY : **DOSREC**

When IDEF(1,2)=-2, the values of discretely acting PK parameters at any nonevent dose time t are obtained with access to both the argument record associated with t *and* the event record describing the dose (see sections B.2 and H). At the call to PK, when the values of the PK parameters are obtained, the (last data record of the) event record describing the dose can be found in DOSREC. (The event record can span several data records; see chapter II.) This record differs from the argument record (the event record found in the argument EVTREC), which is the event record at the next event time following t. The argument record may not even be a dose record, but if it is, it may describe a dose unlike the one entering at time t.

When PK is called at an event time, DOSREC contains 0's.

When NM-TRAN abbreviated code is used, labels of data items defined in the $INPUT record (user defined variables) may be used on the right in the $PK block. By default, the values used for these variables will be from the argument record EVTREC, the "next" record. The $BIND record may be used to request that values be from from the dose record. Values from the "last" (the most recent, i.e., the previous) event record be also be requested. Note that $BIND affects only the generated code, not PREDPP or the arrays DOSREC or EVTREC themselves.

USE NMPRD_REAL , ONLY : **ETA**

The ETA array is in a global module. This allows elements of ETA to be displayed by NONMEM in tables and scatterplots and allows both PK and ERROR to access the values of ETA.

USE NMPRD_INT , ONLY : **NTHES_**=>NWTHT , **NETAS_**=>NWETA , **NEPSS_**=>NWEPS

The dimensions of THETA, OMEGA, SIGMA. Never less than 1.

USE NMPR_REAL , ONLY : **OMEGA**=>VARNF

The current value of OMEGA. The current value of SIGMA is also located in this array as follows:

```
SIGMA(I,J)=OMEGA(NETAS_+I,NETAS_+J)
```

Other useful global variables follow.

USE PROCM_REAL , ONLY : **A**=>AMNT , **DAETA , D2AETA**

USE PROCM_REAL , ONLY : **TSTATE**

Global variable A is the state vector of compartment amounts (See Introduction to Version VI). Elements of A may be used on the right in $PK and PK subroutines.
A(n) = the amount in compartment n.
DAETA(n,i) = the derivative of A(n) wrt eta(i).
D2AETA(n,i,j) = the second derivative of A(n) wrt eta(i), eta(j)
(lower-triangular; j=1, ..., i)
TSTATE is the state time associated with A, i.e., the time t at which A was computed. It may also be used on the right in $PK and PK subroutines.

PK should tell PREDPP whether or not it uses compartment amounts. See Section G, IDEF(1,4).

REAL(KIND=DPSIZE) , POINTER :: **DEN_ , CDEN_(:)**

USE ROCM_REAL , ONLY : DEN_NP

DEN_=>DEN_NP(1)

CDEN_=>DEN_NP(2 :)

These values are computed by NONMEM when the Nonparametric step is performed and marginal cumulatives are requested. DEN_ is the nonparametric density. CDEN_(n) is the marginal cumulative value for the nth. eta. They may be used as right-hand quantities in PK and all PREDPP subroutines.

USE ROCM_INT , ONLY : **MIXNUM**=>MIXCALL , **MIXEST**=>IMIXEST

These are NONMEM variables used with Mixture models (see section L.2). They may be used on the right in $PK abbreviated code or PK subroutines.

MIXNUM is set by NONMEM during Estimation to the index of the subpopulation for which variables are to be computed.
MIXEST and MIXP are set by NONMEM when Estimation is finished to the index of the subpopulation estimated to be that from which the individual's data most probably arises.
MIXP are the mixture probabilities P(i) computed by NONMEM subroutine MIX or by the $MIX block of abbreviated code. They may be used on the right in PK.

USE PROCM_INT , ONLY : **MNOW**=>MTNOW , **MPAST**=>MTPAST , **MNEXT**=>MTNEXT

When IDEF(1,2)=-2 and PK is called at at model event times, the values of discretely acting PK parameters at any nonevent dose time t are obtained with access to the argument record associated with t. *DOSTIM and DOSREC are 0.* PK has access to global variables MNEXT, MPAST, MNOW. These indicator variables can be used to modify some aspect of the system. Details are in Appendix III.

USE ROCM_INT , ONLY : **NIREC**=>NINDREC , **NDREC**=>NDATINDR

USE NMPRD_INT , ONLY : **NPROB , IPROB**

INTEGER(KIND=ISIZE) , POINTER :: **S1NUM , S2NUM , S1NIT , S2NIT , S1IT , S2IT**

USE NMPRD_INT , ONLY : IDXSUP , NITR_SUP , NITSUP

S1NUM=>IDXSUP(1)

S2NUM=>IDXSUP(2)

S1NIT=>NITR_SUP(1)

S2NIT=>NITR_SUP(2)

S1IT=>NITSUP(1)

S2IT=>NITSUP(2)

USE ROCM_INT , ONLY : **NREP** , **IREP**=>NCREP

USE ROCM_INT , ONLY : **LIREC**=>NDATPASS

USE ROCM_INT , ONLY : **NINDR**=>NINDOBS , **INDR1**=>IDXOBSF , **INDR2**=>IDXOBSL

These are NONMEM "counter" variables. They may be used on the right in $PK or PK subroutines. For initialization/finalization, they may be used in $INFN or INFN subroutines. (see INFN in Chapter VI). Counters include (in the order above): record counters; problem iteration counters; super-problem iteration counters; simulation repetition counters; number of data records in the individual record; number of individual records in the data set containing an observation record, and the indices of the first and last such individual records.

See also Chapter IV section D.A for variables giving values of THETA, OMEGA, SIGMA at different NONMEM steps. Some of these may also be of interest to PK.

I.B Global Output Variables

This section describes variables that may be set by PK which are not communcated via the arguments but which are located in modules.

USE PROCM_INT , ONLY : **A_0FLG**

USE PRMOD_REAL , ONLY : **A_0 , DA_0 , D2A_0**

These variables are used with the compartment initialization feature of PREDPP. A_0FLG is an input variable to PK. It may be used on the right in $PK and PK subroutines. When PREDPP sets A_0FLG to 1, the amounts in the various compartments may be set by the PK routine.
A_0(n) = the amount for compartment n
DA_0(n,i) = the derivative of A_0(n) wrt eta(i)
D2A_0(n,i,j) = the second derivative of A_0(n) wrt eta(i), eta(j) (lower-triangular; j=1, ..., i)

PK must tell PREDPP whether or not it peforms compartment initialization. See Section G, IDEF(1,3).

USE NMPRD_INT , ONLY : **ETASXI**

With NONMEM 7.3, an alternative eta shrinkage evaluation using empirical Bayes variances (EBVs, or conditional mean variances) are now also reported. With NONMEM 7.4, reported shrinkage includes ETAshrinkSD, ETAshrinkVR and similarly for EBV and EPS. Reserved variable ETASXI(i) may be used in $PK abbreviated code or PK subroutine to specify certain etas of particular subjects to be included, or to specify certain etas of certain subjects to be excluded, from the average eta shrinkage assessment.
If ETASXI(i) is set to 2, ETA(i) is included.

If ETASXI(i) is set to 1, ETA(i) is excluded.

USE PRMOD_INT , ONLY : **I_SS**

I_SS is used for the Initial Steady State feature of PREDPP, with the general non-linear models (ADVAN6, ADVAN8, ADVAN9, ADVAN13, ADVAN14, ADVAN15). Default: -1 (PK does not request the I_SS feature).
Values 0, 1, 2 and 3 are permitted. Value 0 requests that no steady state be computed. ISSMOD values 1, 2, and 3 requests that PREDPP compute an initial steady state for the model before the first event record of an individual record, or after a reset event. The results are identical to those that would be computed by a steady-state dose event record with SS=I_SS and AMT=0 and RATE=0. If endogenous drug is specified in the differential equations, non-zero initial conditions will be computed. There is no difference between values 1, 2 and 3 of I_SS unless the PK routine also uses the compartment initialization feature A_0. The I_SS feature behaves exactly like a steady state dose record in this regard. Specifically,

With I_SS=1 ("reset"), values of A_0 are ignored.
With I_SS=2 ("sum"), values of A_0 are added to the SS values.
With I_SS=3 ("initial ests"), values of A_0 are used as initial estimates when computing the SS values.
See also ISSMOD (MODEL subroutine ; Chapter VI section B.A).

USE PKERR_REAL , ONLY : **MTIME**

USE PRMOD_INT , ONLY : **MTDIFF**

MTIME (model event time variables) are communicated to PREDPP as additional PK parameters in the GG array. However, they are communicated to subroutine ERROR as global variables in the PKERR_REAL module (note that eta derivatives of MTIME variables are not available to the ERROR subroutine.) MTDIFF is a global variable located in a module. If PK sets MTDIFF to a value other than 0, e.g., MTDIFF=1, then PREDPP will understand that with that call to PK, the values of one or more of the MTIME(i) have possibly been reset. Details are in Appendix III.

USE NMPR_INT , ONLY : **RPTI**=>NRPT_IN , **RPTO**=>NRPT_OUT , **RPTON**=>NRPT_ON

USE NMPR_INT , ONLY : **PRDFL**=>IUSEPRD

These NONMEM variables provide the information controlling the Repetition feature of NONMEM (See Introduction to Version VI). All variables are output from PREDPP except RPTI, which is set by NONMEM. However, they are listed together in module NMPR_INT and are described here together. RTPO, RPTON, PRDFL may be used on the left in $PK and PK subroutines, as well as ERROR and all other PREDPP subroutines. RPTI is set by NONMEM and may be used on the right.

RPTO is used to mark a record as a repetition base or a repetition initiator.
RPTON gives the number of times the repetition series initiated by the data record is to be repeated.
RPTI is set by NONMEM when the record being passed to PRED is being repeated.
PRDFL signals that the output from PRED with a passed record is to be ignored by NONMEM.

I.C Miscellaneous Global Variables

USE PRINFN , ONLY : TLCOM=>ITV

PRINFN is a global module for INFN-defined variables (See Chapter VI, Section A.C). It is meant to be used for communication with other other blocks of abbreviated code or with user-written codes. When NM-TRAN is used, there is a subroutine ASSOCPRINFN in FSUBS similar to ASSOCNMPRD4 (See Section J). This makes it convenient for PK and ERROR to use variables that are initialized or modified in INFN at ICALL values 0 or 1 or 3.

USE DECLAREVARIABLES

DECLAREVARIABLES is an area of storage that is defined using the $ABBREVIATED DECLARE control record with NONMEM 7.3.

Declared variables are global, i.e., are defined in all blocks of abbreviated code. Examples are integer variables for use in DOWHILE loops, or double precision arrays. Declared variables are not used by NONMEM or PREDPP.

include **nonmem_reserved_general**

There is a file nonmem_reserved_general in the util directory. This contains USE statement for a number of other NONMEM variables that may be useful for advanced users. Most of these are variables that are set by NONMEM and can be used in PREDPP (e.g., ITER_REPORT, Iteration number that is reported to output). Some are meant to be set by PREDPP and are used by NONMEM (e.g., MDVI1, MDVI2, MDVI3 which can be used to override the value of MDV=100 or MDV=101 under certain circumstances).

I.D Other Reserved Variables

Other reserved variables may be computed in PK that are not in the argument list or in global modules.

MU_ variables

MU_ variables are useful with NONMEM methods such as Bayesian methods. The association of one or more THETA's with ETA(n) should be identified by a variable called MU_n. If MU_ variables are defined in $PK abbreviated code or PK subroutine, they must also be computed in a subroutine called MUMODEL2. This subroutine should contain only the code that is needed to compute the MU parameters (MU_1, MU_2, etc.). This subroutine may be empty. If NM-TRAN abbreviated code is used, MUMODEL2 is generated by NM-TRAN. As much as possible, define the MU's in the first few lines of $PK or the PK routine. A description of the MUMODEL2 subroutine is beyond the scope of this document.
See Chapter VII.

J. Displaying PK-Defined Items

A value stored in a variable (or array element) V in PK may be displayed in a table or scatterplot. To accomplish this, module NMPRD4 must be defined in PK, and V must be listed in NMPRD4.† Module NMPRD4 also provides a convenient place to store values of variables to be shared between PK and other user routines, and it is used thusly when these routines are generated from NM-TRAN abbreviated code (see Guide IV). (INFN-defined and declared variables are also shared between user routines.)

The implementation of NMPRD4 is different in NONMEM 7 than in earier versions. In order to facilitate dynamic storage allocation, module NMPRD4 contains a single dynamically-allocated array VRBL. The declaration for NMPRD4 is:

†(When there is more than one data record within an event record R2, the value of V computed when PK is called with the preceding event record R1 is displayed as part of the last data record of R1, *and* as part of every data record but the last in R2.) See the example in section L.2. This is not possible when NM-TRAN is used.

```
MODULE NMPRD4
  USE SIZES, ONLY: DPSIZE
  REAL(KIND=DPSIZE), ALLOCATABLE, TARGET :: VRBL(:)
END MODULE
```

where the allocated size is given in SIZES.f90 by PARAMETER (LNP4=4000) and LNP4 can be changed with the $SIZES record.

Within a user-supplied code, the declaration

USE NMPRD4 , ONLY : COM=>VRBL

makes it possible to use the elements of COM as in previous versions of NONMEM. When NM-TRAN is used, the generated code is a little more complicated. FSUBS contains a subroutine ASSOCNMPRD4 containing statements such as

```
REAL(KIND=DPSIZE), DIMENSION (:),POINTER ::COM
COM=>VRBL
KA=>COM(00001);K=>COM(00002);CL=>COM(00003);SC=>COM(00004)
```

ASSOCNMPRD4 is called from PK, ERROR and other subroutines in FSUBS. This allows the generated code to use the user-supplied variables KA, K, etc., which may be helpful to a user who examines FSUBS. E.g.,

```
KA=THETA(001)+ETA(001)
```

is more similar to the abbreviated code than either of the following, which are equivalent:

```
VRBL(1)=THETA(001)+ETA(001)
COM(1)=THETA(001)+ETA(001)
```

Other reserved variables may be used with NMPRD4.

COMRES

COMRES ("common reserve" is a reserved word in abbreviated code and the $ABBREVIATED record that may be used to gives instructions to NM-TRAN about NMPRD4. (It has no sigificance to NONMEM.) It can be used to set aside an initial portion of NMPRD4 as a "reserved" portion. Reserved elements of NMPRD4 may be referred to as COM(i) in abbreviated code, verbatim code, and $TABLE and $SCATTER record.

USE NMPRD_INT , ONLY : **COMACT**

COMACT is set by NONMEM. It may be tested in user-written subroutines or abbreviated code to determine when NONMEM is making a copying pass, i.e., when the data records are being passed to PRED for the purpose of computing values of variables which will be obtained (i.e. copied) from NMPRD4 for tables and scatterplots. NONMEM only makes a copying pass when PRED-defined items are listed in $TABLE or $SCATTER records. The values used in tables and scatterplots are those copied from NMPRD4 with the last copying pass. Values of COMACT are:

COMACT=0: This is not a copying pass. COMACT=1: This is a copying pass with final thetas and zero-valued etas. COMACT=2: This is a copying pass with final thetas and conditional estimates of etas. COMACT=3: This is a copying pass with conditional (nonparametric) estimates of etas.

USE NMPRD_INT , ONLY : **COMSAV**

COMSAV is set by PREDPP. With NM-TRAN, it can be specified as an option of the $ABBREVIATED record. It gives the size of the SAVE Region of NMPRD4. If a variable is stored in the SAVE region, then the value of the variable computed with a given data record during a copying pass will be found in NMPRD4 when the same record is passed during the next copying pass, i.e. it will have been saved from the previous copying pass. This is in contrast to the usual behaviour, where with a given data record, the value in NMPRD4 is the value computed with the previous data record.

K. PRED Error-Recovery

PREDPP may exit with a nonzero PRED error return code. Then either the NONMEM run is immediately aborted or an error-recovery procedure is implemented. An error-recovery procedure entails continued calls to PRED, but with values of THETA or ETA different from those with previous calls which resulted in nonzero return codes. There are two error-recovery procedures: one with which different ETA values are tried, the ETA-recovery , a second with which different THETA (and possibly ETA) values are tried, the THETA-recovery . Whenever it is possible to implement the ETA-recovery, this is done. If this procedure fails, or if it is not possible to implement the ETA-recovery, and the error return code is obtained during either the search in the Estimation Step or the search in the Initial Estimation Step, then a choice exists between an abort and implementation of the THETA-recovery. If the THETA-recovery fails, or if it is not actually possible to implement the THETA-recovery, the NONMEM run is aborted.

A PRED error return code can have values 0, 1, or 2. The value 0 means that the return is a normal return. The value 1 means that if the choice exists between an abort or implementation of the THETA-recovery, then this choice is to be made using control stream information. The value 2 means that if the choice exists between an abort or implementation of the THETA-recovery, then the abort should be chosen.

When an abort occurs, an error message will appear in the NONMEM output, in the intermediate output file (if such a file exists), and in the PRED Error file. When the THETA-recovery is implemented, an error message appears in the intermediate output file (if such a file exists), in the PRED Error file, and if recovery is not possible, in the NONMEM output. The error message is called a PRED error message .

When the PRED error return code is 1, and a choice exists between implementation of the THETA-recovery and an abort, the THETA-recovery is implemented if the value in field 11 (field 3) of the ESTIMATION (THETA CONSTRAINT) record is 1 (or with NM-TRAN, if the NOABORT option is used in the $ESTIMATION ($THETA) record).† If the value is 0 (or if the NOABORT option is not used), then the run is aborted. Often, the most appropriate response to an abort occuring during the Initial Estimate Step, or during the Estimation Step after the 0th iteration summary has been output, is to rerun the problem requesting that the THETA-recovery be implemented. *Warning*: If the implementation of the THETA-recovery is requested before an actual abort has occured, be sure to check the PRED Error file PRDERR for possibly useful diagnostic information that is otherwise available in NONMEM output when an abort occurs.

K.1 PRED Error-Recovery from PREDPP

PREDPP may exit with a nonzero PRED error return code as a result of some computation undertaken in a PREDPP kernal or Library routine, or when PK returns an invalid value of a PK parameter. The nature of the problem will be described in the PRED error message if this message appears.

K.2 PRED Error-Recovery from PK

PK (as well as ERROR, DES, and AES) can can force an immediate return to NONMEM from PREDPP with a nonzero PRED error return code and accompanying user message. The contents of GG are ignored.

Global output variables IERPRD, NETEXT, ETEXT are used.

†Other possible options are $THETA NOABORTFIRST, and $EST NOHABORT. These are discussed in Guide VIII.

USE NMPRD_INT , ONLY : **IERPRD , NETEXT**

USE NMPRD_CHAR , ONLY : **ETEXT**

If a user message is returned, it will appear as part of the PRED error message. The return code is stored in IERPRD. Values are:
IERPRD=0: Normal return
IERPRD=1: PRED is unable to compute. NONMEM should attempt recovery.
IERPRD=2: PRED is unable to compute. NONMEM should abort the run.

The user message can be comprised of up to three character strings, each of length 132. The message is stored in a character string array ETEXT. NETEXT gives the number of lines of text stored by PK in ETEXT.

When NM-TRAN is used, the EXIT statement may be used in abbreviated code. For example, a statement such as the following may be present in $PK:

```
EXIT 1 2
```

This sets IERPRD=1 and
ETEXT(1)='PK SUBROUTINE: USER ERROR CODE = 2'
followed by a return from PK.

USE NMPRD_INT , ONLY : **IQUIT**

Another global output variable, IQUIT, is used within PREDPP so that any subroutine may initiate an error return to NONMEM. When NM-TRAN abbreviated code is used for $PK (or $ERROR), IQUIT is tested after a call to GETETA, SIMETA, or SIMEPS (See Chapter IV). A user-written code should contain similar code, e.g.

```
CALL GETETA(ETA)
IF (IQUIT == 1) RETURN
```

L. Examples

L.1 Example I: Population Data

One example of code for a PK routine involves the simple one compartment linear model. It is based on an analysis of phenobarbitol data (which may be found on a file of the NONMEM distribution medium; see Guide III). More detailed discussion of the data set is to be found in section V.L.1. Essentially, each of 59 neonates was given some loading dose of the drug, had a plasma drug level measured about 2 hours later, was given maintenance doses about every 12 hours thereafter, had possibly one trough level measured during maintenance dosing, and certainly one level measured some time after the last maintenance dose. Due to the long half-life of the drug, for the purposes of analysis all doses can be regarded as intravenous bolus doses. The weight and APGAR score of each individual are available as concomitant information. The APGAR score (1-10) is a measure of a neonate's health at birth.

The one compartment linear model is implemented by choosing subroutine ADVAN1 from the PREDPP Library (see chapter I). The user chooses to parameterize this model in Cl and Vd, so the routine TRANS2 is also chosen from the Library (see section A). Cl is modeled as in (14), with $\tilde{Cl}$ taken to be proportional to weight as in (2) (see section D). Vd is also modeled as in (14), with $\tilde{Vd}$ taken to be proportional to weight as in (2); however the proportionality constant is one of two possibly different values depending on the APGAR score. If APGAR is 2 or less, the proportionality constant is one value, and if APGAR is greater than 2, the proportionality constant is possibly another value.

A code for PK is given in Figure 1. (A corresponding NM-TRAN abbreviated code is shown in Figure 17 as part of an NM-TRAN control stream.) It returns typical values and typical first-partial derivatives. It can be used with the first-order estimation method, but not with conditional estimation methods or with posthoc estimation of η's. Note that the row index of the scaling parameter for the plasma (i.e. central) compartment is set at ICALL=1 to be 3 and that this parameter is identified with Vd. Also note that since both weight and APGAR do not vary within an individual over time, it is stipulated at ICALL=1 that at ICALL=2, PK be called only once per individual record.

A second code for PK is given in Figure 2. (A corresponding NM-TRAN abbreviated code is shown in Figure 17a as part of an NM-TRAN control stream.)† Figure 17a differs from Figure 17 in several ways.

•The $INPUT record includes an extra data item, APFL.
•The $PK block adds the seventh data item to the data set by assigning values to APFL during simulation.
•The $PK block had limited the calls to once per individual record by the statement `CALLFL=1` This has been removed so that PK is called with every event record. This allows APFL to be appended to every data record. Similarly, Figure 2 no longer contains
```
IDEF(1,2)=1
```
•The record
```
$SIML (1111)
```
is added to instruct NONMEM to implement the Simulation Step.
•The first $SCATTER record shows how APFL can be used to partition a scatterplot.

Figure 2 is shown to illustrate the use of PK with the Simulation Step. Upon inspecting it, one should imagine that drug levels are being simulated under the same model as decribed above, and with the same dosing and observation designs and same subject-specific values for the covariables weight and APGAR score as pertain to the real drug level data. One should further imagine that these simulated data are then analyzed as described above. The code at ICALL=2 is exactly like that of Figure 1. At ICALL=4 values for the η variables are obtained using the NONMEM utility SIMETA (see Guide IV, section III.B.13), subject-specific values of the PK parameters are computed under the given model, but η-derivatives are not computed.

It is, of course, possible to simulate data using one model for the PK parameters and to analyze these data using another by allowing the codes at ICALL=2 and ICALL=4 to implement different models. Structural kinetic differences between simulation and analysis models may be accommodated using an ADVAN implementing one of the general linear or nonlinear models (see chapter VI), or by simulating data in one NONMEM run with one model and analyzing them in another NONMEM run with another model.

One also sees in Figure 2 that the 7th data item of each data record is modified to be 1 or 2 according as the APGAR score is or is not, respectively, greater than 2. This indicator type data item might serve to partition scatterplots. Modification of input data items is called <u>transgeneration.</u> In general, data items (not required by NONMEM or PREDPP) can be modified ("transgenerated") at ICALL=4 (but not at ICALL=2; see also section VI.A). However when using PREDPP, only data items in the last data record of an event record can be modified; modifications of data items in data records other than the last data record are simply ignored.

A third code is given in Figure 3. It returns subject-specific values and subject-specific first-partial derivatives (but is not meant to be used with simulation). It can be used with conditional estimation methods and with posthoc estimation of the η's, in particular, but also with the first-Order estimation method. (The NM-TRAN abbreviated code shown in Figure 17 would work just as well for this third code as for the first code. However, yet another NM-TRAN control stream with this same abbreviated code, but explicitly requesting the computation of posthoc estimates, is given in Figure 21.)

†With previous versions of this guide, Figure 17 was used. However, Figure 17 was not appropriate.

L.2 Example II: A Mixture Model

With this example, the data set of example I is again used, and again, the kinetic model is the one compartment linear model parameterized in Cl and Vd, and Cl and Vd are modeled as in (14). However, weight is now ignored, and $\tilde{\mathrm{C}}\mathrm{l}$ and $\tilde{\mathrm{V}}\mathrm{d}$ are modeled as in (1), i.e. taken simply to be elements of θ. With no further changes to example I, the fit is poor, as seen in the scatterplot of CP (measured plasma concentration) vs PRED in Figure 4. If we pretend that weight has not even been recorded, there is little that can done to substantially improve the fit by introducing concomitant variables into the model; the APGAR score is not a very important covariable. We can try to include a correlation between the two η-variables, and this reduces the minimum value of the objective function about 15 points. (This is partially explained by the fact that weight has a considerable effect on both CL and Vd.) However, the scatterplot of CP vs PRED in Figure 4 actually results from a run with which the correlation is already included.

There is, though, some reason to entertain the idea (see below) that the 59 neonates constitute a random sample from a population comprised of two subpopulations, one with one set of typical values of Cl and Vd, and a second with another set of typical values, even though nothing is recorded about a neonate that would indicate of which subpopulation he is a member. A mixture model, a model that explicitly assumes that some fraction p of the population has one set of typical values of CL and Vd, and that the remaining fraction 1-p has another set of typical values, describes such a population and may be fit to the data. Both sets of typical values and the mixing fraction p may be estimated. This example shows how to implement such a model.

If it were known that CL and Vd were both proportional to weight (and we might justifiably suppose this much is known) and that weight were bimodally distributed in the population, this would clearly justify the mixture model assumption. However, from the recorded weights there is no evidence that weight is bimodally distributed. There is evidence, though, that weight is very right-skewed, and the assumed mixture model can also describe a *unimodel* right-skewed distribution, bivariate in CL and Vd (although, other types of models can also describe the same kind of distribution). It is for this reason that such a model might be tried.

The fit with the mixture model does indeed result in such a distribution (see section V.L.2). The minimum value of the objective function is reduced by another 15 points. Also, it is interesting to see how retrospectively, weight can be related to a classification of the 59 neonates into two groups as suggested by the fitted mixture model.

The mixture model is given by:

For subpopulation 1:

$$\mathrm{CL} = \theta_1 \exp \eta_1$$

$$\mathrm{Vd} = \theta_3 \exp \eta_2$$

For subpopulation 2:

$$\mathrm{CL} = \theta_2 \theta_1 \exp \eta_3$$

$$\mathrm{Vd} = \theta_4 \theta_3 \exp \eta_4$$

The parameters θ_2 and θ_4 are the fractional differences in the typical values between the two subpopulations. With a mixture model, different η variables on the same PK parameter must be used with different subpopulations. Certainly, when the magnitude of the dispersion of the parameter within a subpopulation differs between subpopulations, it should be clear that different η variables must be used, but this rule actually applies even when the magnitudes are the same.

A code is given in Figure 5, returning subject-specific values and subject-specific first-partial derivatives. (A corresponding NM-TRAN abbreviated code is shown in Figure 23 as part of an NM-TRAN control stream. Note that, with this version of Guide VI, $PK abbreviated code uses an IF/THEN/ELSE structure similar to Figure 5 rather than an indicator variable Q to select the model. This is a matter of style.)

The integer variable MIXNUM, found in module ROCM_INT, has value 1 or 2 according to whether NONMEM requires PRED to compute outputs for the 1st or 2nd subpopulations, respectively. Accordingly, PK computes different outputs according to the value of MIXNUM. MIXNUM first assumes the value 1, and then PK is called with event records from an individual record. Then MIXNUM assumes the value 2, and PK is called with the event records from the same individual record. This process is carried out for each individual record, and, of course, for each individual record repeatedly as parameter values vary.

For each individual, NONMEM computes an estimate of the number of the subpopulation of which the individual is a member, and it stores this estimate in the integer variable MIXEST, also found in ROCM_INT. When PK is called with any event record from an individual's record, the value of MIXEST is the estimate for the individual, although, this value is 1 for all calls to PK preceding the computation of the estimates. By the time the computations for the Table and Scatterplot Steps are performed, these estimates are actually available, and storing them in a variable EST, listed in module NMPRD4, during calls to PK while these steps are being implemented, enables the estimates to be displayed in tables and scatterplots. The control stream requesting the display is described in section V.L.2.

A code for the user-supplied routine MIX is given in Figure 6. (A corresponding NM-TRAN $MIX abbreviated code is shown in Figure 23 as part of an NM-TRAN control stream.) This routine is required when a mixture model is used. It is called with one individual record after another. Each time it is called the fractions p and 1-p are computed and stored in the array P. In this example p is simply THETA(5) (and is estimated, but constrained to be between 0 and 1; see section V.L.2). The number of subpopulations is stored in the variable NSPOP (in principle, this number, as well as the fractions associated with each subpopulation, can change from individual to individual). The current value of THETA is found in module ROCM_REAL, and if required, data items DATA from the observation records of the individual record can be found in ROCM_REAL, as shown in Figure 6.

L.3 Example III: Single-Subject Data

Another example of code for a PK routine involves the one compartment linear model with first order absorption. It is based on the very same example used in Guide I, section C - the simple nonlinear regression example using theophylline data. Although PREDPP has been designed with population pharmacokinetic data analysis in mind (where there are data from a number of subjects), the user can also take advantage of the ability with PREDPP to facilitate pharmacokinetic computations when the data come from a single subject. The simple nonlinear regression example illustrates how to use PREDPP with such data. In this example a single bolus dose is given to the subject, and then plasma concentrations are observed. Therefore, this example, though typical of single-subject data, does not really illustrate the power of PREDPP. It is simple enough that a *user-supplied* PRED, such as the one given in Guide I, can be developed rather quickly by most users. Were multiple dosing involved, or a more complicated kinetic model used, the advantage in using PREDPP would be clearer.

The code for PK is given in Figure 7. (A corresponding NM-TRAN abbreviated code is shown in Figure 26 as part of an NM-TRAN control stream.) The user-chosen parameters are the rate constants of elimination and absorption. Since all the data are from a single subject, random effects describing subject-to-subject variability are unnecessary. So, η type random variables do not appear in the model for the PK parameters, and neither ETA variables nor η-derivatives are computed in the PK routine. The typical values of the PK parameters are just the subject's values of these parameters. In principle these could be modeled in terms of concomitant variables, e.g. the elimination rate constant might be modeled in terms of rapidly changing serum creatinine. However, in the data situation at hand, only one dose is given, and no concomitant variables other than dose and time are available. So the subject's PK parameters are simply elements of θ, the correspondence being exactly the same as that given in the example in Guide I, section C. Namely, the rate constants of absorption and elimination are θ_1 and θ_2, respectively. As in Example I (section L.1), the scaling parameter of the plasma compartment is identified with Vd, or θ_3.

Since every call to PK with a different event record results in the same output from PK (for fixed THETA), only one call is necessary. Therefore, IDEF(1,2) is set to 1, thereby limiting calls to PK at ICALL=2 to one per data set; see section H.

When the data are from a single subject, and the run involves data simulation, where the simulation and analysis models for PK parameters are the same, PK code at ICALL=4 can be exactly the same as that at ICALL=2. In the example the code in Figure 3 would work just as well when simulation only is implemented, or when estimation only is implemented, or when both simulation and estimation are implemented. When the simulation and analysis models for PK parameters differ, then codes at ICALL=2 and ICALL=4 will simply differ. Structural kinetic differences between simulation and analysis models may be accommodated using an ADVAN implementing one of the general linear or nonlinear models (see chapter VI), or by simulating data in one NONMEM run with one model and analyzing them in another NONMEM run with another model.

L.4 Example IV: Single-Subject Pharmacodynamic Data

This example is similar to Example III, except that a pharmacodynamic observation is recorded at each of ten times, and a simple Emax type model, with the drug concentration being that of an effect compartment, is used. The same kinetic model from Example III is used, with the addition of an effect compartment attached to the central compartment. The purpose of this section is to describe the implementation of these kinetics. The pharmacodynamic model itself, and its implementation, are described in section IV.G.4. In order to implement the kinetics an ADVAN implementing a general linear model is used (although the ADVAN implementing the two compartment linear mammillary model with first order absorption could also be used for these particular kinetics). It is assumed that the PD observations are obtained from the same subject from which the PK observations in Example III are obtained, and that the values of the PK parameters (except for the elimination rate constant from the effect compartment) are those estimates obtained in Example III.

A code for PK is given in Figure 8. (A corresponding NM-TRAN abbreviated code is shown in Figure 30 as part of an NM-TRAN control stream.) As with Example III, compartments 1 and 2 are the depot and central compartments, respectively; compartment 3 is the effect compartment. When using a general linear model, the numbering of the compartments is specified in the user-supplied MODEL routine (see section VI.B). A dummy TRANS routine, TRANS1, that allows the rate constants to be modeled in PK, is the only TRANS routine (from the PREDD Library) that is available with a general linear model (see next section). The elimination rate constant from the effect compartment is to be estimated; it is identified with θ_1. The rate constant from the central compartment to the effect compartment is set to 0.1% of the elimination rate constant from the central compartment, so that the amount in the central compartment is unaffected by the kinetics of the effect compartment. Concentration in the effect compartment is not observed, so the volume in this compartment is not estimable. However, if the volume changes by a known scale factor s, the C_{50} parameter of the Emax model simply changes by s^{-1} . Therefore, the volume is arbitrary and is often obtained, as in this example, so to make the concentration the same as central compartment concentration at steady-state. When using a general linear model, the numbering of the rate constants is specified in the user-supplied MODEL routine (see section VI.B).

Were the data from a population of subjects, there would be interindividual variability in various parameters of the model, in particular in the kinetic parameters. The values of K12, K20, and VD could be set to subject-specific estimates obtained from analyzing PK data from each subject separately, as were the values of these parameters for the subject of this example. The values for a given subject could be included in his event records and then would be available to PK in EVTREC. This would account for the interindividual variability in these parameters. The elimination rate constant from the effect compartment could be modeled with an η-variable, accounting for random individual variability in this parameter. The same considerations regarding such modeling, which have been discussed in previous sections of this chapter and illustrated in sections L.1 and L.2, apply. Alternatively, K12, K20, and VD could themselves be modeled with η-variables. Then there are two possibilities. The θ's and Ω elements that are a part of the model for these parameters could be set to estimates obtained from analyzing PK data. Alternatively, they, along with the population parameters which are a part of the pharmacodynamic model, could be estimated by fitting both PK data and PD data simultaneously.

M. User-supplied TRANS

To compute the kinetic equations for a kinetic model, PREDPP - but more precisely, the chosen ADVAN - uses an internal set of parameters. These parameters may be modeled in PK, but a different set, one preferred by the user, can be modeled there instead. The TRANS subroutine translates (or transforms) the values for a set of basic PK parameters modeled in PK to a set of values for the internal parameters. The PREDPP Library has a number of TRANS subroutines, representing different possible parameterizations, from which the user may choose (see chapter VII). If a suitable translator is not found in the Library, the user may write his own. The form of such a subroutine is described in this section.

For a linear kinetic model, the internal parameters are the microconstants, i.e. rate constants, of the model. For a linear model, other than a general linear model, these parameters are numbered in the same way as the microconstants with TRANS1 from the PREDPP Library (see section VII.C). If the model is a general linear kinetic model, the MODEL subroutine specifies a numbering of these parameters (see section VI.B). For the simple Michaelis-Menton elimination model (see section VII.C.10), the two internal parameters are the same as the first two basic parameters listed for TRANS1, and are numbered the same. The kinetic equations for a general nonlinear kinetic model PREDPP essentially consist of the differential-algebraic equations given by the user in the DES and AES routines to describe the model (see sections VI.C and VI.E). The parameters of these equations comprise the internal set of parameters; they are numbered according to the numbering used in the DES and AES routines.

The preface to TRANS can be

```
SUBROUTINE TRANS(ITRANS,IRGG,GG,NETAS)
USE PRSIZES, ONLY: ISIZE,DPSIZE
USE PRDIMS,  ONLY: GTRD
INTEGER(KIND=ISIZE):: ITRANS,IRGG,NETAS
REAL(KIND=DPSIZE):: GG(IRGG,GTRD+1,GTRD+1)
```

For simplicity, the GG array is shown as two-dimensional in the discussion below. GTRD is equal to GPKD.

The argument ITRANS functions similarly to ICALL in PK. A value of 1 signals to TRANS that it is being called for the first time in the NONMEM problem. A value of 2 signals a regular data analytic call, and a value of 4 signals a regular data simulation call. At ITRANS=1, ITRANS must be reset by TRANS to a number in the range 1-8999. This number appears on NONMEM output, allowing the user to identify the TRANS routine being used. At ITRANS=2 and 4, TRANS must modify values in the GG array.

Suppose P denotes the vector of internal parameters. The user may choose to model instead a vector of parameters R. R may contain more elements than P. Let T be a transformation taking R onto P, i.e. the nth element of P is given by

$$P_n = T_n(R) \tag{46}$$

for some function T_n. E.g. P = (Ke) [ADVAN1]; R = (Cl, Vd) [TRANS2]; and $T_1(Cl, Vd) = Cl/Vd$. Then the typical value of the nth element of P is defined by

$$\tilde{P}_n = T_n(\tilde{R}). \tag{47}$$

The first derivative of the nth element of P with respect to η_k is

$$\frac{\partial P_n}{\partial \eta_k} = \sum_i \frac{\partial T_n}{\partial R_i}(\tilde{R})\,\frac{\partial R_i}{\partial \eta_k}(\eta). \tag{48}$$

The second-partial derivative of the nth element of P with respect to η_k,η_l is

$$\frac{\partial^2 P_n}{\partial \eta_k \partial \eta_l} = \sum_{i,j} \frac{\partial^2 T_n}{\partial R_j \partial R_i}(\tilde{R})\,\frac{\partial R_j}{\partial \eta_k}(\eta)\,\frac{\partial R_i}{\partial \eta_l}(\eta) + \sum_i \frac{\partial T_n}{\partial R_i}(\tilde{R})\,\frac{\partial^2 R_i}{\partial \eta_k \partial \eta_l}(\eta) \tag{49}$$

So in terms of the example:

$$\tilde{Ke} = \tilde{Cl}/\tilde{Vd} \tag{50}$$

$$\frac{\partial Ke}{\partial \eta_k} = \frac{1}{Vd}\,\frac{\partial Cl}{\partial \eta_k} - \frac{Cl}{Vd^2}\,\frac{\partial Vd}{\partial \eta_k} \tag{51}$$

$$\frac{\partial^2 Ke}{\partial \eta_k \partial \eta_l} = -\frac{1}{Vd^2}\,\frac{\partial Vd}{\partial \eta_k}\,\frac{\partial Cl}{\partial \eta_l} - \frac{1}{Vd^2}\,\frac{\partial Cl}{\partial \eta_k}\,\frac{\partial Vd}{\partial \eta_l} + 2\,\frac{Cl}{Vd^3}\,\frac{\partial Vd}{\partial \eta_k}\,\frac{\partial Vd}{\partial \eta_l} + \frac{1}{Vd}\,\frac{\partial^2 Cl}{\partial \eta_k \partial \eta_l} - \frac{Cl}{Vd^2}\,\frac{\partial^2 Vd}{\partial \eta_k \partial \eta_l}\,. \tag{52}$$

At ITRANS=2, on input the GG array has stored in it the values computed by PK (except that were any PK parameter modeled in its logarithm form in PK, PREDPP would have already exponentiated its typical/subject-specific value and multiplied its η-derivatives by its exponentiated typical/subject-specific value). On output the GG array should have stored in it the values that would be computed by PK were the internal parameters P modeled directly in PK (and none in their logarithmic form). So the code for TRANS2, using the above example, is essentially:

```
   GG(1,1) = GG(1,1)/GG(2,1)
   IF (ITRANS.EQ.2) THEN
   DO 10 K = 1,NETAS
10 GG(1,1+K) = GG(1,1+K)/GG(2,1)-GG(1,1)*GG(2,1+K)/GG(2,1)
   ENDIF
```

if only first derivatives are needed, and is:

```
A(1) = GG(1,1,1)/GG(2,1,1)
   IF (ITRANS.EQ.2) THEN
   DO 10 K = 1,NETAS
   A(1+K) = GG(1,1+K,1)/GG(2,1,1)
  1            -GG(1,1,1)*GG(2,1+K,1)/GG(2,1,1)**2
   DO 10 L = 1,K
10 GG(1,1+K,1+L) =
  1    -(GG(2,1+K,1)*GG(1,1+L,1)+GG(1,1+K,1)*GG(2,1+L,1)
  2    +GG(1,1,1)*GG(2,1+K,1+L))/GG(2,1,1)**2
  3    +2*GG(1,1,1)*GG(2,1+K,1)*GG(2,1+L,1)/GG(2,1,1)**3
  4    +GG(1,1+K,1+L)/GG(2,1,1)
   DO 20 K = 1,NETAS
20 GG(1,1+K,1)=A(1+K)
   ENDIF
   GG(1,1,1)=A(1)
```

if first and second derivatives are needed. Note that the total number of η variables used in the problem is given in the last argument of TRANS and can be used as illustrated here. Note also that A is a work array, not the state vector.

At ITRANS=4, on input the GG array has stored in its first column the values of the subject-specific PK parameters computed by PK (except that for each subject-specific PK parameter modeled in its logarithm form in PK, PREDPP has already exponentiated its value). On output the GG array should have stored in its first column the values that would be computed by PK were the internal parameters P modeled directly in PK (and none in their logarithmic form). Since, when ITRANS=4, only the first column of the GG array is relevant, computations involving other columns may be skipped, as in the above code.

TRANS has access to the entire GG array, including the typical/subject-specific values and η-derivatives of the additional PK parameters. It can therefore perform a translation for all parameters, including scaling parameters, bioavailability fractions, etc., not just for the basic PK parameters. This type of translation, however, is usually not needed.

See Section E.5 above for a discussion of the Active Eta List feature, which may optionally be used in a TRANS routine to improve run time.

N. Other Subroutines That May Be Called

RANDOM

The NONMEM utility routine RANDOM may be called by PK during the Simulation Step (ICALL=4) and when data averages are being computed (ICALL=5) to obtain numbers from different random sources.

IV. The ERROR routine

A. Modeling Residual Error

Suppose a set of values for an individual's PK parameters is fixed. Let P denote this set of values, and let f(P) denote the prediction under the kinetic model, based on P, for an observation y from the individual. This prediction is imperfect. Observations that one might imagine obtaining from the individual, under exactly the same conditions as accounted for in f(P), would differ from one another. This "residual variabilty" in observations, "unexplained by f(P)", derives from several sources. One source is random measurement error. Another important source is model misspecification error in f or in P. The terms 'residual variability' and 'residual error' are used here interchangeably. Residual error, therefore, does not generally refer to error from some one identified source. Residual error may be modeled in terms of concomitant variables and random effects. This modeling, along with the modeling of interindividual variability in P, leads to a full statistical model for y. In this section we describe the models for residual error, and in section B we discuss how these models can be implemented with PREDPP. The following discussion parallels that of section III.D.

The simplest model for y itself in terms of f(P) is

$$y = f(P) + \varepsilon \qquad (1)$$

where ε is the realization of a random variable (which also, through a mild abuse of notation, is denoted by ε) with mean 0. This is a very familiar type model describing residual variability.

The difference $y - f(P)$ may not be entirely unexplainable. For example, it might be that there are two assays used to measure drug concentration, and that the difference may vary more widely with y obtained with one assay than with y obtained with the other assay. Let Z be a dichotomous-valued concomitant variable whose value (0 or 1) in the event record for y identifies the assay used. Then instead of (1) one might write

$$y = f(P) + \theta_1^Z \varepsilon_1 \qquad (2)$$

where ε_1 has mean zero and variance σ^2. We have sd(y) = sd(y − f(P)), and if Z=0, sd(y) = σ, but if Z=1, sd(y) = $\theta_1\sigma$. The parameter θ_1 is the ratio of standard deviations of the concentrations between the two assays.

Another simple model for y is

$$y = f(P)(1 + \varepsilon) \qquad (3)$$

where the mean and variance of ε are 0 and σ^2. Here σ is the coefficient of variation of y. Instead of ε having homogeneous variance σ^2, perhaps $\varepsilon = \theta_1^Z\varepsilon_1$ as above. Observation values are often significantly right-skewed distributed (for a fixed x), and so a more appropriate model might be

$$y = f(P)\ \exp\ \varepsilon \qquad (4)$$

where the mean and variance of ε are 0 and σ^2, respectively. This model is equivalent to

$$\log\ y = \log\ f(P) + \varepsilon \qquad (5)$$

(See the discussion concerning models (III.14) and (III.15).)

The difference $y - f(P)$ could be partly explainable by the compartment being observed. If two compartments are being observed (say plasma and urine), and Z is a dichotomous-valued concomitant variable whose value (0 or 1) in the event record for y identifies the compartment observed, then one might write

$$y = f(P) + \varepsilon_1 \quad \text{if } Z = 0 \tag{6}$$

$$= f(P) + \varepsilon_2 \quad \text{if } Z = 1$$

where the means of both ε_1 and ε_2 are 0, and the variances are $\sigma^2{}_1$ and $\sigma^2{}_2$, respectively. If two observations of two different compartments are taken far enough apart in time, ε_1 and ε_2 can be taken to be statistically independent variables. In this case (6) is formally equivalent to (2) (ignoring whether Z identifies assay or observation compartment) since multiple realizations of ε_1 in (2) are assumed to be statistically independent. If, however, two observations of two different compartments are taken close enough in time, they may be correlated, and with NONMEM, it is possible to estimate a correlation between the two *different* ε variables of model (6); see Guide I, section E.4.

In all the above models for y, when all ε's are set to their mean value 0, we have y = f(P); residual error is 0. f(P) is the subject-specific prediction of y . We write $\tilde{P}$ for the typical value of P, the value with all η's set to 0. We write $\tilde{y}$ for the quantity $f(\tilde{P})$, the prediction "for the typical individual in the in the population" (of individuals with given values for the concomitant variables). It is the population prediction of y . It may also be regarded as the typical value of y .

The routine ERROR specifies the model for residual variability. It does so in different ways, depending on whether ERROR is being called for the purposes of data analysis or data simulation. For the purposes of data simulation, the specification closely follows the type of mathematical expressions shown above. As with the specification of PK parameters in routine PK, for the purposes of data analysis, the specification of the model for residual variability entails partial derivatives of different types. These different types are defined below. With the first-order estimation, or with a conditional estimation method where an $\eta - \varepsilon$ interaction is either absent or ignored, typical first-partial derivatives of y with respect to the ε's must be computed, although subject-specific first-partial derivatives will also suffice since from the latter the former can be computed. With the first-order conditional estimation method with interaction, where an $\eta - \varepsilon$ interaction is not ignored, both subject-specific first-partial derivatives of y with respect to the ε's, as well as mixed-second-partial derivatives, must be computed.

The first-partial derivatives of y with respect to the ε's, evaluated at the subject-specific value of P and at all ε's equal to 0, are called the subject-specific first-partial derivatives of y . For (1)-(4) and (6) for example, these derivatives are

$$(1):\ \frac{\partial y}{\partial \varepsilon} = 1 \tag{7}$$

$$(2):\ \frac{\partial y}{\partial \varepsilon_1} = \theta_1^Z \tag{8}$$

$$(3):\ \frac{\partial y}{\partial \varepsilon} = f(P) \tag{9}$$

$$(4):\ \frac{\partial y}{\partial \varepsilon} = f(P) \tag{10}$$

$$(6):\ \frac{\partial y}{\partial \varepsilon_1} = 1 \tag{11}$$

$$\frac{\partial y}{\partial \varepsilon_2} = 1 \ .$$

The first-partial derivatives of y with respect to the ε's, evaluated at the typical value of P, $\tilde{P}$, and at all ε's set equal to 0, are called the <u>typical first-partial derivatives of y</u> . The derivatives of examples (7), (8) and (11) are also typical first-partial derivatives, and the derivatives of examples (9) and (10) are typical first-partial derivatives when P takes the value $\tilde{P}$ in those expressions.

Note that the derivatives under (3) and (4) are identical. When the model for residual variability is specified via subject-specific or typical first-partial derivatives only, NONMEM estimation methods cannot distinguish between models (3) and (4). That is, the same fit will result from using either model. In effect, an assumption is being made that under (4), the variance of ε is small, and that the mean and coefficient of variation of y are well approximated by f(P) and σ, respectively. If, though, the DV data items are the log transformed observations, and these are modeled with (5), then this approximation is avoided, and a different fit will result from that obtained under (3). Yet a different fit again will be obtained when (4) is specified by including mixed-second-partial derivatives.

The <u>mixed-second-partial derivatives of y</u> are the derivatives of $\frac{\partial y}{\partial \varepsilon}$ (for the various ε's) with respect to the η's, expressed as functions of the η's and evaluated at all ε's equal to 0. Note that in general these are different from the derivatives with respect to the η's of the subject-specific first-partial derivatives of y, since first the derivative is taken with respect to η, and then the ε's are set to 0. However, in virtually all particular cases they will be the same. Some examples will illustrate this. The mixed-second-partial derivatives corresponding to (1)-(4) and (6) are

$$(1): \ \frac{\partial^2 y}{\partial \eta \partial \varepsilon} = 0 \tag{12}$$

$$(2): \ \frac{\partial^2 y}{\partial \eta \partial \varepsilon_1} = 0 \tag{13}$$

$$(3): \ \frac{\partial^2 y}{\partial \eta \partial \varepsilon} = \frac{\partial f(P)}{\partial \eta} \ . \tag{14}$$

$$(4): \ \frac{\partial^2 y}{\partial \eta \partial \varepsilon} = \frac{\partial f(P)}{\partial \eta} \tag{15}$$

$$(6): \ \frac{\partial^2 y}{\partial \eta \partial \varepsilon_1} = 0 \tag{16}$$

$$\frac{\partial^2 y}{\partial \eta \partial \varepsilon_2} = 0.$$

It should be emphasized that f(P) (and f($\tilde{P}$)) and $\frac{\partial f(P)}{\partial \eta}$ are not quantities that the user is expected to compute in order to compute in turn derivatives such as those given in (9) and (14); this is PREDPP's job. One of the arguments of the routine ERROR is f(P) (f($\tilde{P}$)), and the user can make use of it, incorporating it into partial derivatives of y. Similarly, $\frac{\partial f(P)}{\partial \eta}$ is an argument of ERROR.

When some mixed-second-partial derivative is not zero, there is said to be an $\eta - \varepsilon$ interaction. Such an interaction is taken into account only by the first-order conditional estimation method with interaction, and with this method the mixed-second-partial derivatives must be computed.

Routine ERROR allows specification of the derivatives of log y, rather than of y (presumably, the DV data items are untransformed data items). That is, the derivatives of log y, rather than the derivatives of y itself, may be specified. If log y is given by (5), for example, then

$$(5):\ \frac{\partial \log\ y}{\partial \varepsilon} = 1 \tag{17}$$

$$(5):\ \frac{\partial^2 \log\ y}{\partial \eta \partial \varepsilon} = 0\ . \tag{18}$$

PREDPP transforms $\frac{\partial \log\ y}{\partial \varepsilon}$ to $\frac{\partial y}{\partial \varepsilon} = f(P)\,\frac{\partial \log\ y}{\partial \varepsilon}$, since it needs the latter. The mixed-second-partial derivatives of log y are also appropriately transformed.

When the data are from a single-subject, random subject-to-subject variability does not occur, and in this case random effects describing such variability do not appear in the model. Only random effects describing residual variablity appear. Through information in the NONMEM control stream NONMEM learns that only one of these two possible types of random effects appears, but not which type. This, in turn, is communicated to PREDPP, which then *assumes* that the only random effects appearing in the model are those appearing in the model for residual variability and that the data do indeed come from a single subject. This assumption has various natural and helpful consequences. To enable the reader to identify these, they are explicitly described in this document as being consequences of the single-subject assumption . See sections B.1, B.2, III.H, V.H, and V.K.

Starting with NONMEM 7.3, there is an alternate approach. See "Single-Subject Analysis using Population with Unconstrained ETAs" in the Introduction to NONMEM 7. With this approach, there are multiple subjects in the data set. NONMEM and PREDPP treat the subjects as part of a population analysis, while the statistical algorithms treat the data as single-subject.

B. Implementation in ERROR

Specification of the model for residual variability is done with the required user-supplied ERROR subroutine. This is described in section B.1. In the discussion of section A f(P) denotes the prediction under the kinetic model for a pharmacokinetic observation, and the residual variability is that of such an observation. However, in that discussion f(P) could just as well have denoted a prediction under a pharmacodynamic model, and the residual variability could be that of a pharmacodynamic observation. The way a model for residual variability for a PD observation is specified is essentially the same way this is done for a PK observation. However, there are some further considerations, such as how to obtain a PD prediction. These are discussed in section B.2.

B.1 Implementation of the Model for Residual Error

The preface of the ERROR routine must be

```
SUBROUTINE ERROR (ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,F,G,HH)
USE SIZES,     ONLY: DPSIZE,ISIZE
USE PRDIMS,    ONLY: GERD,HERD
IMPLICIT REAL(KIND=DPSIZE) (A-Z)
REAL(KIND=DPSIZE) :: EVTREC
INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS
DIMENSION :: IDEF(*),THETA(*),EVTREC(IREV,*),INDXS(*)
REAL(KIND=DPSIZE) :: G(GERD,*),HH(HERD,*)
```

This is the NONMEM 7 version. The preface was different with earlier versions of NONMEM. Global variables GERD and HERD are needed because G and HH are sized according to the number of etas in the problem. G may be declared as a 1 dimensional array when the Laplacian method is not used `G(GERD)` and HH may be declared as a 1 dimensional array `HH(*)` when the first-order conditional estimation method with interaction is not used. For simplicity, they will be used that way in the examples below, although, when G or HH are declared 2 dimensional, the second subscript should be understood to be ",1"; E.g., HH(1) should be understood to be HH(1,1). However, when one or both of these methods might be used later with the given data set, it is a good idea to develop an ERROR code that allows this.

When ERROR is called by PREDPP, it is passed values for the vector θ in THETA. It is also passed a complete event record in EVTREC, the argument record. Specifically, EVTREC(I,J) contains the Jth data item of the Ith data record of the event record. ERROR is also passed the total number N of data records comprising the event record. Typically N=1, and so the first subscript of EVTREC will always be 1; however, see chapter II. With NM-TRAN, the CONT data item cannot be used and N is 1.

The argument ICALL functions similarly to the ICALL argument described in section III.C. It has several possible values when ERROR is called. The value 1 signals to ERROR that the routine is being called for the first time in the NONMEM problem. At such a time ERROR can store certain initializing information in IDEF (see below). The value 2 signals to ERROR that the routine is being called in a regular fashion for data analytic purposes and that the subject-specific/typical first-partial derivatives, and if necessary, the mixed-second-partial derivatives, are to be stored in HH. The value 4 is used in conjunction with the NONMEM Simulation Step. It signals to ERROR that the routine is being called for data simulation purposes and that the simulated value for y is to be stored in F.

The value 5 signals to ERROR that the routine is being called in a regular fashion when expectations are being computed; multiple calls occur. Expectation blocks are described in the help Guide VIII. No eta derivatives need be computed.

The value 6 signals to ERROR that the routine is being called in a regular fashion when raw data averages are being computed; multiple calls occur. Raw data average blocks are described in the help Guide VIII. No eta derivatives need be computed.

If there is abbreviated code in the $ERROR block that tests for ICALL=0, ICALL=1, or ICALL=3, this code is moved by NM-TRAN to the INFN routine as if it had been coded explicitly as part of an $INFN block. Such code is called $ERROR-INFN code. The initialization code described for IDEF below is generated in FSUBS by NM-TRAN regardless of the presence of $ERROR-INFN code.

At ICALL=2, derivatives of y with respect to the ε's must be computed and stored in HH. With the first-order estimation method, or with a conditional estimation method where an $\eta - \varepsilon$ interaction is either absent or ignored, the *typical* first-partial derivative of y with respect to ε_L is placed in HH(L) (or, if HH is declared to be 2-dimensional, HH(L,1)). For this purpose, the ε's are enumerated as are their variances in the specification of the initial estimate of Σ. For models (1-4) and (6) of section A, we could have the code

```
(1):  HH(1) = 1.
```

```
(2):  IF(Z.EQ.0.)  THEN
          HH(1) = 1.
      ELSE
          HH(1) = THETA(1)
      ENDIF

(3):  HH(1) = F

(4):  HH(1) = F

(6):  HH(1) = 1-Z
      HH(2) = Z
```

F is the 8th argument of ERROR. On input to ERROR it is the value $f(\tilde{P})$ or f(P) under the kinetic model, whichever prediction (typical or subject-specific) has been computed by PREDPP. Note that 0's must be explicitly stored in elements of HH as needed; the HH array is *not* initialized to 0 immediately before a call to ERROR. However, HH is initialized to 0 once, early in the problem, so that if whenever ERROR is called, an element of HH would always be set to 0, this never actually need be done in ERROR.

With the first-order conditional estimation method with interaction the subject-specific first-partial derivative of y with respect to ε_L is placed in HH(L,1), and the mixed-second-partial derivative of y with respect to η_K and ε_L is placed in HH(L,K+1). For models (1-4) and (6) of section A,

```
(1):  HH(1,1) = 1.
      HH(1,K+1) = 0.

(2):  IF(Z.EQ.0.)  THEN
          HH(1,1) = 1.
      ELSE
          HH(1) = THETA(1)
      ENDIF
      HH(1,K+1) = 0.

(3):  HH(1,1) = F
      HH(1,K+1) = G(K)   (or G(K,1))

(4):  HH(1,1) = F
      HH(1,K+1) = G(K)   (or G(K,1))

(6)   HH(1,1) = F
      HH(1,K+1) = 0.
      HH(2,K+1) = 0.
```

where actually, HH(1,K+1) and HH(2,KK+1), wherever they appear, must be set for all K from 1 to the number of η's in the problem.

G is the 9th argument of ERROR. On input to ERROR G(K) (or, if G is declared to be 2-dimensional, G(K,1)) is the value $\frac{\partial f(P)}{\partial \eta_K}(0)$ or $\frac{\partial f(P)}{\partial \eta_K}(\eta)$ under the kinetic model, whichever first-partial derivative (typical or subject-specific) has been computed by PREDPP. Note that zeros may need to be explicitly stored in elements of HH; the HH array is *not* initialized to zero immediately before a call to ERROR. However, it is initialized once, early in the run, so that if a mixed partial derivative is always 0, the corresponding element of HH need not be set to 0.

ICALL=4 signals that ERROR is being called during the Simulation Step. During the Simulation Step, values for ε variables (and η variables; see section B.2) can be simulated in ERROR using the NONMEM utility routine SIMEPS (and SIMETA; see section B.2). This in turn allows a value for the observation y to be computed in a direct fashion, using an expression such as (1) in section A. The value for y should be returned by ERROR in the argument F. This necessitates modifying the value of F input to ERROR. Values returned in HH are ignored. For an example, see section G.1.

With data from a single subject, as with population data, during the Simulation Step, values for y can be computed in ERROR in a direct fashion. However, NONMEM explicitly recognizes two types of random variables, η-variables and ε-variables, and these two types are nested, i.e. for any set of fixed values for the η-variables, the ε-variables can assume different values, but not conversely. *Whenever* nested variables are not needed, only one of these two types of variables need appear in the entire statistical model, and by NONMEM convention, these should be η-variables. With single-subject data there is no subject-to-subject variability, only nonnested random variables appear; they appear in the model for residual variability. Consequently, in this context the expressions in section A must be understood to have η-variables occuring wherever ε-variables occur. It does not matter what FORTRAN variable is used in the ERROR code to represent an η variable; it could be ETA, or EPS, or anything else. However, to obtain simulated values using a NONMEM utility routine, the routine SIMETA, rather than SIMEPS, must be called. See an example in section G.3. During data analysis computations, i.e. when values returned in HH are not ignored, PREDPP understands that HH contains η-derivatives (as a consequence of the single-subject assumption; see section IV.A).

The IDEF array is used at ICALL=1. Usually, a 0 should be stored in IDEF(1), indicating that the user acknowledges that when ICALL=2, the derivatives of y with respect to the ε-variables (or with single-subject data, the η-variables) are to be found in HH. In fact, when ICALL=1, PREDPP has initialized the IDEF array to zero, so if the user stores nothing in IDEF(1), the same effect is achieved. If, though, a 1 is stored in IDEF(1), the user is specifying that when ICALL=2, the derivatives of log y are to be found in HH. For example, if log y is given by (5), then at ICALL=2 one needs

```
(5):  HH(1) = 1.
```

If the first-order estimation method with interaction is used, then one needs

```
(5):  HH(1,1) = 1.
      HH(1,K+1) = 0.
```

A reason for setting IDEF(1)=1 is discussed in section C.

If the observation is that of a scaled drug amount, then ERROR does not need to change the value of F from the input value (i.e. at ICALL=2); this value is a scaled drug amount and serves as an appropriate prediction. There are, though, situations where the observation is not that of a scaled drug amount and where ERROR would need to modify the input value of F to a prediction that is appropriate for the observation (see section B.2). One might expect PREDPP to understand that when IDEF(1)=1, whatever value of F is returned by ERROR, it is the *logarithm of the prediction* for the observation, and that PREDPP exponentiates this value. (After all, with routine PK when GG(M,1)=1 at ICALL=1, the typical/subject-specific value of the logarithm of the Mth PK parameter is returned by PK in GG(M,1).)†

† The logarithms of PK parameters cannot be modelled in this way with NM-TRAN.

However, the value returned by ERROR is always understood to be the *prediction* for the observation.‡

The value in IDEF(3) describes whether ERROR uses derivatives of compartment amounts (i.e. whether compartment amounts themselves are used as random variables in arithmetic statements in ERROR).

IDEF(3)=-1 ERROR may use derivatives of A (this is the default)
IDEF(3)= 0 ERROR does not use derivatives A.
IDEF(3)= 1 ERROR does use derivatives of A.

When ERROR does not use A, then if IDEF(3) is set to 0, PREDPP can avoid some time-consuming processing.

The one-dimensional array, INDXS, functions in the way described in Guide I, section C.4.1. The user places integers into this array, using the INDEX control record. These integers are then available to PREDPP and, therefore, to ERROR. For further details see section III.C.‡

B.2 Implementation of a Pharmacodynamic Model

The ERROR routine has been described as the place where a model for residual error is specified. More generally, it may be described as the place where the prediction is specified, along with a model for the residual error. The prediction is taken to be the value of F output by ERROR. Often, the value of F that is input to ERROR is simply left unaltered by the routine, and so it is also the value output. That is, the prediction is specified to be that already computed by PREDPP. This is a typical way to proceed when the observation y is a pharmacokinetic response, and its prediction can be taken as the scaled drug amount found in the input value of F. However, when y is a pharmacodynamic response, this input value must be altered to obtain a prediction appropriate for y.

To take an example, if the PD prediction is proportional to the PK prediction, where the proportionality constant θ_4 is to be estimated, the essential code might be:

```
   F=THETA(4)*F
   HH(1)=F
```

The model for the residual error between y and its prediction might be given by (3) or (4). The way to implement this model is the same as when the response is PK, rather than PD, and is illustrated by this example. Care must be taken that θ_4 and any parameters to be estimated in the model for the scaling parameter used to compute the input value of F are all identifiable.

If the data are population data, and the model for PK parameters involves η-variables, then the first-partial derivative $\frac{\partial f(P)}{\partial \eta_K}(0)$ or $\frac{\partial f(P)}{\partial \eta_K}(\eta)$, whichever derivative (typical or subject-specific) has been computed by PREDPP and is input to ERROR, must *also be recomputed*. Recall that the derivative is found in G(K) (or G(K,1)). Therefore, if there are 3 η-variables, the above example might be continued thusly:

```
   F=THETA(4)*F
   HH(1)=F
   DO 10 K=1,3
10 G(K)=THETA(4)*G(K)
```

If the Laplacian method is used, the second-partial derivative $\frac{\partial^2 f(P)}{\partial \eta_L \partial \eta_K}(\eta)$, $L \le K$, is stored in G(K,L+1).

‡ An exception is with non-continuous odd-type data, below.

‡ The INDXS array cannot be used with NM-TRAN.

In this case the code might look like:

```
   F=THETA(4)*F
   HH(1)=F
   DO 10 K=1,3
   G(K,1)=THETA(4)*G(K,1)
   DO 10 L=1,K
10 G(K,L+1)=THETA(4)*G(K,L+1)
```

The second-partial derivatives are not needed with every call to ERROR; certainly, they are not needed unless the Laplacian method is used. In order to save computation time, information is provided in the NONMEM global variable MSEC as to whether second-partial derivatives are needed with a particular call to ERROR. MSEC is set by NONMEM to the value 1 or 0, according as the second-partial derivatives are needed or not. Consequently, an alternative code to the above might be:

```
   F=THETA(4)*F
   HH(1)=F
   DO 10 K=1,3
   G(K,1)=THETA(4)*G(K,1)
   IF (MSEC.EQ.1) THEN
      DO 5 L=1,K
 5    G(K,L+1)=THETA(4)*G(K,L+1)
   ENDIF
10 CONTINUE
```

If the first-order conditional estimation method with interaction is used, the code might look like:

```
   F=THETA(4)*F
   HH(1,1)=F
   DO 10 K=1,3
10 HH(1,K+1)=THETA(4)*G(K)
```

In the above example, the various codes are suitable whether typical or subject-specific derivatives are required. This is because the values returned in G and HH depend on the η's only through the values of F and G that are input to ERROR. If typical derivatives are required, the input values of F and G are computed with all η's equal to 0; if subject-specific values are required, they are computed with other values of the η's as well.

Suppose, however, that the PD prediction is proportional to the PK prediction, where the proportionality constant itself involves an η-variable, e.g. $\theta_4 \exp \eta_4$. Then the code with only typical first-partial derivatives might look like:

```
   F=THETA(4)*F
   HH(1)=F
   DO 10 K=1,3
10 G(K)=THETA(4)*G(K)
   G(4)=F
```

while the code with subject-specific first-partial derivatives might look like:

```
   USE NMPRD_REAL,ONLY: ETA
   ...
   CALL GETETA (ETA)
   E4=EXP(ETA(4))
   F=THETA(4)*E4*F
   HH(1)=F
   DO 10 K=1,3
10 G(K)=THETA(4)*E4*G(K)
   G(4)=F
```

Notice that GETETA can be called by ERROR as well as by PK (see section III.E). The ETA array is in a global module. GETETA can be called only by ERROR if η-variables are only used in ERROR. In this case GETETA may still be initialized by PK (see section III.E.2), but if it is not, then it should be initialized by ERROR. This involves simply calling GETETA at ICALL=1.

If the Laplace method is used, then the code might be:

```
   USE NMPRD_REAL,ONLY: ETA
   ...
   CALL GETETA (ETA)
   E4=EXP(ETA(4))
   F=THETA(4)*E4*F
   HH(1)=F
   DO 10 K=1,3
   G(K,1)=THETA(4)*E4*G(K,1)
   DO 10 L=1,K
10 G(K,L+1)=THETA(4)*E4*G(K,L+1)
   G(4,1)=F
   DO 15 L=1,4
15 G(4,L+1)=G(L,1)
```

If the first-order conditional estimate method is used with interaction, then the code might be:

```
   USE NMPRD_REAL,ONLY: ETA
   ...
   CALL GETETA (ETA)
   E4=EXP(ETA(4))
   F=THETA(4)*E4*F
   HH(1)=F
   DO 10 K=1,3
   G(K)=THETA(4)*E4*G(K)
10 HH(1,K+1)=G(K)
   G(4,1)=F
   HH(1,5)=F
```

During the Simulation Step, i.e. at ICALL=4, just as it is not necessary to store derivatives in HH, it is not necessary to store derivatives in in G. When the data are population data, ERROR can call the NONMEM utility SIMEPS to obtain the values for $\varepsilon_1, \varepsilon_2, \cdots$, but also it can call SIMETA to obtain values for $\eta_1, \eta_2, \cdots$ (see section III.E.2). By default, as long as ERROR (or PK) is being called with an event record from the same individual record, each time SIMETA is called, these values remain the same; there is only one set of values obtained for the individual.

However, the simulation can be done in such a way that the values change each time SIMETA is called (see Guide IV, section III.B.13). Consider this situation, and suppose that ERROR calls SIMETA, but PK does not. Then only the first time ERROR itself is called with an event record of a given individual record should ERROR call SIMETA (see section C for a discussion about the sequence of calls to ERROR). This assures that there is only one set of values obtained for the individual, as in the default situation. Unlike that situation, though, during this first call to ERROR, multiple calls to SIMETA might occur. So for example, simulated values of η_1, obtained with multiple calls to SIMETA and such that $|\eta_1| \geq 2$, can be rejected until a value < 2 is obtained, i.e. the distribution on η_1 can be truncated. If PK does call SIMETA, calls to SIMETA by ERROR can be avoided. PK can obtain the values of ETA elements for the individual by a series of calls to SIMETA (this is illustrated explicitly in section III.E.2) and store them (i.e. list ETA) in a FORTRAN module shared with ERROR. Indeed, in this case ERROR must obtain the values this way; for if SIMETA is called by ERROR, the values so obtained are different from the values obtained by PK.

When the data are single-subject data, SIMETA should be called, rather than SIMEPS (see discussion in section B.1).

Consider one further example where it may be advantageous to recompute F in ERROR. Suppose the data were changed from observations y to log y values. Suppose too that a reasonable model for the original data is given by (4), so that a reasonable model for the new data is given by (5). A suitable code would be:

```
   A=F
   F=LOG(F)
   HH(1)=1
   DO 10 K=1,3
10 G(K)=G(K)/A
```

(assuming three η-variables in the model for the PK parameters). Note that A is a work array, not the state vector in PROCM_REAL (see section D.A). If there are no η-variables in this model, G need not be recomputed. The effect of this code with the new data is different from the effect of

```
   IF (ICALL.EQ.1) IDEF(1)=1
   HH(1)=1
```

with the original data. Whereas the first code implements (4,5) exactly, this second code does not; see section A.

The value for F and the values for the G array returned to NONMEM by PREDPP (see Guide I) are exactly those values returned to PREDPP by ERROR in the arguments with the same names. The values returned by ERROR in HH to PREDPP (after possible "exponentiation" by PREDPP; see section B.1) are just those values returned by PREDPP in H to NONMEM. So, by taking advantage of the ability to *recompute* F and G in ERROR, as well as to compute H, PREDPP becomes in effect a PRED of the most general kind. Any regression problem that can be handled by NONMEM, can in principle be handled via PREDPP. It would not generally be efficient, though, to use PREDPP unless the value in F (and possibly G) input to ERROR is in fact needed.

With single-subject data the values returned by ERROR in HH to PREDPP are just those values returned by PREDPP in G to NONMEM. This is the usual behaviour required by NONMEM for single-subject data (and is a consequence of the single-subject assumption; see section A).

There is one type of population pharmacodynamic regression problem where the value in F (and possibly G) input to ERROR is needed, where this value is recomputed in ERROR, but where HH need not be computed. This is one where intraindividual variability cannot be expressed by a model for residual variability involving continuously distributed random ε variables, such as those models described in section A. An example of this occurs when the observation is pharmacodynamic, but discrete, a binary-valued outcome (0/1), say, where the probabilities that a 1 or 0 occurs depend on a scaled drug amount. However, they also may depend on pharmacodynamic parameters that vary from subject to subject, and/or the PK parameters themselves may vary from subject to subject. NONMEM can handle this type of problem, using an objective function that incorporates a model for random intraindividual variability which is appropriate for this type of an observation. Non-continuous observed responses are referred to as "odd-type data". Options LIKELIHOOD and -2LOGLIKELIHOOD on the ESTIMATION record are used to indicate to NONMEM that Y (with NM-TRAN abbreviated code) or F (with a user-supplied PRED or ERROR code) will be set to a (conditional) likelihood or -2 log (conditional) likelihood value, respectively. This can be done for all observations, or using F_FLAG (see Section D.B), for particular observations. The η's, if any, are understood to be population etas. No ε variables may be used. NOMMEM gives instructions to PREDPP to avoid an incorrect single-subject assumption.

C. ERROR Calling-Protocols

As the pharmacokinetic system is advanced, ERROR is called one or more times, each time with some argument record. The event records comprise these argument records, and are passed to ERROR in time order. The simulation and/or data analytic computations will normally be done correctly if routine ERROR is called with one event record after another (within an individual record), no event records being skipped, and no event record being repeated. This is the default. However, PREDPP can implement a few different protocols for calling ERROR. A protocol is specified by setting IDEF(2) to various values at ICALL=1 (for more about IDEF, see section B.1). The default is IDEF(2)=-1 (call with every event record), described above. The ERROR routine can be called only with each observation event record of the individual record. If this more limited sequence of calls is desired, this can be accomplished by setting IDEF(2)=0. Note, though, that *in this case*, IDEF(2) must be explicitly set to 0.

There are two general considerations to be remembered about calling-protocols with ERROR. First, no matter what calling-protocol is used, it applies only to calls to ERROR at ICALL=2; at ICALL=4 (i.e. during simulation) ERROR is always called with every event record. [This differs from calling-protocols with PK, where whatever calling-protocol is used for PK at ICALL=2 is the same as that used at ICALL=4.] Second, a calling-protocol that limits the sequence of calls to ERROR should not be requested when the value of F returned by ERROR may differ from the value input to ERROR.

Often, none of the ε derivatives depend on concomitant variables whose values vary within an individual record i.e. vary over time. This is true if y is given by (1) or (6), for example, or if log y is given by (5). This may not be true if y is given by (2); it depends on whether the value of Z varies over time. Unless there is only one observation per individual, this generally is not true if y is given by (3) or (4), since when time varies, f(P) varies, and time itself is to be regarded as a concomitant variable. When though this is true, then the values stored in the HH array must be the same for each observation event record of a given individual record (for THETA and ETA fixed). Considerable computation time can be saved; PREDPP need call ERROR *only once* per individual record, with the first event record only (for any given values of the THETA and ETA arrays). The user can request this calling-protocol by setting IDEF(2)=1. Since the first event record need not be an observation record, care must be taken that HH is indeed set with this record, and that any data items needed for this purpose are contained in that record.

Notice that with y given by (1), for example, the ε-derivatives do not depend on any concomitant variables, nor do they depend on θ or η. In such a case the values stored in the HH array must be the same for each observation event record of the *entire* data set; indeed, these values could be computed only once during the entire NONMEM problem. If at ICALL=1, IDEF(2) is set to 2, then at ICALL=2 values for HH will always be taken to be (constant) values stored in HH at ICALL=1. Note that the HH array is initialized to zero immediately before the call to ERROR with ICALL=1.

If log y is given by (5), the derivatives of log y with respect to the ε-variables are particularly simple; they satisfy the requirement for using IDEF(2)=2. By setting IDEF(1)=1, the user is specifying that when ICALL=2, the derivatives of log y are to be found in HH (see section B.1). By also setting IDEF(2)=2, these derivatives are actually taken to be the values stored in HH at ICALL=1.

With NM-TRAN, the abbreviated code for error model (4)

```
$ERROR
Y=F*EXP(EPS(1))
```

is implemented as described above. The generated ERROR subroutine contains this code:

```
IF (ICALL <= 1) THEN
IDEF(2)=2
IDEF(1)=1
HH(1,1)=1.0D0
ENDIF
```

See V.L.1 and figure 19 for the PREDPP summary that describes the above implementation. Note that there was a change starting with with NONMEM IV. Earlier versions of NM-TRAN implemented the model (4) as in figure 9. Figure 17 gives the same results if the $ERROR abbreviated code is replaced by the subroutine of figure 9; the difference is only in the run time.

Even when IDEF(2)=0, 1, or 2, a call to ERROR with any given event record can be forced with the use of the CALL data item (see section V.J).

IDEF(2) may be set to -1, or not set at all, i.e. this is the default. This results in ERROR being called with *every* event record (see above).

When NM-TRAN abbreviated code is used, the reserved variable CALLFL may be used to specify the value for IDEF(2). E.g., to set IDEF(2)=-1, use:

```
CALLFL=-1
```

Alternately, A calling protocol phrase can be used instead of the CALLFL pseudo-statement, e.g.,

```
$ERROR (CALL WITH EVERY EVENT)
```

D. Global and Other Reserved Variables

ERROR, like PK, can access variables other than through its argument list. Most of the modules are discussed in Chapter III, Section I. This section lists some of the variables of interest for ERROR. The help Guide VIII can be used to obtain the exact declarations and all variables of interest.

D.A Global Input Variables

USE PROCM_REAL , ONLY : **A**=>AMNT , **DAETA , D2AETA**

USE PROCM_REAL , ONLY : **TSTATE**

With both PK and ERROR, the amount in the Ith compartment A_I at the event time at which the call to the subroutine occurs can be found in A(I). This is the state vector. These amounts are particulary useful when pharmacodynamic models are specified in ERROR (see section B.2 and the example in section G.4). If η-variables are used in PK, the first and second partial derivatives of these amounts with respect to the η's are also useful. The first-partial derivative $\partial A_I/\partial\eta_K$ can be found in DAETA(I,K), and the second-partial derivative $\partial^2 A_I/\partial\eta_L\partial\eta_K$, $K \geq L$, can be found in D2AETA(I,K,L).

TSTATE is the state time associated with A, i.e., the time t at which A was computed.

ERROR should tell PREDPP whether or not it uses compartment amounts. See Section B.1, IDEF(3).

USE NMPRD_INT , ONLY : **NEWL2**

NEWL2 is similar to NEWIND.
NEWL2 = 1 if the data record is the first of an L2 record.
NEWL2 = 2 otherwise.

USE NMPRD_INT , ONLY : **NPDE_MODE**

NONMEM sets NPDE_MODE=1 when it is computing NPD and NPDE for TABLE or SCATTERPLOT. Otherwise, NPDE_MODE=0 See DV_LOQ in section D.B.

USE NMPR_REAL , ONLY : **TEMPLT**=>VRAW

May be used on the right in code for raw data averages at ICALL=6.

USE NMPRD_INT , ONLY : **NTHES_**=>NWTHT , **NETAS_**=>NWETA , **NEPSS_**=>NWEPS

The dimensions of THETA, OMEGA, SIGMA. Never less than 1.

The following group of variables gives the values of THETA, OMEGA, SIGMA at different NONMEM steps. Some of these may also be used in PK and other PREDPP subroutines.

USE NMPR_REAL , ONLY : **OMEGA**=>VARNF

The current value of OMEGA.
The current value of SIGMA is also located in this array as follows:

```
SIGMA(I,J)=OMEGA(NETAS_+I,NETAS_+J)
```

USE NMPR_REAL , ONLY : **THSIMP**=>THET_P , **OMSIMP**=>OMEG_P , **SGSIMP**=>SIGM_P

Values of THETA, OMEGA and SIGMA that are produced during a Simulation Step using the user-supplied routine PRIOR.

The following group of variables are also used in PK and are discussed in Chapter III, Section I.A.

USE NMPRD_INT , ONLY : **MSEC**=>ISECDER , **IFIRSTEM**

USE PROCM_INT , ONLY : **NEWIND**=>PNEWIF

USE NMPRD_REAL , ONLY : **ETA**

USE NMPRD_INT , ONLY : **ETASXI**

USE ROCM_INT , ONLY : **MIXNUM**=>MIXCALL , **MIXEST**=>IMIXEST

USE PROCM_INT , ONLY : **MNOW**=>MTNOW , **MPAST**=>MTPAST , **MNEXT**=>MTNEXT

USE PKERR_REAL , ONLY : **MTIME**

USE PRMOD_INT , ONLY : **MTDIFF**

USE NMPR_INT , ONLY : **RPTI**=>NRPT_IN , **RPTO**=>NRPT_OUT , **RPTON**=>NRPT_ON

USE ROCM_INT , ONLY : **NIREC**=>NINDREC , **NDREC**=>NDATINDR

USE NMPRD_INT , ONLY : **NPROB , IPROB**

USE ROCM_INT , ONLY : **NREP** , **IREP**=>NCREP

USE ROCM_INT , ONLY : **LIREC**=>NDATPASS

USE ROCM_INT , ONLY : **NINDR , INDR1 , INDR2**

INTEGER(KIND=ISIZE) , POINTER :: **S1NUM , S2NUM , S1NIT , S2NIT , S1IT , S2IT**

REAL(KIND=DPSIZE) , POINTER :: **DEN_ , CDEN_(:)**

D.B Global Output Variables

USE NMPRD_REAL , ONLY : **CORRL2**

Used when the values of epsilons used in the intraindividual model may be correlated across the observations contained in the L2 record. The ERROR routine should set CORRL2(k,m), for k=1,...,n, to the correlation between the values for the kth L2 observation and the nth L2 observation.

USE NMPR_REAL , ONLY : **CTLO**=>CTLW , **DCTLO**=>DCTLW , **DDCTLO**=>DDCTLW

USE NMPR_REAL , ONLY : **CTUP , DCTUP , DDCTUP**

USE NMPR_REAL , ONLY : **PR_CT**

With a given data record, either of the limits CTLO or CTUP may be set, thus defining an interval of values comprising one of several categories equated with the possible values of a potential observa- tion. During a copying pass (and during ICALL=5 and 6), NONMEM sets PR_CT to the estimated probability that an observation will be of the category in question. DCTLO are first eta derivatives of CTLO; DDCTLO are second eta derivatives. Similarly for DCTUP and DDCTUP.

USE NMPR_REAL , ONLY : **DV_LOQ**

"LOQ" stands for "limit of quantification". If the user's ERROR/PRED sets DV_LOQ and NPDE_MODE is set by NONMEM to 1, then the NPDE is being evaluated during this call, and this censored value is to be treated as if it is a non-censored datum with value of DV_LOQ. See NPDE_MODE in section D.A.

USE NMPR_INT , ONLY : **F_FLAG**=>IPRDFLG1

Setting F_FLAG to 1 (or 2) signals that Y is being set to a likelihood (or -2 log likelihood) value for this particular observation.

USE NMPRD_INT , ONLY : **MDVRES**

MDVRES stands for missing dependent variable (MDV) for residual (RES). Setting MDVRES to 1 is equivalent to temporarily declaring an observation as missing during the computation of residuals and weighted residuals

USE NMPR_REAL , ONLY : **YLO** , **YUP**

USE NMPR_REAL , ONLY : **PR_Y**

With a given data record, either of the limits YLO or YUP may be set so that during the analysis an interval is defined in which (or outside of which) an observation is conditioned to exist. During a copying pass (and during ICALL=5 and 6), NONMEM sets PR_Y to the estimated probability that an observation will fall within (or outside) the interval.

D.C Miscellaneous Global Variables

nonmem_reserved_general

There is a file nonmem_reserved_general in the util directory. This contains USE statement for a number of other NONMEM variables that may be useful for advanced users. See Chapter III, Section I.C, for a dicussion.

E. Displaying ERROR-Defined Items

A value stored in a variable (or array element) V in ERROR may be displayed in a table or scatterplot. To accomplish this, module NMPRD4 must be defined in ERROR, and V must be listed in NMPRD4. See the example in section G.4. Module NMPRD4 also provides a convenient place to store values of variables to be shared between ERROR and other routines, and it is used thusly when these routines are generated from NM-TRAN abbreviated code (see Guide IV). Reserved variable COMACT may be used on the right in ERROR. (INFN-defined and declared variables are also shared between user routines.)

The required declarations for NMPRD4 are described in section III.J.

F. PRED Error-Recovery from ERROR

The reader should see section III.K for a discussion about PRED-error recovery from PREDPP. ERROR can force an immediate return to NONMEM from PREDPP with a nonzero PRED error return code and accompanying user message. The contents of F, G, and HH are ignored. The required declarations are described in section III.K.2.

G. Examples

G.1 Example I: Population Data

In this section example I, described in section III.L.1, is continued. In particular, the code for ERROR is considered here. An observation is modeled as in (4) (section A). The code for ERROR is given in Figure 9. (A corresponding NM-TRAN abbreviated code is shown in Figure 17, as part of an NM-TRAN control stream.) All elements of IDEF should be 0 at ICALL=1. Since it is unnecessary to explicitly set any element to 0, this is not done. At ICALL=1 the value of HH is ignored.

An alternative code for ERROR is given in Figure 10. (The NM-TRAN abbreviated code shown in Figure 17 would work just as well for this second code as for the first code.) It is shown to illustrate the ERROR routine when the Simulation Step is implemented along with a data analysis step. One should imagine that drug levels are being simulated under the same model as has been described for the data analysis (except see the remark below). The code is essentially like that of Figure 9, except that at ICALL=4 the value for the ε-variable is obtained using the NONMEM utility SIMEPS (see Guide IV, section III.B.13), and the value for the observation y is computed. The value for y is returned in F. Note that y is modeled as in (4), and the computation at ICALL=4 follows this model exactly. However, the computation at ICALL=2 actually implements a slightly different model for residual variability than is simulated; see the discussion in section A.

G.2 Example II: A Mixture Model

In this section example II, described in section III.L.2, is continued. An observation is modeled as in (4) (section A). Therefore, the code for ERROR is the same as that for Example I and shown in Figure 9.

G.3 Example III: Single-Subject Data

In this section example III, described in section III.L.3, is continued. An observation is modeled as in (1) (section A). The data is from a single subject. Therefore, the ε-variable is to be regarded as an η-variable (see section B.1). The code for ERROR is given in Figure 11. (A corresponding NM-TRAN abbreviated code is shown in Figure 26, as part of an NM-TRAN control stream.) Since the η-derivative does not depend on any concomitant variables, nor does it depend on θ or η, ERROR need be called only once in the problem. Therefore, at ICALL=1 IDEF(2) is set to 2, and the derivative is stored in HH. Since ERROR is called only once (at ICALL=1), ICALL need not be tested.

An alternative code is shown in Figure 12. (The NM-TRAN abbreviated code shown in Figure 26 would work just as well for this second code as for the first code.) It is shown to illustrate the ERROR routine when the Simulation Step is implemented along with a data analysis step.

G.4 Example IV: Single-Subject Pharmacodynamic Data

In this section example IV, described in section III.L.4, is continued. An observation is modeled as in (3) (section A). The data is from a single subject. Therefore, the ε-variable is to be regarded as an η-variable (see section B.1). The code for ERROR is given in Figure 13. (A corresponding NM-TRAN abbreviated code is shown in Figure 30, as part of an NM-TRAN control stream.) The pharmacodynamic model is given by a simple "Emax type" model:

$$f(P) = \frac{E_{max}Ce}{C_{50} + Ce} \tag{19}$$

where $P=(K12,K20,K30,VD,E_{max},C_{50})$, and Ce is the concentration in the effect compartment based on K12, K20, K30, and VD.

An alternative code is shown in Figure 14. (A corresponding NM-TRAN abbreviated code is shown in Figure 32, as part of an NM-TRAN control stream.) It is shown to illustrate the the ability to display ERROR-defined items. In this case the intent is to display the concentrations in both the central and effect compartments. The control stream requesting the display is described in section V.L.4. The code in Figure 14 just shows how these concentrations become ERROR-defined. It also illustrates the use of the state vector A in PROCM_REAL (see section D.A).

H. Other Subroutines That May Be Called

RANDOM

The NONMEM utility routine RANDOM may be called by ERROR during the Simulation Step (ICALL=4) and when data averages are being computed (ICALL=5) to obtain numbers from different random sources.

V. Event Records

A. Introduction

In this chapter we describe data items required by PREDPP. The reader should be familiar with chapter II before proceeding with this chapter.

Use of PREDPP requires that each data record contain certain NONMEM data items, i.e. data items needed by NONMEM. These are the ID and MDV data items (see chapter II). The MDV data item is required because with PREDPP not all data records will contain observations, and when MDV data items appear, so must the ID data items (even with single-subject data).

Use of PREDPP also requires that each event record contain two data items which are needed by it, i.e. these data items are examined by PREDPP, but not by NONMEM. These are the time (TIME) and the event identification (EVID) data items. Depending on the EVID data item in an event record, PREDPP also requires that the event record contain certain additional data items. Depending on some one of these additional data items in an event record, further additional data items may be required in the event record. Lastly, certain data item types are recognized PREDPP, though their use is not required e.g. the CONT data item. The CONT data item was discussed in Chapter II; it allows an event record to span several data records. It is a little different from the other types of data items needed by PREDPP because if it appears, it must appear in all data records of an event record. Each other type of data item needed by PREDPP need appear only in the last data record of an event record.†

There are 11 different types of data items altogether that are recognized and acted on by PREDPP. These are called the PREDPP data items . Any particular type of PREDPP data item may appear in any field of a data record, as long as for a given type, the same field is used in each data record containing data items of this type. If a PREDPP data item type appears in the data set, its field number, or index , is communicated to PREDPP in a way described next. Otherwise, it is said to be missing .

The index of the Ith type of data item should be placed in the Ith field of the INDEX control record, which is the fourth control record. (Control records are discussed in Guide I, and the INDEX record, in particular, is discussed in section C.4.1 of that document.) *If NM-TRAN is used, it is sufficient to place the reserved name of the data item in the nth position of the $INPUT record, where n is the field number*. The effect of so doing is to place the index of the Ith type of data item in INDXS(I). The values in the INDXS array are available to the PRED subroutine. For example, the CONT data item is the 10th type of PREDPP data item. If, as in the example in Chapter II, the index of the CONT data item is 3, then 3 should be placed in the 10th field of the INDEX record, and in so doing, it is also placed in INDXS(10). If NM-TRAN is used, the name CONT is placed in the 3rd position of the $INPUT record.

Although this chapter describes the use of CONT, there is no need for it any longer.

If the Ith type of data item is missing, then the INDEX record would not contain its index, i.e. the Ith field of the INDEX record would contain either a zero or a blank. *If NM-TRAN is used, the reserved name of the data item type would not appear in the $INPUT record*. It is assumed that all data items of this type are 0 in all event records. If the CONT data item is missing, it is assumed that all data items of this type are 0 in all data records.

Indices of data items other than PREDPP data items may be needed in a user-supplied PK, ERROR, or INFN (see section VI.A) subroutine. This is illustrated in section III.C. These indices may be placed in the INDEX record beginning with field 12. In the user-supplied subroutine they appear in the INDXS array beginning with position 1 and in the order in which they appear in the INDEX record. *If NM-TRAN is used, the labels of the data items may simply be placed in the $INDEX record*. In the user-supplied routine, the indices appear in the INDXS array beginning with position 1 and in the order

† The CONT data item described in this paragraph cannot be used with NM-TRAN abbreviated code. It is not needed because the number of data items per data record can be made as large as needed. See the discussion of PD and $SIZES in Chapter II.

in which the labels appear in the $INDEX record.†

Each of the different types of PREDPP data items, except the CONT data item, are discussed in turn in the next 10 sections. The CONT data item has already been discussed adequately in chapter II. The possible values for all 11 different data items types are summarized for the reader's convenience in Appendix I.

B. The Event Identification (EVID) Data Item

This type of data item (PREDPP data item no. 1) *must appear* in event records. It identifies the type of event. It has 5 possible values.

A value of 0 indicates that the event is an observation event . With this type of an event the DV data item on the last data record of the event record is an observation, and the MDV data item on this data record is 0. With all other types of events, the DV data item on the last record of the event record may be any value (e.g. blank), and the MDV data item on this data record is 1.

A value of 1 indicates that the event is a dose event . There are 3 types of doses: a bolus dose, an infusion (of finite duration), and a (continuous) constant rate infusion considered to be of effectively infinite duration and leading to steady-state. This last type of dose is called a steady -state infusion . Repetitions of doses of the first two types can also lead to steady-state (see section F). A dose event corresponds to either the time a bolus dose is given, or the time an infusion is initiated, or the time a steady-state infusion terminates. With a dose event certain other data items which define the event further may also appear in the event record.

A value of 2 indicates that the event is neither an observation nor dose event. The corresponding event is referred to as an other-type event. Examples of other-type events are: the time a urine collection begins, the time a urine collection ends, and the time a change in a covariable (such as glomelular filtration rate) is noted. The user may create an other-type event for whatever reasons he wishes; he need only mark an occurence of this type of event with an event record containing an EVID data item equal to 2.

A value of 3 marks a reset event . With a reset event the kinetic system is reinitialized. Time is reset to *whatever* time appears in the event record (e.g. 0), the amounts in each compartment are reset to zero, and the on/off status of each compartment is reset to its initial status (see section H).

A value of 4 marks a reset-dose event . With such an event the characteristics of a dose event and a reset event are combined. The system is first reset, and then a dose is given according to the information given in those fields giving dose information (see below).

The "Repeated Observation Records" feature exists to assist in specialized methodologies such as stochastic differential equations. A record in a data file may be set up for repeated calls to PK and ERROR. Each time, the same record is passed to PK and/or ERROR, but with a different EVID. The user's control stream model in $PK or $ERROR may then take advantage of executing certain code conditional on the EVID value. For this to occur, the user must introduce one or more of the following data items in the data file, with these names:

XVID1 XVID2 XVID3 XVID4 XVID5

These stand for "extra" EVIDs. On the first call to PK/ERROR, the EVID is set to the value given in XVID1. On the second call, the EVID is set to that in column XVID2, etc. up to XVID5. Only as many XVIDs as are required are needed to be defined. All the other items in the record do not change, except that if the present EVID used is not 0, then the MDV value is set to 1 for that call. If an XVID is -1, then the call to PK/ERROR for that XVID is not made, nor for the remaining XVIDs. If there is an EVID column, the value in this column is not passed to PK/ERROR unless XVID1=-1 in which case a "normal" call with that record occurs.

† The INDXS array cannot be used with NM-TRAN abbreviated code.

C. The Time (TIME) Data Item

This type of data item (PREDPP data item no. 2) *must appear* in event records. It gives the time of the event. (An exception occurs with ADVAN9 when there are only algebraic equations to be solved, in which case the time data item is optional.) With NONMEM 7.4, time data items may have any value, including negative values; with earlier versions, time data items must be non-negative. (NM-TRAN has always allowed negative clock times when day-time translation is performed, and converted them to non-negative relative times for PREDPP.) Also, event records must be ordered (within an individual record) so that if event record A is followed by event record B, and the corresponding times are t_A and t_B, then $t_A \le t_B$. Exceptions to this are provided by reset and reset-dose events where the time may be *any* number.

Note: when IDEF(1,2) is set to 0 in PK (see section III.H), and $t_A = t_B$, then PK is not called with event record B unless a call is forced with the use of the CALL data item; see section J.

The PREDPP TIME data item is a unitless relative time. However, when NM-TRAN is used, TIME values of the NM-TRAN data set may contain clock times (e.g., hh:mm). The DATE data item may also be present. In these cases NM-TRAN performs date-time conversion, converting DATE (if present) and clock times to relative time for the NONMEM/PREDPP data record. In addition, the NM-TRAN $DATA record may contain the option TRANSLATE=(TIME/f/d). This requests that TIME values be divided by f (typically, f is 24 to convert hours to days, with d significant digits.

D. The Dose Amount (AMT) Data Item

This type of data item (PREDPP data item no. 3) often appears in event records, but it need not appear. In a dose event record it gives the amount of a bolus dose or of an infusion of finite duration. This amount will be a positive number. However, if other data items in a dose event record (see below) indicate that the event record marks the termination of a steady-state infusion, then if the AMT data item appears, it must be 0. If all doses are steady-state infusions, then the dose amount data item can be missing.

If the AMT data item appears, then it must be 0 in observation, other-type, and reset event records. In a reset-dose event record the AMT data item should satisfy the requirements of the AMT data item in a dose event record.

E. The Dose Rate (RATE) Data Item

This type of data item (PREDPP data item no. 4) often appears in dose event records, but it need not appear. For example, if all doses are bolus doses, then the dose rate item can be missing. In a dose record the RATE data item can assume 4 different kinds of values. A positive number indicates an infusion rate. The appearance of a positive AMT data item in the event record, giving the amount of the infusion, indicates that the infusion is of finite duration.

A 0 indicates that the dose is an instantaneous bolus dose. A -1 indicates that the dose is either (i) a zero-order bolus dose (see section III.F.3) and that its rate is modeled in PK (see section III.F.4) or (ii) a rate-modeled steady-state infusion (see section III.F.5). A -2 indicates that the dose is a zero-order bolus dose and that its duration is modeled in PK (see section III.F.3). If the dose is a zero-order bolus dose, then a positive AMT data item must appear in the event record.

The RATE data item must be 0 in observation, other-type, and reset event records. In a reset-dose event record the RATE data item should satisfy the requirements of the RATE data item in a dose event record.

F. The Steady State (SS) Data Item

F.1 Background

Before giving the detailed description of this data item, it is helpful to give some considerable background concerning steady-state and its relationship to PREDPP.

Steady-state in a kinetic system can be achieved in several ways. One way is by administering a sequence of doses in a regular cyclic fashion over a sufficiently long time. The doses may be input into the same or different compartments. The doses given during any one cycle are the same as the doses given during any other cycle; collectively, they are called the dose cycle . They may consist of a mixture of bolus doses and infusions of finite duration. A first cycle of doses is given, then a second cycle, etc. until a steady-state has been reached. For example, a dose cycle may consist of an instantaneous bolus dose of 10mg, immediately followed by a 30 minute infusion of 10mg. This cycle might be repeated every 4 hours, the so-called period of the cycle. Let π denote the period, and suppose the dose times of the last cycle are $t_1 \le t_2 \le \cdots \le t_n$. The system is at steady-state at a time t in the (closed-open) interval $[t_1 , t_1 + \pi)$ if the drug amounts in the compartments at time t equal the amounts at time $t + \pi$, *were the dose times of the last cycle actually* $t_1 + \pi \le t_2 + \pi \le \cdots \le t_n + \pi$ (and there were no event times in the interval $[t_1 + \pi , t_1 + \pi)$).

A second way to achieve steady-state is to initiate a number of simultaneous steady-state infusions. Suppose these infusions terminate at times $t_1 \le t_2 \le \cdots \le t_n$. The system is at steady-state at a time t in the interval (t_1 , t_n) if for a positive Δ, the drug amounts in the compartments at time t equal the amounts at time $t + \Delta$, *were the infusions actually terminated at* $t_1 + \Delta \le t_2 + \Delta \le \cdots \le t_n + \Delta$. A third way is to use a regular cyclic dosage pattern in conjunction with simultaneous steady-state infusions. Steady-state is an ideal state, but it can be approximated.

If a large enough number of dose cycles have occurred, the user might consider that an (approximate) steady-state has been reached by at least the time the last cycle begins. Rather than include a large number of dose event records into the data set, the user may be able to take advantage of a PREDPP feature requiring that only dose records for the doses in the last cycle be given. This last dose cycle is called the steady-state cycle , and the doses in this cycle are called steady-state doses . If steady-state infusions are involved, only dose records marking their terminations are required. These special dose event records are called steady-state dose records . They are identified as such by means of the steady-state data item (see section F.2). A steady-state dose record describing a bolus dose or an infusion of finite duration actually describes an sequence of doses all just alike, given regularly at a specified interdose interval, and ending with the steady-state dose. All these doses, excluding the steady-state dose, are called the implied doses . Without loss of generality the sequence of times at which the implied doses are given is imagined to extend infinitely backward in time (beyond the first event time). These times are called the implied dose times . A steady-state infusion is imagined to have started at $-\infty$.

The amounts of drug in the various compartments are *computed by PREDPP* to be at their steady-state values at the time of the last dose of a steady-state cycle, or at the latest termination time of a group of simultaneous steady-state infusions, but not before this time. PREDPP computes these steady-state values by using steady-state kinetic formulas in conjunction with the usual transient type kinetic formulas. This computation is called the steady-state computation . Computation with steady-state dose records and steady-state formulas takes less computer time than does computation with transient dose records and transient kinetic formulas. So when the former can be incorporated, substantial computer time, as well as data set construction time, can be saved.

Steady-state can be achieved in linear kinetic systems and sometimes in nonlinear kinetic systems. With linear systems the full range of steady-state dosage patterns can be implemented with PREDPP; with nonlinear systems, only a limited range of steady-state dosage patterns can be implemented. In linear systems the so-called superimposition principle holds. This principle implies that if steady-state is achieved with a dose cycle of m doses, with period π, then the amount of drug in a compartment at steady-state and at a given time t is $U_1 + U_2 + \cdots + U_m$, where U_i is the amount of drug in the compartment at time t that would be present using a dose cycle consisting of only the ith of the m doses and with period π. PREDPP takes advantage of this in a way that may be explained by means of the following example.

Suppose the dosage pattern -- 3 doses daily: 20 mg at 8 AM, 10 mg at 3 PM, and 10 mg at 8 PM -- is given, and steady-state is reached. The dose cycle could consist of the 8 AM dose, followed by the 3 PM dose, followed by the 8 PM dose. However, a steady-state solution is valid if a dosing pattern has continued for a sufficiently long time, and if valid, it is independent of when this pattern was initiated. So alternatively, the dosing cycle could consist of the 3 PM dose, followed by the 8 PM dose, followed by the 8 AM dose. Or, it could begin with the 8 PM dose, but in any case, the order of the doses should be chronological within a cycle, so that, for example, the 8 PM dose should not be followed by the 3 PM dose. Which of the 3 alternatives to choose is simply a matter of user strategy. To help explain this, suppose an observation is made at 4 PM March 9 and that the dosing cycle is given many days preceding March 9 and continuing beyond this date. The steady-state computation occurs only at the time of the last dose of the steady-state cycle (although, as shall be seen, partial steady-states are computed at earlier dose times). If the steady-state computation does not occur before 4PM March 9, dose records accounting for *all* the doses given before this time must be included in the data set. So, to take advantage of the steady-state feature, a steady-state cycle must end before 4 PM March 9. If a dose cycle is defined to begin with the 8 AM dose, then the steady-state cycle must end at 8 PM March 8 or earlier. Say it ends at 8 PM March 8; then non-steady-state dose records at 8AM and 3 PM March 9, accounting for the two doses given at these two times, must also be included. If, though, a dose cycle is defined to begin with the 8 PM dose, then the steady-state cycle can end at 3 PM March 9, and no nonsteady-state dose records need be included between the 3 PM March 9 dose record and the 4 PM observation record. Suppose then that one chooses this more efficient strategy.

Then a steady-state dose record at time 8 PM March 8 appears. This record should specify that the amounts in all the compartments be reset to the steady-state amounts that would result were only 10 mg given daily, 8 PM March 8 being the time of the last such dose (U_1 amounts at 8 PM March 8). There is a way to do this using the steady-state data item. There should also be a steady-state dose record at time 8 AM March 9. This record should specify that (i) the amounts in all compartments be updated to amounts valid for 8 AM March 9, using transient type kinetic formulas (U_1 amounts at 8 AM March 9), and that (ii) they then be changed by adding to them the steady-state amounts that would result were only 20 mg given daily, 8 AM March 9 being the time of the last such dose (U_2 amounts at 8 AM March 9). There is a way to do this using the steady-state data item. Finally, a steady-state dose record at 3 PM March 9 should be included, specifying that (i) the compartment amounts be updated to amounts valid for 3 PM March 9 ($U_1 + U_2$ amounts at 3 PM March 9) and that (ii) they then be changed by adding to them the steady-state amounts that would result from a daily 10 mg dose, 3 PM March 9 being the time of the last such dose (U_3 amounts at 3 PM March 9). At 3 PM March 9 the compartment amounts are thus set to their correct steady-state values. If a number of additional dosing cycles occur after 4 PM March 9 and before the next non-dose event record, then similarly, the user can minimize the total number of required dose event records by including steady-state dose records before this next record.

A slightly more general application of the superimposition principle allows "transient doses" to be handled along with those of a steady-state dosing cycle. If, for example, a 20 mg dose was given (perhaps inadvertantly) at 3PM March 5, then, in effect, an extra dose of 10 mg was given on March 5 which cannot be properly taken into account in the steady-state cycle. However, the user may firstly, include a nonsteady-state dose record for 3 PM March 5 with dose amount 10 mg and then, specify on the steady-state dose record for 8 PM March 8, that (i) the compartment amounts be updated to amounts valid for 8 PM March 8 and that (ii) they then be changed by adding to them the steady-state amounts corresponding to the 8 PM dose.

The superimposition principle also implies that if steady-state is achieved with a number (n) of steady-state infusions, then the amount of drug in a compartment at steady-state and at time t is $V_1 + V_2 + \cdots + V_n$, where V_i is the amount of drug in the compartment at time t that would be present were only the ith infusion given. If n=2, for example, then 2 steady-state dose records should be used, each containing a positive dose rate data item. The times on these records should be the times t_1 and t_2 when the first and second infusions terminate. We will suppose $t_1 \le t_2$. The first dose record may specify that the compartment amounts at t_1 be reset to the steady-state amounts corresponding to the first infusion (V_1 amounts at t_1). The second dose record should specify that (i) the compartment amounts be updated to amounts valid for time t_2 (V_1 amounts at t_2) and that (ii) they then be changed by adding to them the steady-state amounts corresponding to the second infusion (V_2 amounts at t_2). Note that t_1 may equal t_2.

We have discussed repetitive doses leading to a steady-state in terms of a dose cycle with a period. Indeed, repeated doses of possibly different amounts are often given in a cyclic manner. However, the only really important structure of repeated dosing, allowing the application of the superimposition principle as described above, is that there be a group of doses of possibly different amounts, each of which is repeated with its own constant period. So using the above example, the 20 mg dose is given repeatedly with a period of 24 hours, the 3 PM 10 mg dose is given repeatedly with a period of 24 hours, and the 8 PM 10 mg dose is given repeatedly with a period of 24 hours. If, though, the values of these three periods differed, a type of steady-state would still be achieved. As before, one could simply include one steady-state dose event record for each type of dose. For each type of dose, its steady-state dose record would have a time data item equal to the time the last dose of that type is given and a period (called the interdose interval data item) *specific* for that particular dose type. In the same way, repeated bolus doses can be handled in conjunction with steady-state infusions. Simply regard a steady-state infusion conceptually as a particular sequence of bolus doses, each of very small amount Δa, and given very small Δt time units apart, such that $\Delta a/\Delta t$ is the infusion rate. (However, since only this rate and the termination time matter, a Δa and Δt are not given on the steady-state dose record.)

With nonlinear kinetic systems the superimposition does not hold, and steady-state amounts do not add as described above. Consequently, a steady-state cycle can consist of at most one dose (either a repeated bolus dose or a repeated infusion), and steady-state infusions cannot be simultaneous. But within those limitations, steady-state dosing can be accommodated with PREDPP.

When the state vector of compartment amounts is advanced from one time to the next using the transient type kinetic formulas, the PK parameters act continuously or discretely, as described in chapter III. When a steady-state formula is used to compute a state vector at event time t, a PK parameter that regularly acts continuously over (finite) state-time intervals, now acts continuously over the interval $(-\infty, t)$. Its value is assumed to be constant over this interval and is computed from the information in the steady-state dose event record for time t. A PK parameter that regularly acts discretely at state times, now acts discretely at the infinite set of implied dose times. For example, if a steady-state record describes a bolus dose at time t hours, terminating a sequence of implied bolus doses given every 12 hours, then the PK parameter (e.g. a bioavailability fraction or duration parameter) acts discretely at times t, t-12, t-24, t-36, etc. Its value is assumed to be the same at these times and is computed from the information in the steady-state dose event record for time t.

An absorption lag time for a given compartment (see section III.F.6) applies to all steady-state doses into the compartment, except to steady-state infusions. According to what has just been stated about discretely acting parameters, all the implied doses, as well as the steady-state dose, are lagged, and by the same amount L. Steady-state kinetics are computed at the event time t on a steady-state dose record. The amounts computed for time t are the steady-state amounts at time t-L were the doses not lagged. Care must be taken when an absorption lag time is applied to a steady-state dose. Usually, the user intends that the system be at steady-state at time t+L. If an event time occurs between t and t+L, this can result in a perturbation of the system such that steady-state no longer exists at time t+L. Or, if at the first event time s succeeding t+L the value of a continuously acting PK parameter is changed, then this too can "destroy" the steady-state at t+L, because the kinetics governing the advance over (t,t+L) are governed by the values of the continuously acting PK parameters computed at time s.

F.2 Specifics of the Steady-State (SS) Data Item

The steady-state data item (PREDPP data item no. 5) may appear in event records, or it may be missing. If it appears, then its value in dose event records is either 0, 1, or 2.

A 0 indicates that the dose event record is not in fact a steady-state dose event record.

A 1 indicates that the dose event record is a steady-state dose event record and that the compartment amounts are to be reset to the steady-state amounts. These amounts result from either (a) repeated bolus doses with a given period, (b) repeated infusions with a given period, or (c) a steady-state infusion, as specified further on the record. In case (a) the event record must also have positive dose amount and interdose interval data items. If the repeated doses are instantaneousr bolus doses, the event record must also have a 0 dose rate data item. If the repeated doses are zero-order bolus doses, the event record must also have a negative dose rate data item. The dose amount and dose rate data items apply to all the repeated bolus doses. In case (b) the event record must also have positive dose amount, dose rate, and interdose interval data items. The dose amount and dose rate data items apply to all the repeated infusions. In case (c) the rate data item on the event record must either be positive or -1, and the dose amount and interdose interval data items must be 0.

A 2 indicates that the dose event record is a steady-state dose event record and, letting t be the time on the record, that (i) the amounts in the compartments are to be updated to amounts valid for time t (using transient kinetic formulas) and (ii) they then are to be changed by adding to them steady-state amounts. These steady-state amounts are the same as those that would be computed were the steady-state data item equal to 1, and the necessary additional data items on the event record are as described for cases a-c above.

A 3 indicates that the dose event record is a steady-state dose event record. SS = 3 is identical to SS = 1 with one exception: with Steady State routines SS6 and SS9, the existing state vector (compartment amounts and eta derivatives) is used as the initial estimate in the computation of the steady-state amounts. The user supplies the initial estimate with some combination of prior event records, e.g., reset, transient dose, and other-type event records.

The SS data item must be 0 in observation, other-type and reset event records. In a reset-dose event record the SS data item should satisfy the requirements of the SS data item in a dose event record.

If SS is 1 or 2 or 3, the output compartment must be off.

F.3 Initial Steady State

If endogenous drug production terms are present in the differential equations for a general non-linear model (ADVAN6, ADVAN8, ADVAN9, ADVAN13, ADVAN14, ADVAN15), the system may be at a non-zero initial condition before any doses are introduced. This is referred to as an <u>initial</u> <u>steady-state</u> condition. One way to obtain the initial condition is by using a dose event record having SS>0, AMT=0, RATE=0. Another way is to use the Initial Steady State feature of PREDPP. With this feature, the initial condition is computed prior to the first event record (and after a reset event record) exactly as if such a dose record were present. It is not necessary to include the a dose record or, indeed, to define the data item SS, AMT or RATE. There are three ways to request this feature:

I_SS ($MODEL)
    Set the I_SS option of $MODEL. This applies to all subjects.

ISSMOD (MODEL subroutine)
    Set the ISSMOD variable in subroutine MODEL. This applies to all subjects.

I_SS ($PK)

Set the I_SS variable in $PK (or PK routine). This allows I_SS to be set conditionally, e.g., if some subjects are at steady-state and others are not.

If ADVAN9 is used and there are equilibrium compartments, they are computed at the same time. This is then an "initial equilibrium" feature.

The initial steady-state feature may be used with the compartment initialization feature of PREDPP (See Chapter III, Section I.B).

F.4 Implementation of Steady State

Different kinds of Steady-State doses are pictured in Users Guide V, Introductory Guide, Chapter 6, Section 8.2.3. It might be helpful to review these dosing patterns.

With the analytic SS routines, steady-state uses well-known analytic solutions. For example, with ADVAN1/SS1 for the one-compartment model, the differential equation is $dA/dt = -K * A$. When this is integrated from t=0 to t=T the solution is $A = \exp(-K * T)$

Suppose a bolus dose AMT is given at the end of the Steady State interval II. At steady-state,

$A_{ss} * \exp(-K * II) + AMT = A_{ss}$

That is, A_{ss} is the compartment amount such that after an advance in time of II, plus the dose AMT, the compartment amount is still equal to A_{ss}. This can be solved as

$A_{ss} = AMT/(1 - \exp(-K * II))$

With all analytic models (SS1, SS2, SS3, SS4, SS5, SS7, SS11, SS12) the solution is the same. A_{ss} is a vector of compartment amounts and eta derivatives, K is a matrix of rate constants and their eta derivatives, and AMT is a vector with dose to a single compartment and that dose's derivatives. The matrix exponential is solved in a general way in SS5 and SS7. Other SS routines contain a solution for a specific number of compartments.

With multiple infusions, steady-state is solved for in the same way using the known analytic solution.

Steady-State with constant rate infusion is computed differently. At steady-state, input equals output:

$dA_{ss}/dt + R = 0$
$-K * A_{ss} + R = 0$
and thus
$A_{ss} = R/K$

As with steady-state with multiple dosing, the solution is the same with all analytic models. A_{ss} is a vector of compartment amounts and their eta derivatives, K is a matrix of rate constants and their eta derivatives, and R is vector with rate into a single compartment and that rate's eta derivatives,

When the model is defined by differential equations (SS6, SS8, SS9), PREDPP cannot use an analytic solution, even if one happens to exist. Instead, the solutions are found by a numerical technique using ZSPOW, a root-finding subroutine from IMSL.

With multiple dosing, ZSPOW searches for a vector of compartment amounts and their eta derivatives such that the initial and final (end of dosing interval) vector is the same. ADVAN6, ADVAN8, and ADVAN9 perform a numerical integration of the differential equations (and in the case of ADVAN9, algebraic equations). The ADVAN routine is used to advance (integrate) the state vector A from time 0 to time II with the appropriate dose pattern. Let A_{II} represent the state vector after advance to time II with the appropriate dose. A search is made for A_{ss} such that $A_{ss} - A_{II} = 0$

If SS=3, the state vector A contains user-supplied initial estimates. Otherwise, the SS routines sets A=0 and makes three calls to ADVAN. Each call advances the state vector from t=0 to t=II with the appropriate dosing pattern. The result is the initial estimate.

Once the SS solution is found, the SS routine adds the final bolus dose to the state vector, or starts the final infusion.

As with analytic advans, steady-state with constant rate infusion is computed differently. At steady-state, input equals output. A search is made by the root finder ZSPOW for A_{ss} such that

$0 = dA_{ss}/dt + R$
Within PREDPP, $dA_{ss}/dt = DADT(A_{ss})$, so this search is for:
$0 = DADT(A_{ss}) + R$

ZSPOW requires an initial estimate. As above, if SS=3, the state vector A contains an initial estimate.

Otherwise, an initial estimate is obtained using the first two terms of the Maclaurin series approximation for dA/dt. For any f(x), this is $f(x) = f(0) + f'(0) * x$

Here, f(x) is dA/dt and $f'(x)$ is $\partial f(x)/\partial A$

The DES routine computes dA/dt as DADT and $\partial DADT/\partial A$ as DA. DES is called with A=0 to obtain DADT(0) and DA(0). The SS routine computes DAINV = 1/DA(0). The approximation is:

$DADT(A) = DADT(0) + DA(0) * A$

Solve for A in:

$0 = DADT(0) + DA(0) * A + R$
$A = -(DADT(0) = R)/DA(0)$
$A = -DAINV * (DADT(0) + R)$

This A is the initial estimate.

Note 1

The time T_{ss} of the steady-state dose record is ignored. Steady-state is computed for time=0.

Note 2

Multiple infusions with duration > II are permitted. The SS routines treat these as a constant rate infusion superimposed on multiple infusions with duration < II.

Note 3

As stated in chapter III, if an absorption lag parameter ALAGn is defined for the compartment receiving a steady-state multiple dose, it applies not only to this dose, but also to all the preceding implied doses. The lag time should not exceed the interdose interval. Steady-state amounts are computed as usual, then the state vector is advanced to time II − ALAG.

For example, suppose II=24 and ALAG=8. First, steady-state is computed as if there were no lag time, so that the most recent implied doses were given at times -72, -48, -24, 0. Because ALAG=8, they were actually given at times -64, -40, -16. Thus, 16 units of time have passed since the last of the implied doses. The steady-state amounts are adjusted to time II − ALAG = 24 − 8 = 16. Event records with times up to (but not including) time 8 will have appropriate steady-state amounts in the compartments. A transient dose at time 8 will maintain steady-state from time 8 until the next transient dose is needed at time 24 + 8 = 32.

Note 4

Model event time MTIME is not a dose-related feature and has no effect on steady-state doses. Even if PK computes MTIME < II, this does not apply retroactively to the preceding implied doses. For example, if MTIME is used to model EHC (enterohepatic circulation), then steady-state doses should not be used because they will give inappropriate results. Instead, EHC can be modelled with continuous terms in the differential equations. An example is provided using a smooth step model with Hill terms in a sigmoid emax model. With such models, steady-state doses may be used. See "Enterhepatic circulation examples" in on-line help, and mtimess.ctl and hillss.ctl in the examples directory.

G. The Interdose Interval (II) Data Item

This type of data item (PREDPP data item no. 6) may appear in event records or be missing. If some steady-state dose event record has a positive AMT data item, then the II data item must also appear in the record. If some non-steady-state dose event record has a positive ADDL data item (see section K), then the II data item must also appear in the record.

In a dose record the II data item can be 0 or a positive number. For repeated doses with a given period, the II data item is this period. For a steady-state infusion the II data item must be 0.

The II data item must be 0 in observation, other-type, and reset event records. In a reset-dose event record the II data item should satisfy the requirements of the II data item in a dose event record.

Like TIME, the PREDPP II data item is unitless. However, when NM-TRAN is used, II values of the NM-TRAN data set may contain clock times (e.g., hh:mm), and NM-TRAN converts them to hours for the NONMEM/PREDPP data record. In addition, the NM-TRAN $DATA record may contain the option TRANSLATE=(II/f/d). This requests that II values be divided by f (typically, f is 24 to convert hours to days) with d significant digits.

H. The Compartment (CMT) Data Item

The prediction for an observation may be either a scaled drug amount in some particular compartment called the observation compartment (see section III.F.1), or a modification of this quantity (see section IV.B.2). With every dose there is associated a certain dose compartment (see section III.F.2). With every kinetic model there are designated a default observation compartment and a default dose compartment . For some kinetic models, these designations are built into the model definition; for others, these designations are given by the user in a special subroutine MODEL (see section VI.B). Unless otherwise specified on an observation event record by means of the compartment data item, the observation compartment is taken to be the default observation compartment. Unless otherwise specified on a dose event record by means of the compartment data item, the dose compartment is taken to be the default dose compartment.

The compartment data item (PREDPP data item no.7) may be appear in event records or be missing. If it has the value 0 on an observation, dose, or reset-dose event record, then the appropriate default compartment is used for the observation compartment or dose compartment. Otherwise, the number given as the compartment data item on the record is the number of the observation or dose compartment, respectively.

By convention, the number of the output compartment may also be given by the CMT data item as the number 1000 (this is true regardless of the type of the event record). With small models (number of compartments <= 99), this number may also be given as 100.

Some compartments can be turned on and off at event record times. When a compartment is off, the amount of drug in the compartment is 0. When a compartment is on, drug distributes from or into the compartment according to the kinetic model. For example, the output compartment may be turned on at a time coinciding with the beginning of a urine collection. Or, a drug depot compartment can be turned off at a time when there should be virtually no drug left in this compartment, and the user wishes to eliminate the unnecessary computation involved in computing the amount of drug in this compartment. Every compartment is initially either on or off --its so-called *initial status*, at the "beginning" of an individual record. If a compartment is not turned on or off as the result of some event record in the individual record, then the on-off status of the compartment at each event time in the individual record is given by the compartment's initial status. (Compartments can also be turned on and off when all the data are from a single subject, and then the term 'individual record' here should be "understood" to mean the entire data set. This is a consequence of the single-subject assumption; see section IV.A.) With every kinetic model only certain designated compartments can be turned on and off, and the initial status of each compartment is also designated. For some kinetic models, these designations are built into the model definition; for others, these designations are given by the user in a special subroutine MODEL (see section VI.B).

With a dose event record the dose compartment is turned on at the time the dose enters the system, if this compartment is not already on at that time. With an observation event record the observation compartment must either be the default observation compartment (which can never be turned off from its initial status of being on), or a compartment whose status at the time of the observation event record is on.

Any drug which would distribute from a compartment A to a compartment B, were B on, leaves compartment A when B is off. It goes into the output compartment when the output compartment is on, or is lost to the system entirely if the output compartment is off.

Generally, giving a compartment number in the compartment data item has the effect of (i) specifying the observation compartment, if the event record is an observation record, (ii) specifying the dose compartment, if the event record is a dose record, *and* (iii) turning the compartment on at the event record time, if it is not already on. If all one wants to accomplish at a certain time is to turn a compartment on (e.g. the output compartment, in order to begin a urine collection), then one should construct an other-type event record that simply turns the compartment on, i.e. that gives the compartment number as the compartment data item. For some kinetic models, the compartment numbers are built into the model definition; for others, these numbers are given by the user in a special subroutine MODEL (see section VI.B).

To turn a compartment off at a given time, one should construct an other-type event record that gives the compartment number, preceded by a minus sign, as the compartment data item. A minus sign can also precede the compartment number on an observation record, provided the observation compartment is the output compartment or an output-type compartment. In this case the compartment must be on at the event time, but it is turned off "immediately thereafter". Thus one can record the observation of a urine concentration in an event record and conveniently turn off the urine collection using the same record.† One can do the same thing less conveniently by not using the minus sign in the observation record, but using it in a subsequent other-type event record with the same time data item.

With a reset or reset-dose event record the on-off status of each compartment is reset to its initial status, and then if the compartment data item is nonzero, the designated compartment is turned on or off according to whether the data item is plus or minus (it must be plus with a reset-dose event record).

I. The Prediction Compartment (PCMT) Data Item

NONMEM expects PRED to compute a prediction for each *data record*, whether or not the data record contains an observation. These predictions appear in all tables of data records, and they can appear on scatterplots. For example, one may want to scatterplot predictions versus time. Other-type event records can then be included in the data set with times of interest which are not observation times. For all data records with CONT data item 1, PREDPP produces a prediction of 0. For the last data record of an observation event record, the prediction is always either the scaled drug amount in the observation compartment (see section III.F.1), or a modification of this quantity (see section IV.B.2). For the last data record of a non-observation record, the prediction is the prediction that would result for an observation of the default observation compartment (see section H), unless otherwise specified in the event record by

† Output-type compartments have been part of PREDPP since the first version, but the use of a minus sign for CMT on an observation record was not mentioned. An output-type compartment is initally off, may be turned on and off, and may not receive a dose. It must be turned on with an other-type event record in order to start accumulating drug.

means of the prediction compartment data item.

The prediction compartment data item (PREDPP data item no. 8) may appear or be missing. If it has the value 0 in a non-observation event record, then the prediction for the last data record of the event record is the prediction that would result for an observation of the default observation compartment. Otherwise, the PCMT data item is a compartment number, and the prediction for the last data record of the event record is the prediction that would result for an observation of the specified compartment. Thus predictions for different compartments can be displayed.

By convention, the number of the output compartment may also be given by the PCMT data item as the number 1000 (this is true regardless of the type of the event record). With small models (number of compartments <= 99), this number may also be given as 100.

J. The Call (CALL) Data Item

The PK and ERROR subroutines need not be called with every event record (see sections III.H and IV.C). Calling-protocols limiting calls to PK or ERROR are called <u>call-limiting protocols</u>. When such protocols are implemented, one can use the call data item to force a call to PK or ERROR, or to both, with any given event record. This can be useful, for example, when except for a few individual records, none of the PK parameters depend on concomitant variables whose values vary within an individual record. Then one can limit calls to PK to once per individual record, but for the few individual records where this limitation is inappropriate, one can force a call to any event record where some concomitant variable assumes a different value from that in earlier event records.

The CALL data item (PREDPP data item no. 9) may appear in event records, or be missing. If it appears it has the values 0, 1, 2, 3, or 10. If it has the value 0 in an event record, then calls to PK and ERROR with the record occur according to the calling-protocols implemented with PK and ERROR. If it has the value 1, then a call to PK is determined according to the calling-protocol implemented with PK, and a call to ERROR occurs. If it has the value 2, then a call to ERROR is determined according to the calling-protocol implemented with ERROR, and a call to PK occurs. If it has the value 3, then calls to both ERROR and PK occur.

It may have the value 10 with ADVAN9. Then a call to ADVAN9 occurs whether or not the TIME data item is defined, and whether or not AES has used IDEFA(2) to limit calls to ADVAN9 (See Chapter VI, Section E.B). May be combined with values 1, 2, and 3. e.g., CALL = 11 forces calls to both ADVAN9 and ERROR with the event record.

It is not an error for the CALL data item to be positive when calls to either PK or ERROR are limited.

K. The Additional Dose (ADDL) Data Item

Often, a number of doses are given in series, each exactly like the other, and spaced one from the other by a constant interdose interval. Such a series can be specified with a series of dose event records. Such a series can also be specified on a single dose event record by using the addiional dose data item. The dose specified on the dose event record is called the <u>initiating dose</u>. The additional dose data item contains the number of doses in the series, excluding the initiating dose. The interdose interval data item (see section G) is used to specify the constant interdose interval. If the initiating dose is lagged, the additional doses are also lagged (see section III.F.6). The term <u>additional dose time</u> refers either to the time an additional dose is given, if the dose is not lagged, or to the lagged time at which the dose actually enters the system, if the dose is lagged. Such a time is a nonevent dose time (see section III.B.2). At each additional dose time a dose like the initiating dose automatically enters the system. Until such a time as all the additional doses have entered, or until the last event time in an individual record is attained, whichever comes first, the dose record initiating the dose series is said to be <u>active</u>. Afterwards, the record is said to be <u>inactive</u>. (The additional dose data item can also be used when all the data are from a single subject, and then the term 'last event time in an individual record' should be understood to mean the last event time in the data set. This is a consequence of the single-subject assumption; see section IV.A.) There can be at most 10 active dose event records at a time.

Clearly, use of the additional dose data item can make the specification of regular dose regimens easier. One disadvantage, though, is that since an additional dose time is not an event time, the administration of a particular additional dose will not be expressed explicitly in a NONMEM table or scatterplot, so a little care may need to be exercised upon interpreting these outputs.

The ADDL data item (PREDPP data item no. 11) may appear in event records, or be missing. If it has the value 0 in a dose or reset-dose event record, then no additional doses are given; just the initiating dose is given. If it has the value n > 0, then n additional doses are given. In this case, then the II data item must also be nonzero.

The ADDL data item must be 0 when the initiating dose is a steady-state infusion. However, the ADDL data item can be nonzero when the initiating dose is any other type of steady-state dose. In this case, after the steady-state dose, an additional number of (transient) doses just like the steady-state dose are given. (Recall that the steady-state dose is just the last of many doses identical to it and *leading* to a steady-state.) On the other hand, any steady-state dose event record with SS data item equal to 1 will deactivate any previous dose event record which happens to be active. Also, a reset and a reset-dose event record will deactivate any previous dose event record which happens to be active. Also, if a compartment is turned off with an other-type event record (see section H), this deactivates any previous dose event record with a dose to this compartment which happens to be active.

If the ADDL data item appears, then it must be 0 in observation, other-type, and reset event records.

L. Examples

L.1 Example I: Population Data

In this section example I, described in section III.L.1, is continued. In particular, the data set is described in greater detail here.

The first several individual records are shown in Figure 15 along with the NONMEM control stream in Figure 16. (A NM-TRAN control stream is shown in Figure 17.) Each event record consists of a single data record. The data items in each data record are: the ID, TIME, AMT, WT, APGR, CP, EVID, and MDV data items, where WT and APGR label the weight and APGAR score data items, and CP labels the DV data item, i.e. the observed plasma concentration. (NM-TRAN automatically generates EVID and MDV data items; so these are not listed in the NM-TRAN $INPUT control record.) Since the RATE data item is missing, all doses are understood to be instantaneous bolus doses. Note that the AMT data item is blank in observation records and that the CP data item is (though it need not be) blank in dose records. One might think that both the WT and APGR data items could be blank in all but the first event record of each individual record. After all, PK is only being called with the first event record of each individual record. However, both of these data items are being scatterplotted individually against residuals and weighted residuals. So they must be nonblank in at least the observation records as well.

The important thing to notice about the NONMEM control stream is that the indices of the 3 types of PREDPP defined data items appearing in event records (EVID, TIME, and AMT data items) are set in the INDEX control record. In this regard note that the integer 11 appears in field 4 of the ITEM record, so that the indices of the other 8 types of PREDPP defined data items not appearing in the event records are to be explicitly read as blanks, i.e. as 0's. (The same effect can be achieved by placing a 3 in field 4 since the INDXS array is zeroed out before values on the INDEX record are read into it.)

The first one or two pages of NONMEM output consist of a report concerning the information supplied on NONMEM control records, the so-called problem summary. Some illustrations of problem summaries are given in Guides I & II. The problem summary resulting from the data and control stream of this example is given in Figure 18. With PREDPP, the NONMEM problem summary is followed by two other summary pages output by PREDPP. For this example these are shown in Figure 19. They summarize information supplied by the user's choices of ADVAN and TRANS, and by the routines PK and ERROR at ICALL=1. When routines TOL and DES are included, the summary includes information from these subroutines.

Figure 19 contains the statements

```
ERROR IN LOG Y IS MODELED
DURING SIMULATION, ERROR SUBROUTINE CALLED WITH EVERY EVENT RECORD.
OTHERWISE, ERROR SUBROUTINE CALLED ONCE IN THIS PROBLEM.
```

The implementation by NM-TRAN of error model (4)

```
Y=F*EXP(EPS(1))
```

is discussed in IV.C.†

In section III.L.1 two PK codes are shown: one that returns typical values and typical first-partial derivatives, and one that returns subject-specific values amd subject-specific first partial derivatives. The second code can be used to obtain post-hoc estimates of the η's. A NONMEM control stream that requests these estimates and displays them is shown in Figure 20. (A NM-TRAN control stream is shown in Figure 21.) In particular, scatterplots of both η_1 and η_2 versus both weight and APGAR score are requested.

L.2 Example II: A Mixture Model

In this section example II, described in section III.L.2, is continued. The NONMEM control stream is shown in Figure 22. (A NM-TRAN control stream is shown in Figure 23. Figure 23 differs from earlier versions of the guide. Changes to $PK and use of $MIX were described in chapter III. In addition, $THETA has slightly different estimates of the initial estimates of θ_2 and θ_4 to make the sub-problems distinct, and the $EST record includes options METHOD=1 and INTERACTION. With these options NONMEM 7.4 obtains very similar final estimates as those from NONMEM IV described in previous versions of this guide.)

The variance-covariance of η_1, η_2, quantifying the random interindividual variability of the first subpopulation, is constrained to equal the variance-covariance of η_3, η_4, quantifying the random interindividual variability of the second subpopulation. This homoscedastic assumption has the effect of suppressing the bimodality character of the distribution on (CL,Vd). Indeed, the final parameter estimates (θ_1 = .00598, θ_2 = 3.60, θ_3 = 1.26, θ_4 = 2.69, θ_5 = .893 ω_{11} = .0982, ω_{12} = .073, ω_{22} = .0714) describe a distribution that is very right skewed, and essentially unimodal.‡ A scatterplot of EST versus weight is requested. EST is a PK-defined item described in section III.L.2. It is an estimate (1 or 2) of the subpopulation of which an individual is a member. The scatterplot is shown in Figure 24. (Since there are 59 individuals in the data set, but since EST has the same value with every data record in an individual record, there are only 59 points in the plot. However, the characters used for the plotted points indicate the level of overstriking due to multiple data records per individual record.) It appears that the individuals with lower weights are estimated to be from one subpopulation and those with higher weights from another.

---

†Note that with versions of NONMEM previous to NONMEM IV, the implementation in the generated ERROR subroutine was that of figure 9, and figure 19 included instead

```
ERROR SUBROUTINE CALLED WITH EVERY EVENT RECORD.
```

‡ These values are from NONMEM 7.4. They differ slightly from values in earlier editions of this guide.

L.3 Example III: Single-Subject Data

In this section example III, described in section III.L.3, is continued. The NONMEM control stream with the data embedded is shown in Figure 25. (A NM-TRAN control stream is shown in Figure 26.) Each event record consists of a single data record. The data items in each data record are: the AMT, TIME, CP, EVID, MDV, and ID data items, where CP labels the DV data item, i.e. the plasma concentration. (NM-TRAN automatically generates EVID, MDV, and ID data items; so these are not listed in the NM-TRAN $INPUT control record.) Since the RATE data item is missing, all doses are understood to be instantaneous bolus doses. Note that the AMT data item is blank in observation records and that the CP data item is (though it need not be) blank in the dose record.

The important thing to notice about the NONMEM control stream is that the indices of the 3 types of PREDPP defined data items appearing in event records (EVID, TIME, and AMT data items) are set in the INDEX control record. In this regard note that the integer 11 appears in field 4 of the ITEM record, so that the indices of the other 8 types of PREDPP defined data items not appearing in the event records are to be explicitly read as blanks, i.e. as 0's. (The same effect can be achieved by placing a 3 in field 4 since the INDXS array is zeroed out before values on the INDEX record are read into it.)

Unlike the control stream given for this problem in Guide I, section C, ID and MDV data items appear. These data items are always required when using PREDPP. The ID data items are different for different observation event records. In the case of data from a single subject different observation event records must be contained in different individual records. Individual records do not correspond on a one-to-one basis to animal or human subjects.

The problem summary resulting from the control stream of this example is given in Figure 27. The summary pages from PREDPP are given in Figure 28.

L.4 Example IV: Single-Subject Pharmacodynamic Data

In this section example IV, described in section III.L.4, is continued. A NONMEM control stream with the data embedded is shown in Figure 29. This control stream may be used with the ERROR subroutine shown in Figure 13. (A corresponding NM-TRAN control stream is shown in Figure 30.) It is very similar to the control stream for example III shown in Figure 25.

Another NONMEM control stream is shown in Figure 31. (A corresponding NM-TRAN control stream is shown in Figure 32.) Note that Figure 32 uses η-variables but does not contain an $OMEGA record. Figure 31 contains the record

```
DIAG   2
```

This tells NONMEM to obtain the initial estimate of OMEGA.

This control stream may be used with the ERROR subroutine shown in Figure 14, which defines items to be displayed. These are Cp and Ce concentrations. The four scatterplots involving these items are given in Figures 33-36. The scatter in the plot of Ce versus Cp results from the non-steady-state nature of the experiment; a single dose was administered to the subject. If each of the points of the scatterplot had been obtained at a steady-state, after multiple dosing, the points would fall on the line of identity. Although this plot looks like a hysteresis loop, it is not because the points of the loop are not time-ordered.

VI. Other User Subroutines

A. INFN

There is an INFN routine supplied on the NONMEM distribution medium that must be linked with the other routines for a NONMEM-PREDPP load module; see section VII.A. It is a "dummy" routine; it does nothing. It may be replaced by a user-supplied INFN routine that allows the user to carry out any of his own programmed computations at both the beginning of a problem and again, at the ending of the problem. At the beginning of the problem PREDPP calls INFN -- the initialization call , and at the ending of the problem PREDPP calls INFN -- the finalization call . At each call, the user has read-write access to his data via use of the NONMEM utility routine PASS described in Guide II. Thus data can be transgenerated, and additional data items can be produced at both the beginning and ending of a problem. Since the finalization call actually occurs before the Table and Scatterplot Steps, new data items generated by INFN at this call can be tabled and scatterplotted.

The preface to INFN may be

```
SUBROUTINE INFN(ICALL,THETA,DATREC,INDXS,NEWIND)
USE SIZES,     ONLY: DPSIZE,ISIZE
USE NMPRD_INT, ONLY: NWIND
INTEGER(KIND=ISIZE) :: ICALL,INDXS,NEWIND
REAL(KIND=DPSIZE) :: THETA
REAL(KIND=DPSIZE) :: DATREC
DIMENSION :: THETA(*),DATREC(*),INDXS(*)
```

The argument ICALL is 1 or 3, according to whether the call to INFN is the initialization or finalization call. A value ICALL=0 indicates that the call is a special one occuring before the initialization call of problem 1, allowing initialization to take place in a multiple problem run.

The order of calls to INFN, PK, and ERROR is:

INFN	ICALL 0
ERROR	ICALL 1
INFN	ICALL 1
PK	ICALL 1
INFN	ICALL 3

Code for a generated INFN may be specified by an $INFN block of abbreviated code. If there is abbreviated code in the $PK and/or $ERROR blocks that tests for ICALL=0, ICALL=1, or ICALL=3, this code is moved by NM-TRAN to the INFN routine as if it had been coded explicitly as part of an $INFN block. Such code is called $PK-INFN and $ERROR-INFN code, respectively.

The arguments THETA and DATREC function just as they do in any PRED routine when the argument ICALL is 0, 1, or 3. In particular, when ICALL=1, THETA is the array of initial estimates of θ for the problem, and when ICALL=3, THETA is the array of final estimates of θ for the problem. When ICALL=0, THETA is the array of initial estimates of θ for problem 1. When ICALL=1 or 3, the DATREC array initially contains the first data record of the data set for the problem, but using PASS, the contents of DATREC are replaced by other data records of the data set for the problem. (When ICALL=0, DATREC is initially the first data record of the data set for problem 1, but using PASS, the contents of DATREC are replaced by other data records for problem 1.)

Note that NM-TRAN generated code for $INFN will contain

```
USE NMPRD_REAL,ONLY: PASSRC
```

and will use PASSRC rather than DATREC. They are the same variable.

Note that subroutine argument NEWIND should not be used. Variable NWIND in module NMPRD_INT is used instead. NWIND is an integer variable acting like the NEWIND variable described in Guide I. It changes value during a pass through the data using PASS. It assumes one of 3 values: 0 if the data record is the first of the entire data set, 1 if the data record is the first data record of an individual record (other than the first individual record), and 2 if the data record is other than the first data record of an individual record. (A description of individual records with single-subject data is given in chapter II.)

When NM-TRAN abbreviated code is used for the $INFN block, the abbreviated code may use variable name NEWIND. The generated code in FSUBS will have variable name NWIND instead.

In Figure 37 the declarations are somewhat different. The subroutine argument is named NEWIN; the name NEWIND is assigned to NWIND. Thus, the variable name NEWIND may be used in the Fortran code. This is a matter of style. The declarations are:

```
SUBROUTINE INFN (ICALL,THETA,DATREC,INDXS,NEWIN)
USE NMPRD_INT, ONLY: NEWIND=>NWIND
INTEGER(KIND=ISIZE) :: NEWIN
```

See also NEWL2, below.

The user should not use PASS to modify either the ID or MDV data items.

An initialization call can be used for any sort of initialization. For example, user-arrays in the routines PK and ERROR can be initialized if these arrays are stored in module PRINFN (see below). Also, interpolated values of a concomitant variable can be computed for event records in which values are missing, e.g. other-type event records that have been included so that predictions can be obtained at the times in these records. This could also be done in PK or ERROR, but then this would be done with *every* call to these routines; if done in INFN, the computation is done once only.

An example of a user-supplied INFN routine is given in Figure 37. This INFN routine can be used to obtain linearly interpolated values of an independent variable V for those event records in which a value is missing. Simple linear interpolation may not be adequate for all data situations. With this INFN it is assumed that there is one data record per event record. It is also assumed that each individual record has no fewer than 2 event records with measured values of V and that every event record has a data item which the user might call the missing independent variable data item (which assumes values defined in the comment statements of the code). The routine uses PASS to pass through the data twice: once, to store all pairs of values for time and V from event records with measured values of V, and again, to store the interpolated values in records with missing values of V.

The INFN routine of Figure 37 is also in the help Guide VIII as infn1.exa "INFN_INTERPOLATION EXAMPLE 1". A version using $INFN abbreviated code is also in the help Guide VIII as infn2.exa "INFN_INTERPOLATION EXAMPLE 2".

The one-dimensional array, INDXS, functions in the way described in Guide I, section C.4.1. The user places integers into this array, using the INDEX control record. These integers are then available to PREDPP and therefore to INFN. For further details see section III.C where the use of INDXS is illustrated with subroutine PK.†

The finalization call can use the NONMEM utility routine GETETA to obtain conditional estimates of the η's (see section III.E.2). When used in conjunction with PASS, the values returned for the η's with each call to GETETA are appropriate for the individual whose data record is currently in DATREC.

When NM-TRAN abbreviated code is use, the following code can be used to loop through all records of the data set. The generated code will contain the necessary code to call PASS and GETETA.

† The INDXS array cannot be used with NM-TRAN abbreviated code.

```
DOWHILE (DATA)
 ...
ENDDO
```

A.A Global Input Variables

INFN can access variables other than though its argument list. The following sections list some of the variables of interest. The help Guide VIII can be used to obtain the exact declarations and all variables of interest.

USE NMPRD_INT , ONLY : **NWIND**

NWIND is an integer variable acting like the NEWIND variable described in Guide I. It is discussed above.

USE NMPRD_REAL , ONLY : **PASSRC**

PASSRC contains the data record at ICALL values 0, 1 and 3. PASSRC changes in conjunction with calls to PASS, at which time transgeneration is permitted. Thus, PASSRC may be both an input and an output variable. The DATREC argument of subroutine INFN is in fact PASSRC from module NMPRD_REAL, so either name may be used in a user-supplied INFN routine. When NM-TRAN abbreviated code is used, neither PASSRC nor DATREC may be referred to explicitly. Instead, use the name of the input data item of interest.

USE NMPRD_INT , ONLY : **NEWL2**

This is the NEWL2 variable as described for ERROR.
NEWL2 = 1 if the data record is the first of an L2 record.
NEWL2 = 2 otherwise.
NEWL2 changes value in conjunction with calls to PASS.

USE NMPRD_REAL , ONLY : **ETA**

At ICALL==3, these are the Posthoc (conditional) estimates of ETA. They change value with calls to PASS and iterations of the DOWHILE (DATA) loop, above.

USE ROCM_INT , ONLY : **IERE**=>IEST_ERR , **IERC**=>ICOV_ERR

The return codes from the Estimation and Covariance Steps, respectively.

USE ROCM_REAL , ONLY : **IIDX**=>DIDVALX

USE ROCM_REAL , ONLY : **CNTID**=>OFV_IND

IIDX gives values of the ID data item. CNTID gives values of the individual contribution to the objective function for the corresponding values of IIDX. E.g., IIDX(n) is the ID data item for the nth. individual record, and CNTID(n) is the contribution to the objective function for the nth. individual record. These values should only be displayed at ICALL = 3 (finalization block).

USE ROCM_REAL , ONLY : **MIXP**

USE ROCM_INT , ONLY : **MIXNUM**=>MIXCALL , **MIXEST**=>IMIXEST

When ICALL=3 MIXEST is the index of the subpopulation estimated to be that from which the individual's data most probably arises. MIXEST changes value in conjunction with calls to PASS. MIXNUM has the same values as MIXEST. MIXP are the values of MIXP

USE ROCM_REAL , ONLY : **OBJECT**

The final value of the objective function.

The following variables may be referenced as right-hand quantities in initialization and finalization blocks. Note also that unsubscripted arrays may appear in a WRITE statement.

USE ROCM_REAL , ONLY : **OMEGAF , SIGMAF**

OMEGAF(i,j) = zero, initial, or final value of omega(i,j), according to the current value of ICALL (0, 1, 3).
SIGMAF(i,j) = zero, initial, or final value of sigma(i,j), according to the current value of ICALL (0, 1, 3).

Note that there is no THETAF variable; the argument THETA of INFN has the appropriate values of THETA. With NM-TRAN abbreviated code, these variables are coded as OMEGA and SIGMA.

(See **Parameter Values: Initial and Final**).
(See **write print**.)

The following variables may be referenced as right-hand quantities in finalization blocks (ICALL==3).
(Note also that unsubscripted arrays may appear in a WRITE statement.)

USE NMPR_REAL , ONLY : **SETHET**=>SETH , **SEOMEG**=>SEOM , **SESIGM**=>SESIG,
SETHETR=>SETHR

SETHET(i) = the standard error of the estimate of internal theta(i).
SETHETR(i) = the standard error of the estimate of reported theta(i) (SETHET=SETHETR if $THETAR record not used).
SEOMEG(i,j) = the standard error of the estimate of omega(i,j).
SESIGM(i,j) = the standard error of the estimate of sigma(i,j).

USE NMPR_REAL , ONLY : **THSIMP**=>THET_P , **OMSIMP**=>OMEG_P , **SGSIMP**=>SIGM_P,
SIMTHPR

THSIMP(i) = the "prior" for theta(i) in internal domain.
THSIMPR(i) = the "prior" for thetar(i) in reported domain. THSIMP=THSIMPR If $THETAR record not used.
OMSIMP(i,j) = the "prior" for omega(i,j).
SGSIMP(i,j) = the "prior" for sigma(i,j).

Values of THETA, OMEGA and SIGMA that are produced during a Simulation Step using the user-supplied routine PRIOR.

The following group are NONMEM "counter variables."

USE ROCM_INT , ONLY : **NIREC**=>NINDREC , **NDREC**=>NDATINDR

USE NMPRD_INT , ONLY : **NPROB**, **IPROB**

USE ROCM_INT , ONLY : **NREP** , **IREP**=>NCREP

USE ROCM_INT , ONLY : **LIREC**=>NDATPASS

USE ROCM_INT , ONLY : **NINDR , INDR1 , INDR2**

INTEGER(KIND=ISIZE), POINTER: **S1NUM**, **S2NUM**, **S1NIT**, **S2NIT**, **S1IT**, **S2IT**

They may be used on the right in $INFN or INFN. They change values when PASS is called, if appropriate. Counters include (in the order above): record counters; problem iteration counters; super-problem iteration counters; simulation repetition counters; number of data records in the individual record; number of individual records in the data set containing an observation record, and their indices. More information can be found in Chapter III.

USE ROCM_REAL , ONLY : **PRED_**, **RES_**, **WRES_**

These variables are the values of the PRED, RES, WRES items displayed in tables. They may be used on the right in $INFN or INFN when ICALL=3. They change values when PASS is called. They may be used in NM-TRAN abbreviated code. If the INFN routine is user-coded, the variables are located in the APPND array. The declarations are as follows:

```
USE ROCM_REAL, ONLY: APPND
PRED_=>APPND(001)
RES_=>APPND(002)
WRES_=>APPND(003)
```

In addition, the following related variables may be used and are also located in APPND:

```
IWRS_ IPRD_ IRS_
NPRED_ NRES_ NWRES_
NIWRES_ NIPRED_ NIRES_
CPRED_ CRES_ CWRES_
CIWRES_ CIPRED_ CIRES_
PREDI_ RESI_ WRESI_
IWRESI_ IPREDI_ IRESI_
CPREDI_ CRESI_ CWRESI_
CIWRESI_ CIPREDI_ CIRESI_
EIWRES_ EIPRED_ EIRES_
EPRED_,ERES_,EWRES_,NPDE_,NPD_,ECWRES_
OBJI_
```

See nm730.pdf and help Guide VIII for more information on these variables. Note that OBJI_ is the value of the individual contribution to the objective function and can also be found in the appropriate positition of CNTID, above.

REAL(KIND=DPSIZE) , POINTER : : **DEN_ , CDEN_(:)**

USE ROCM_REAL , ONLY : DEN_NP

DEN_=>DEN_NP(1)

CDEN_=>DEN_NP(2:)

These values are computed by NONMEM when the Nonparametric step is performed and marginal cumulatives are requested. DEN_ is the nonparametric density. CDEN_(n) is the marginal cumulative value for the nth. eta. At ICALL==3, they change value with calls to PASS and iterations of the DOWHILE (DATA) loop, above.

USE NMPR_REAL , ONLY : **PR_Y**

USE NMPR_REAL , ONLY : **PR_CT**

See Chapter IV.

A.B Global Output Variables

USE NMPRD_REAL , ONLY : **PASSRC**

PASSRC is both an input and an output variable. See above.

USE NMPR_INT , ONLY : **SKIP_**=>SKIP

SKIP_ controls premature termination of a problem (with subproblems), superproblem or superproblem iteration. May be set at ICALL=3.

A.C Miscellaneous Global Variables

PRINFN is a global module for INFN-defined variables. It is meant to be used for communication with other other blocks of abbreviated code or with user-written codes. The declaration is:

```
MODULE PRINFN
 USE SIZES, ONLY : DPSIZE,DIMTMP
```

```
 REAL(KIND=DPSIZE), TARGET, DIMENSION (DIMTMP):: ITV
END MODULE PRINFN
```

where the size is given in SIZES.f90 by PARAMETER (DIMTMP=500) and DIMTMP can be changed with the $SIZES record. PREDPP is recompiled by default with every NONMEM run, and so PRINFN is always sized to the current value of DIMTMP.

Within a user-supplied code, the declaration

USE PRINFN , ONLY : TLCOM=>ITV

makes it possible to use elements of TLCOM, e.g., TLCOM(1), TLCOM(2).

When NM-TRAN is used, the generated code is a little more complicated. For example, suppose the $INFN block contains

```
$INFN
SUM=0
```

FSUBS will contain `MODULE INFNP` which contains:

```
USE PRINFN, ONLY: ITV
REAL(KIND=DPSIZE), DIMENSION (:),POINTER ::TLCOM
REAL(KIND=DPSIZE), POINTER ::SUM
```

INFNP also contains a subroutine ASSOCPRINFN containing statements such as

```
TLCOM=>ITV
SUM=>TLCOM(00001)
```

ASSOCPRINFN is called by INFN, PK, ERROR and other subroutines in FSUBS. SUM can be used in all these routines and the generated code will use the user-supplied variable SUM.

Note that PRINFN is not initialized or modified by NONMEM or PREDPP. Unlike module NMPRD4, the variables in the module may be initialized or modified by $INFN at ICALL values 0 or 1 or 3, and will retain whatever values they are given.

A.D Displaying INFN-Defined Variables

When NM-TRAN abbreviated code is used, variables that are defined first in $INFN or in $PK-INFN or $ERROR-INFN blocks are listed in PRINFN. These variables cannot be displayed in tables or scatterplots. If they are to be displayed, WRITE statements can be used, or their values can be assigned to variables that are listed in module NMPRD4.

A.E Other Subroutines That May Be Called

The NONMEM utility routine SUPP is used to suppress portions of the NONMEM output report. SUPP may be called only when ICALL is 0, 1, or 3.

B. MODEL

Some ADVAN routines, specifically those implementing general linear or nonlinear kinetic models, require an additional user-supplied routine called MODEL which allows the user to conveniently specify important aspects of the model. MODEL is called once only, at the beginning of a NONMEM-PREDPP run. Its preface is

```
SUBROUTINE MODEL (IDNO,NCM,NPAR,IR,IATT,LINK)
USE PRMOD_CHAR, ONLY: NAME
USE SIZES,     ONLY: ISIZE
INTEGER(KIND=ISIZE) :: IDNO,NCM,NPAR,IR,IATT,LINK
DIMENSION :: IATT(IR,*),LINK(IR,*)
```

The user may have several MODEL routines and use different ones for different runs. The routine used in a particular run can be identified from the output. For this to happen, an integer identification value should be assigned by MODEL to the variable IDNO. This value will be printed on the first PREDPP problem summary page. When NM-TRAN abbreviated code is used, IDNO is set to 9999.

The total number of compartments to be used, excluding the output compartment, is specified by assigning this number to NCM. This number cannot exceed 29, unless the $SIZES record is used to increase PC to a larger value than 30. The largest possible value of PC is 999, in which case NCM may be 998.

Basic PK parameters exist for general linear and nonlinear models as they do for more specific models. A TRANS routine is used which translates the set of basic PK parameters whose values are computed in the PK routine to a set of parameters used internally by the ADVAN routine. The maximum number of basic PK parameters to be used in any TRANS routine (M_B; see section III.G) is specified by assigning this number to the variable NPAR. NPAR may be 0, e.g., when only THETA and data record items are used in differential equations. The maximum number of basic PK parameters plus the number of additional parameters whose row indices are given explicitly in PK (at ICALL=1) cannot exceed PG, a constant in SIZES.f90 which is given by PARAMETER (PG=50+PCT), where PCT is the maximum number of model event time parameters given by PARAMETER (PCT=30) Both these parameters can be changed with the $SIZES record.

Values of compartment attributes must be given for each compartment (other than the output compartment, for which these values are fixed by PREDPP; see below). There are 5 required attributes; one should choose a value for each of these attributes for each compartment. To choose these values for the Ith compartment first answer these 5 defining questions:

Attribute	Question
1	Is the initial status of the compartment on?
2	May the compartment be turned on and off?
3	May the compartment be a dose compartment?
4	Is the compartment the default observation compartment?
5	Is the compartment the default dose compartment?

Values of attributes are given by assigning values to IATT. IATT is a two-dimensional array of 4 byte integers. If the answer to the Jth question for compartment I is 'yes', then set IATT(I,J)=1; if the answer is 'no', then set IATT(I,J)=0.

There is really no reason for answering 'no' to question 3 for compartment I other than to activate a check for dose and reset-dose event records with CMT data item equal to I. Similarly, the only reason for answering 'no' to question 2 for compartment I is to activate a check for other and reset event records with CMT data item equal to -I. If the Ith compartment is an equilibrium compartment (see below), then the answer to question 3 must be 'no'; one cannot put a dose into an equilibrium compartment.

An <u>output-type</u> compartment is one that has initial status off; no doses allowed; may be turned on and off (questions 1-5 are 'off', 'yes', 'no', 'no', 'no').† It must be turned on with an other-type event record in order to start accumulating drug. There may be more than one such compartment, in addition to the default output compartment. For such compartments, the value of CMT may be negative on observation records to obtain an observation and turn the compartment off.

† When NM-TRAN's $MODEL record is used, a clause such as `COMP=NAME`
defines a compartment with name "NAME" and with attributes initial on, off/on allowed, dose allowed. In order to make this an output-type compartment, the attributes should be
`COMP=(NAME,NODOSE,INITIALOFF)`

Labels for the various compartments should also be given. Labels are displayed in the PREDPP problem summary; see the Compartment Attributes Table, under column heading FUNCTION (the label 'OUTPUT' is supplied by PREDPP). For example, see Figure 28. A label consists of 8 characters, including blanks. (With NONMEM 7.4, the maximum number of characters is given by SD in SIZES. The default is 30. If any label exceeds 8 characters, a wider table is generated.) The compartment name is given in a character array, NAME, listed in a PREDPP module. The appropriate declarations are:

```
USE PRMOD_CHAR , ONLY : NAME
```

The label for compartment I is stored in NAME(I), e.g.

```
NAME(2)='CENTRAL'
```

The attributes for the output compartment are fixed by PREDPP: initial status off; no doses allowed; may be turned on and off; is neither the default observation nor default dose compartment.

Figure 38 is an example of a MODEL routine that can be used to describe the one compartment linear model with first order absorption, just as this model is described in section VII.C.2. Figure 39 is an example of a MODEL routine that can be used to describe the three compartment linear model used with the Example IV developed in sections III.L.4, IV.G.4, and V.L.4. Both of these routines illustrate the use of the LINK argument which is discussed below. (An NM-TRAN $MODEL record corresponding to the MODEL routine of Figure 39 is as follows:

```
$MODEL COMP=(DEPOT,NOOFF,DEFDOSE) COMP=(CENTRAL NOOFF NODOSE)
   COMP=(EFFECT, NOOFF NODOSE DEFOBS)
```

It could be used in place of the $MODEL record of Figure 30. The information in the MODEL routine which is not explicitly specified in the $MODEL record is obtained from the $PK record.)

Most pharmacokinetic models involve only nonequilibrium compartments, compartments such that the amounts of drug in them can be given by a solution to a system of differential equations. However, one ADVAN routine - ADVAN9 - allows a model (a general nonlinear model) to be comprised of both nonequilibrium and equilibrium compartments. Equilibrium compartments are, roughly speaking, compartments such that at any time the amounts of drug in them can be given by a solution to a system of algebraic equations. ADVAN9 allows the user to define a model consisting only of equilibrium compartments, in which case ADVAN9 becomes a purely algebraic equation solver. The quantities A(i) may be thought of as unknowns to be determined rather than as "compartment" amounts. Records with EVID=1 and EVID=4 (dose, reset and dose) may not be present in the data set because doses cannot be placed in equilibrium compartments. Similarly, data items related to doses (RATE, AMT, SS, II, ADDL) may or may not be used in the data set. If used, they must contain null values. The TIME data item is optional. If TIME is used, ADVAN9 is called exactly as other ADVAN routines are called: when the value of TIME increases. If TIME is not defined, AES routine specifies the calling protocol for ADVAN9: call with every event record (the default) or once per individual record (see section E.C). If calls to ADVAN9 are limited, the CALL data item may be defined in the data set and given the value 10 to force additional calls to ADVAN9 (see chapter V, section J).

There are two additional attributes to which values must be given only when ADVAN9 is used. The first of these simply identifies the compartment as an equilibrium compartment or not. The second of these relates to the output compartment. The amount of drug in the output compartment at time t is given by A+B-C, where A is the total amount in the system at s, the time the compartment is turned on, B is the total amount of drug input to the system between times s and t, and C is the total amount in the system at time t. If an equilibrium compartment represents a subcompartment, i.e. the amount D of drug in the compartment is part of the amount of drug in another compartment, D should be excluded from A and C. The attribute in question is concerned with whether D should or should not be excluded from A and C. To choose values for the two attributes for the Ith compartment first answer:

Attribute	Question
8	Is the compartment an equilibrium compartment?
9	Should the total drug amount in the system interior exclude the amount in the compartment?

If the answer to the Jth question (J=8,9) for compartment I is 'yes', then set IATT(I,J)=1; if the answer is 'no', then set IATT(I,J)=0.

The internal parameters of a general linear model are the rate constants. The user assigns each rate constant a number, any number between 1 and the value assigned to NPAR. The numbering is established using the LINK array.

LINK is a two-dimensional integer array. If no drug can distribute from the Ith compartment to the Jth compartment, then one can assign 0 to LINK(I,J), or not bother assigning any value to LINK(I,J). In particular, LINK(I,I) should always be ignored. If drug can distribute from the Ith compartment to the Jth compartment, then one assigns to LINK(I,J) the number of the rate constant quantifying this (first-order) distribution. The LINK array should, in particular, account for the distribution of drug from compartment I to the output compartment. That is, if such distribution can take place, then a number should be assigned to LINK(I,NCM+1). The numbers of the rate constants need not start with the number 1, and they need not be consecutive, i.e. numbers may be skipped. Moreover, two rate constants can be assigned the same number; this forces their values and η-derivatives to be the same.

With NM-TRAN, there are two ways to specify LINK. When the $PK abbreviated code is used, link relationships are implied by variable names such as Kij or KiTj (where "T" stands for "TO"). When a user-supplied PK is used, the LINK relationships must be specified explicitly on the $MODEL record using options such as Kij.

With a general nonlinear model the user explicitly specifies a set of differential equations (see section C). These equations are parameterized in terms a set of internal kinetic parameters. The LINK array plays no role with a general nonlinear model. The internal parameters are numbered between 1 and the number assigned to NPAR. The rules concerning internal parameter numbering are the same as given above for rate constant numbering.

B.A Global Output Variables

USE PRMOD_CHAR ONLY : **NAME**

This is discussed above.

USE PRMOD_INT , ONLY : **ISSMOD**

Used for the Initial Steady State feature of PREDPP, with the general non-linear models (ADVAN6, ADVAN8, ADVAN9, ADVAN13, ADVAN14, ADVAN15). By default, the initial conditions (i.e., compartment amounts) are zero at the start of each individual record and after a reset record. Different initial conditions may be computed. This is done by setting ISSMOD (in a user-written MODEL routine) or using the I_SS option of the $MODEL record. The I_SS option causes the ISSMOD global variable to be set to the given value in SUBROUTINE MODEL. Values of ISSMOD are the same as I_SS in $PK (See Chapter III Section I.B).

C. DES

C.A DES General Description

This user-supplied subroutine is required by a general nonlinear model. (With ADVAN9, a $DES block may not appear when there are only equilibrium compartments.) Also, regardless of the model chosen, it is also required whenever the user chooses to compute steady-state kinetics using the SS6 routine; see section VII.B. Usually, though, SS6 is only used when steady-state kinetics are computed with a general nonlinear model, so that DES would be required in any event. (When the model is not a general nonlinear model and SS6 is used, then unless steady-state infusions are present in the data set, the only executable statement in DES need be: RETURN.)

DES allows the user to conveniently specify the system of ordinary differential equations describing the underlying kinetics (of the nonequilibrium compartments; see section B). Its preface is†

```
SUBROUTINE DES (A,P,T,DADT,IR,DA,DP,DT)
USE SIZES,     ONLY: DPSIZE,ISIZE
IMPLICIT REAL(KIND=DPSIZE) (A-Z)
INTEGER(KIND=ISIZE) :: IR
DIMENSION :: A(*),P(*),DADT(*),DA(IR,*),DP(IR,*),DT(*)
```

The system of differential equations can be expressed mathematically by

$$\frac{dA_i}{dt} = h_i(A, P, t) \quad i = 1, 2, \ldots, n \qquad (1)$$

where A is the vector of drug amounts in *all* the various compartments (other than the output compartment), P is the vector of internal PK parameters (see section B.) , t is time, and n is the number of compartments (excluding the output compartment and all equilibrium compartments; see section B). The parameter P_k is defined to be the kth internal parameter. It is a continuously acting parameter. An example of such a system is this.

$$n = 2$$

$$P = (Ke, Ka)$$

$$h_1(A, P, t) = -Ka \cdot A_1$$

$$h_2(A, P, t) = Ka \cdot A_1 - Ke \cdot A_2$$

i.e. the familiar one compartment model with linear elimination and absorption from a drug depot. In general the system need not be linear, or even homogeneous (i.e. the h functions can depend on t).

The arguments A, P, and T of DES are inputs to the routine and correspond to A, P, and t in (1). The GG array computed by the PK routine is passed to DES as P.‡ T is a "continuously-valued" variable,

† Note that with earlier versions of the guide there was no argument DT. This limited the way time T could be used in the model. If a rate parameter, duration parameter, or absorption lag time parameter was a random variable, then T could only be used only in logical conditions in DES. This restriction was lifted with NONMEM V. Argument DT is required and time T may be used freely in DES routines and in $DES abbreviated code.

‡ When NM-TRAN is used, the basic PK parameters are assigned the first positions in GG. Explicit Basic PK parameters are P(n) (nth basic PK parameter) Implicit basic PK parameters are PK-defined variables used also in $DES block (and AES block, if present). A user-written PK routine should also assign the first positions in GG to the P(i) that will be used by the DES routine (and AES routine, if any). The DES routine should not use positions in GG beyond the basic PK parameters. I.e., it should not use the additional

not the discretely-valued TIME data item.

The primary task of DES is to compute the values of the h functions and store these values in DADT. The value of h_I is stored in DADT(I).

In addition to computing the values of the h_i, DES must also compute the values

$$\frac{\partial h_i}{\partial A_j}(A, P, t) \quad i = 1,2,...,n \text{ and } j = 1,2,...,m$$

where m is the number of all compartments, *including equilibrium compartments* should they exist (see section B). With population data DES must also compute the values

$$\frac{\partial h_i}{\partial P_k}(A, P, t) \quad i = 1,2,...,n \text{ and all } k$$

and

$$\frac{\partial h_i}{\partial t}(A, P, t) \quad i= 1,2,...,n$$

The value of $\frac{\partial h_I}{\partial A_J}$ is stored in DA(I,J), the value of $\frac{\partial h_I}{\partial P_K}$ is stored in DP(I,K), and the value of $\frac{\partial h_I}{\partial t}$ is stored in DT(I).

In the above example:

$$\begin{array}{ll}
\frac{\partial h_1}{\partial A_1} = -\,\mathrm{Ka} & \frac{\partial h_1}{\partial A_2} = 0 \\
\frac{\partial h_2}{\partial A_1} = \mathrm{Ka} & \frac{\partial h_2}{\partial A_2} = -\,\mathrm{Ke} \\
\frac{\partial h_1}{\partial \mathrm{Ke}} = 0 & \frac{\partial h_1}{\partial \mathrm{Ka}} = -\,A_1 \\
\frac{\partial h_2}{\partial \mathrm{Ke}} = -\,A_2 & \frac{\partial h_2}{\partial \mathrm{Ka}} = A_1 \\
\frac{\partial h_1}{\partial t} = 0 & \frac{\partial h_2}{\partial t} = 0
\end{array}$$

A routine for implementing the example is given in Figure 40. The numbering of the compartments is that specified by the MODEL routine given in Figure 38. Since the LINK array plays no role with a general nonlinear model, the numbers set in the LINK array are ignored. Note that it is unnecessary to set 0's in either DA or DP or DT.

We emphasize that it is only necessary for the computations in DES to characterize the underlying kinetics. These computations are not to be concerned with matters of boundary conditions or drug input functions described on dose event records. Those matters are handled by PREDPP itself. However, the differential equations may include terms for endogenous drug production, and initial conditions A_0 may be specified in the PK routine (See chapter I, section I.B).

PK parameters such as Fi (biavailability franctions), because they are used by PREDPP.

ADVAN routines such as ADVAN6 use a subroutine from third party sources to do the integration. For example, ADVAN6 calls DVERK from IMSL, ADVAN13 calls LSODA, etc. These subroutines are the ones that call DES. They call DES with various values of T during the integration (i.e., the advance in time) from T1 to T2. T1 and T2 are beginning and ending event time. Typically, these are the times on a pair of event records, but the end time of an integration interval may also be a non-event time (i.e., the time of a lagged or additional dose), or a model event time (i.e., MTIME(i) for some i. T2 may also be a time associated with the termination of a zero-order bolus dose. The integrating subroutine may decide it has enough information after a call with a value of T that is not exactly T2.(It might be a little less or a little more.) After the advance, DES is called by the ADVAN routine itself (i.e., $DES statements are evaluated) at the exact value of the event time. See ISFINL, below.

For a given integration interval, the values of DADT(I) must be continuous. Fortran functions that are continuous, such as SIN and COS, may be used. The INT, MOD, and ABS functions (or other functions that introduce a discontinuity) should not be used in DES, especially if the end time of the interval happens to coincide with a change in DADT. Instead, use model event times MTIME in PK, and either set flag variables in PK that can be tested in DES, or use MPAST in DES.
Appendix III describes MTIME parameters in detail, and gives an example of the use of MPAST in DES for modelling EHC (Enterohepatic Recycling). It also describes where other important examples of modelling step functions and circadian rhythm may be found.

Appendix IV describes how the chain rule is used to obtain $\frac{\partial}{\partial \eta_j} \frac{dA_i}{dt}$.
The summation includes terms $\frac{d}{dP_k}(\frac{dA_i}{dt}) \cdot \frac{\partial P_k}{\partial \eta_j}$. But if some P is defined only in terms of THETA and EVTREC, then $\frac{\partial P_k}{\partial \eta_j}$ is always zero. By using THETA and EVTREC directly in DES and $DES, as described below, there is less time spent calculating terms that always zero, and this may improve run times.

The reader should see section III.K for a discussion about PRED-error recovery from PREDPP. DES can force an immediate return to NONMEM from PREDPP with a nonzero PRED error return code and accompanying user message. The required declarations are described in section III.K.2.

C.B Global Input Variables

USE PRMOD_INT , ONLY : **ICALL**=>ICALLD

ICALL has the same value as PK and ERROR's ICALL argument. At ICALL=1, DES may optionally store values in IDEFD (see below) in order to improve run time. Values may be stored in DADT, etc., as at other calls, but these values are ignored. At ICALL=1 a user-written DES routine that does not check ICALL will compute DADT, etc., using values that PK computed (at ICALL=1) with the first event record and compartment amounts that are 0. There is some risk of arithmetic difficulties if, for example, a compartment amount of 0 is raised to a power. If this occurs, the user should insert the following statement at the top of the routine (or $DES block):

```
IF (ICALL.EQ.1) RETURN
```

ICALL blocks testing for ICALL values 4 (simulation) and 5 (data averages) are permitted.

USE PROCM_INT , ONLY : **ISFINL**

A copying pass occurs when the $TABLE record is present, and specified PRED-defined items listed in NMPRD4. When NONMEM is performing simulation or a copying pass (COMACT>0), DES is called immediately after the advance to an event time or non-event time, with a value of T equal to this time, so that DES-defined variables listed in NMPRD4 are displayed with their final values for that advance. Global variable ISFINL is set by PREDPP to 1 on this final call to DES. ISFINL can be tested on a WRITE statement if such statements are used in DES. ISFINL is *not* set to 1 immediately after the advance to a model event time.

USE PROCM_REAL , ONLY : **DOSTIM , DOSREC**

DOSTIM and DOSREC are described in Chapter III, Section I.A. A DES routine may test DOSTIM and elements of DOSREC in a logical condition. It may use them on the right-hand side of an assignment statement. If DOSTIM is a random variable, DOSTIM must not be used in DES or $DES. However, DOSTIM may always be used in a $PK block or PK routine to define a random variable which may be used in the DES routine. With NM-TRAN, the $BIND record may be used, but it only affects the DOSREC for additional and lagged doses (See chapt III).

USE PROCM_REAL , ONLY : **EVTREC**

Data record items may be used explicitly in DES and $DES abbbreviated code (rather than obtained as PK parameters). EVTREC is the current event record (the event record to which the system is being advanced). This may improve run time.

USE PROCM_REAL , ONLY : **THETA**

Elements of THETA may be used explicitly in DES and $DES abbreviated code (rather than obtained as PK parameters). This may improve run time.

USE ROCM_INT , ONLY : **MIXEST , MIXNUM , MIXP**

Mixture-model variables may be used in DES. For a description, see INFN, above.

C.C Global Output Variables

USE PRMOD_INT , ONLY : **IDEFD**

At ICALL=1, DES may optionally store values in IDEFD in order to improve run time.

IDEFD(1) may optionally be set by DES to indicate how many thetas it uses. Set to 0 if none. Otherwise, set to the index of the highest numbered theta used. PREDPP determines from the value stored in IDEFD(1) how many elements of theta to copy from its input argument THETA to THETA in PROCM_REAL. When IDEFD(1) is set by DES to -9, PREDPP copies all thetas in the problem to THETAS. IDEFD(1) defaults to -9, which does no harm, but may cause the run to be slower than necessary.

IDEFD(2) is the full/compact flag for DES. DES sets this to describe the format of DA and DP and DT on subsequent calls to DES.
IDEFD(2)=0 DES will return compact arrays.
IDEFD(2)=1 DES will return full arrays.
Section C above describes DA, DP and DT with full arrays. Appendix IV describes compact arrays. It also describes the implementation of 2nd. partial derivatives for the Laplacian method. Compact arrays are required with the Laplacian method; they are optional otherwise. With a large system, run times may be improved when compact arrays are used.

With NM-TRAN, DES=FULL or DES=COMPACT can be specified as an option of the $ABBREVIATED record. Compact arrays are the default except for ADVAN9 and ADVAN15. With ADVAN9 and ADVAN15, full arrays are required, and the $ESTIMATION record option NUMERIC is required with the Laplacian method.†

† With versions of NONMEM 7 prior to 7.4, this was true of ADVAN13 as well.

C.D Displaying DES-Defined Items

A value stored in a variable (or array element) V in DES may be displayed in a table or scatterplot. To accomplish this, module NMPRD4 must be defined in DES, and V must be listed in NMPRD4. The values of DES-defined variables displayed for the first event record of the data set are 0's. The values displayed for the first event record of subsequent individuals are those from the last event record of the previous individual. This is because there is no call to DES with the first event record of the individual record. If the first event record is followed by records with the same value of TIME, there is also no call to DES and the displayed values are the same as for the first record. NM-TRAN has a warning about such values.

```
WARNING  48) DES-DEFINED ITEMS ARE COMPUTED ONLY WHEN EVENT TIME
INCREASES. E.G., DISPLAYED VALUES ASSOCIATED WITH THE FIRST EVENT RECORD
OF AN INDIVIDUAL RECORD ARE COMPUTED WITH (THE LAST ADVANCE TO) AN EVENT
TIME OF THE PRIOR INDIVIDUAL RECORD.
```

DES is called immediately after the advance to an event time or non-event time, with a value of T equal to this time, so that DES-defined variables listed in NMPRD4 are displayed with their final values for that advance. See also ISFINL (above).

Module NMPRD4 also provides a convenient place to store values of variables to be shared between DES and other user routines, and it is used thusly when these routines are generated from NM-TRAN abbreviated code (see Guide IV). (INFN-defined and declared variables are also shared between user routines.)

D. TOL

This user-supplied subroutine is required when using an ADVAN or SS routine that uses numerical techniques. These are the general nonlinear models (ADVAN6, 8, 9, 13, and, with NONMEM 7.4, 14 and 15), the one compartment model with Michaelis-Menten elimination (ADVAN10), and two particular steady-state routines (SS6,9); see section VII.C. The TOL subroutine changed significantly with NONMEM 7.4. The two versions are described separately.

D.A TOL (prior to NONMEM 7.4)

With earlier versions, TOL is called only once at the start of a run. TOL's preface is

```
SUBROUTINE TOL(NRD)
DIMENSION :: NRD(*)
USE SIZES, ONLY: ISIZE
INTEGER(KIND=ISIZE) :: NRD
```

The purpose of TOL is to set NRD(I) to the number of accurate digits that are required in the computation of the drug amount in compartment I, i.e. the relative tolerance . ADVAN9 has the capability of using different values of NRD for different compartments. However, all the other ADVAN and SS routines requiring TOL take the relative tolerance to be the same for all compartments; NRD(I), I > 1, is ignored, and only NRD(1) is used. Even with ADVAN9 the accuracy of a *steady-state* amount (computed by SS9) for compartment I, I > 1, is only controlled to be NRD(1) digits. The user will usually be content with a single relative tolerance which applies to all compartments. In this case, as a rule of thumb, one should begin by setting NRD(1) to n+1 or n+2, where n is the number of significant digits required in the parameter estimates, or with double precision, perhaps to n+2 or n+3. If one succeeds with this setting, one might try increasing NRD slightly.

Indeed, NRD can be a scalar, rather than an array appearing in a DIMENSION statement, in which case NRD is simply assigned a unique relative tolerance. Even with ADVAN9 this is true.

An example of a TOL routine for use with NONMEM 7.3 and earlier versions is given in Figure 41.

When NM-TRAN is used, TOL=n may be specified as an option on the $SUBROUTINES record. Subroutine TOL is generated by NM-TRAN. It will contain the statement
NRD(1)=n.

It is also possible to use the $TOL record, e.g.

```
 $TOL NRD=10
```

When $TOL is used for ADVAN9, different values of NRD may be specified for each compartment, e.g.,

```
 $TOL
   NRD(1)=5
   NRD(2)=10
```

TOL values may also be specified in the NONMEM control stream. With NM-TRAN, this may be done using the $ESTIMATION and $COVARIANCE records. The $COVARIANCE record has an option TOL=n. This can be used to set a different value of TOL for the Covariance Step. With ADVAN9, ADVAN13, ADVAN14, ADVAN15, the $ESTIMATION and $COVARIANCE record options ATOL=n may also be used. These can be used to set Absolute tolerance for the Estimation and Covariance Steps. The default is 12 (that is, accuracy is 10**(-12)).

D.B TOL (NONMEM 7.4)

With NONMEM 7.4, there are multiple calls to TOL during the run, for each NONMEM step. TOL's preface is

```
SUBROUTINE TOL(NRD,ANRD,NRDC,ANRDC)
USE SIZES, ONLY: ISIZE
INTEGER(KIND=ISIZE) :: NRD(0:*), ANRD(0:*), NRDC(0:*), ANRDC(0:*)
```

An example of a TOL routine for use with NONMEM 7.4 is given in Figure 41a.

Optional declarations with NONMEM 7.4:

```
USE NMPRD_INT, ONLY: IPROB
USE NM_BAYES_INT, ONLY: NM_STEP,BASE_STEP,EST_STEP,COV_STEP, &
 TABLE_STEP,SIML_STEP,INE_STEP, NONP_STEP
```

The purpose of TOL is to set the following variables. Any value that may be set in subroutine TOL may also be set using the NM-TRAN $TOL record.

NRD(I)
: The number of digits that are required to be accurate in the computation of the drug amount in compartment I, i.e., the relative tolerance. ADVAN 9, 13, 14, and 15 have the capability of using different values of NRD for different compartments. For compartments not specified, the tolerance of the last compartment specified will be used.

: However, all the other ADVAN routines requiring TOL take the relative tolerance to be the same for all compartments; NRD(I), I > 1, is ignored, and only NRD(1) is used. NRD(0) is the relative tolerance for the Steady State computations. If NRD(0) is not specified, NRD(1) is used.

: The value of NRD(1) can also be specified using $SUBROUTINES option TOL.
The value of NRD(0) can also be specified using $SUBROUTINES option SSTOL.

ANRD(I)
: The absolute tolerance in the computation of the drug amount in compartment I. The default is 12 (that is, accuracy is 10**(-12)). Used by ADVAN 9, 13, 14, and 15, which have the capability of

using different values of ANRD for different compartments. For compartments not specified, the absolute tolerance of the last compartment specified will be used.

ANRD(0) is the absolute tolerance for the Steady State computations. If ANRD(0) is not specified, ANRD(1) is used.

The value of ANRD(1) can also be specified using $SUBROUTINES option ATOL.
The value of ANRD(0) can also be specified using $SUBROUTINES option SSATOL.

NRDC(I)

Same as NRD(I), but used for the FOCE/LAPLACE covariance step. Used with ADVAN 9, 13, 14, and 15. If not set, NRDC defaults to the value of NRD. NRDC(0) is used for Steady State computations during the FOCE/LAPLACE covariance step.

The value of NRDC(1) can also be specified using $SUBROUTINES option TOLC.
The value of NRDC(0) can also be specified using $SUBROUTINES option SSTOLC.

ANRDC(I)

Same as ANRD(I), but used for the FOCE/LAPLACE covariance step. Used with ADVAN 9, 13, 14, and 15. If not set, ANRDC defaults to the value of ANRD. ANRDC(0) is used for Steady State computations during the FOCE/LAPLACE covariance step.

The value of ANRDC(1) can also be specified using $SUBROUTINES option ATOLC.
The value of ANRDC(0) can also be specified using $SUBROUTINES option SSATOLC.

A TOL routine may assign values of NRD and ANRD specifically for the initial (base) setting and each NONMEM step (estimation, covariance, simulation, table/scatter step, simulation, initial parameters estimate, nonparametric). For example, create a toluser.f90 file,

```
SUBROUTINE TOL(NRD,ANRD,NRDC,ANRDC)
USE NMPRD_INT, ONLY: IPROB
USE NM_BAYES_INT, ONLY: NM_STEP,BASE_STEP,EST_STEP,COV_STEP, &
 TABLE_STEP,SIML_STEP,INE_STEP, NONP_STEP
IMPLICIT NONE
INTEGER :: NRD(0:*), ANRD(0:*), NRDC(0:*), ANRDC(0:*)
IF(NM_STEP==EST_STEP) THEN
NRD(1)=6
ANRD(1)=10
ELSE IF (NM_STEP==COV_STEP) THEN
NRD(1)=7
ANRD(1)=8
ELSE IF (NM_STEP==TABLE_STEP) THEN
NRD(1)=8
ANRD(1)=7
ELSE
NRD(1)=9
ANRD(1)=12
ENDIF
IF(IPROB>1) THEN
NRD(1)=NRD(1)+1
ANRD(1)=ANRD(1)+1
ENDIF
RETURN
END
```

and incorporate using $SUBR, e.g.,

```
$SUBROUTINES ADVAN13 TRANS1 TOL=toluser.f90
```

E. AES

E.A AES General Description

This user-supplied subroutine is required by a general nonlinear model with equilibrium compartments (i.e. ADVAN9 and ADVAN15). AES allows the user to conveniently specify the system of algebraic equations describing the underlying kinetics of the equilibrium compartments. Its preface is

```
SUBROUTINE AES (INIT,A,P,T,E,IR,DA,DP,DT)
USE SIZES,     ONLY: DPSIZE,ISIZE
IMPLICIT REAL(KIND=DPSIZE) (A-Z)
DIMENSION :: A(*),P(*),E(*),DA(IR,*),DP(IR,*),DT(*)
```

The system of algebraic equations is actually part of a *coupled* system of differential and algebraic equations describing the underlying kinetics of all the compartments, equilibrium and nonequilibrium compartments. This coupled system can be expressed mathematically by

$$\frac{dA_i}{dt} = h_i(A, P, t) \qquad (2)$$

$$i = 1, 2, \ldots, n_1$$

$$0 = g_i(A, P, t) \qquad (3)$$

$$i = 1, 2, \ldots, n_2$$

where A is the vector of drug amounts in *all* the various compartments (other than the output compartment), P is the vector of internal PK parameters, t is time, n_1 is the number of nonequilibrium compartments (excluding the output compartment), and n_2 is the number of equilibrium compartments. The parameter P_k is defined to be the kth internal parameter. It is a continuously acting parameter. A very simple example is this.

$$n_1 = 2 \text{ and } n_2 = 1$$

$$P = (Ke, Ka, R)$$

$$h_1(A, P, t) = -Ka \cdot A_1$$

$$h_2(A, P, t) = Ka \cdot A_1 - Ke \cdot A_2$$

$$g_1(A, P, t) = A_3 - R \cdot A_2$$

i.e. the familiar one compartment model with linear elimination and absorption from a drug depot, but where a third compartment is in equilibrium with the central compartment. The parameter R is a ratio of drug amounts, not a rate parameter. In general the system need not be linear, or even homogeneous (i.e. the h and g functions can depend on t).

The arguments A, P, and T of AES are inputs to the routine and correspond to A, P, and t in (2) and (3). The GG array computed by the PK routine is passed to AES as P. As with DES, some calls to AES (including the final call during integration) may have a value of T that is larger than the end time of the integration interval.

The primary task of AES is to compute the values of the g functions and store these values in E. The value of g_I is stored in $E(n_1+I)$. (When $AES abbreviated code is used, the index i for algebraic expressions E(i) is optional; it is be supplied by NM-TRAN if omitted.) The values of the h functions are computed and stored by the DES subroutine (see section C).

In addition to computing the values of the g_i, AES must also compute the values

$$\frac{\partial g_i}{\partial A_j}(A, P, t) \quad i = 1,2,...,n_2 \text{ and all } j$$

$$\frac{\partial g_i}{\partial t}(A, P, t) \quad i= 1,2,...,n_2$$

With population data AES must also compute the values

$$\frac{\partial g_i}{\partial P_k}(A, P, t) \quad i = 1,2,...,n_2 \text{ and all } k$$

The value of $\frac{\partial g_I}{\partial A_J}$ is stored in $DA(n_1+I,J)$, the value of $\frac{\partial g_I}{\partial P_K}$ is stored in $DP(n_1+I,K)$, and the value of $\frac{\partial g_I}{\partial t}$ is stored in $DT(n_1+I)$.

When INIT=0, this signals a regular call to AES in which the values of the g functions must be stored in E and the partial derivatives just enumerated must be stored in DA, DP, and DT. Such a call occurs with many different values of time over an integration interval, although the g functions usually do not depend on time. When INIT=1, this signals an initial-condition call to AES. When $AES abbreviated code is used, the code for the call with INIT=1 is placed in the $AESINIT block of abbreviated code. Such a call occurs only with that value of time, t_0, coinciding with the beginning of an integration interval, and at such a call values that hold for A at t_0, for the equilibrium compartments, must be stored in A. (No partial derivatives need be stored.) These values are used as initial conditions for the integration. Usually, these values have already been obtained by ADVAN9, and an initial-condition call is unnecessary and does not occur. However, when, for example, t_0 is a time at which a bolus dose is given (such a dose must go into a nonequilibrium compartment), then an initial-condition call is necessary. At an initial-condition call, the values for A at t_0, for the nonequilibrium compartments, are available. Values for P are also available. (Let t_1 be the end of the integration interval. Both t_0 and t_1 are state times; see section III.B.2. The values for P are computed from the information in the argument record associated with t_1, and, if t_1 is a nonevent dose time, also from the information in the record describing the dose.) Therefore, in principle the system of algebraic equations given by (3) can be solved for the amounts in the equilibrium compartments at t_0, although, obtaining an explicit solution may not be possible. However, only an approximate solution is actually needed at INITI=1. If only approximate values for A are computed at an initial-condition call, rather than exact values, then INIT itself should be reset to 0 in AES. These approximate values serve as an initial solution, and a more precise solution is obtained numerically by ADVAN9. The number of accurate digits in the final solution with respect to *all* equilibrium compartments will be the number given in NRD(1) by subroutine TOL (see section D). If at INIT=1 a solution is returned that is at least as accurate as this, then the value of INIT should be left unchanged, i.e. INIT=1.

In the above example:

$$\frac{\partial g_1}{\partial A_1} = 0 \quad \frac{\partial g_1}{\partial A_2} = -R \quad \frac{\partial g_1}{\partial A_3} = 1$$
$$\frac{\partial g_1}{\partial Ke} = 0 \quad \frac{\partial g_1}{\partial Ka} = 0 \quad \frac{\partial g_1}{\partial R} = -A_2$$
$$\frac{\partial g_1}{\partial t} = 0$$

Also, at an initial-condition call A_3 can be given by

$$A_3 = R \cdot A_2$$

A routine for implementing the example is given in Figure 42. The numbering of the compartments and the parameters is that specified by the MODEL routine given in Figure 43, which is an extension of the MODEL routine given in Figure 38. (Note though, that the LINK array plays no role with a general non-linear model, and no values in this array are set in the routine of Figure 43.) The DES subroutine shown in Figure 40 can be used with this example. Note that it is unnecessary to set 0's in either DA, DP, or DT.

The reader should see section III.K for a discussion about PRED-error recovery from PREDPP. AES can force an immediate return to NONMEM from PREDPP with a nonzero PRED error return code and accompanying user message. The required declarations are described in section III.K.2.

E.B Global Input Variables

USE PRMOD_INT , ONLY : **ICALL**=>ICALLD

ICALL has the same value as PK and ERROR's ICALL argument. At ICALL=1, AES may optionally store values in IDEFA (see below) in order to improve run time. Values may be stored in E, etc., as at other calls, but these values are ignored. At ICALL=1 a user-written AES routine that does not check ICALL will compute E, etc., using values that PK computed (at ICALL=1) with the first event record and compartment amounts that are 0. There is some risk of arithmetic difficulties if, for example, a compartment amount of 0 is raised to a power. If this occurs, the user should insert the following statement at the top of the routine (or $AES block):

```
IF (ICALL.EQ.1) RETURN
```

ICALL blocks testing for ICALL values 4 (simulation) and 5 (data averages) are permitted.

USE PROCM_INT , ONLY : **ISFINL**

Multiple calls to AES are associated with each event record, as with DES, and global variable ISFINL is set by PREDPP to 1 on the final call to AES immediately after the advance to an event time or model-event time. When NONMEM is performing simulation or a copying pass (COMACT>0), ISFINL can be tested on a WRITE statement. As with DES, ISFINL is *not* set to 1 immediately after the advance to a model event time.

USE PROCM_REAL , ONLY : **DOSTIM , DOSREC**

DOSTIM and DOSREC are described in Chapter III, Section I.A. An AES routine may test DOSTIM and elements of DOSREC in a logical condition. It may use them on the right-hand side of an assignment statement. If DOSTIM is a random variable, DOSTIM must not be used in AES or $AES. However, DOSTIM may always be used in a $PK block or PK routine to define a random variable which may be used in the AES routine. With NM-TRAN, the $BIND record may be used, but it only affects the DOSREC for additional and lagged doses (See chapt III).

USE PROCM_REAL , ONLY : **EVTREC**

Data record items may be used explicitly in AES and $AES abbbreviated code (rather than obtained as PK parameters). EVTREC is the current event record (the event record to which the system is being advanced). This may improve run time.

USE PROCM_REAL , ONLY : **THETA**

Elements of THETA may be used explicitly in AES and $AES abbreviated code (rather than obtained as PK parameters). This may improve run time.

USE ROCM_INT , ONLY : **MIXEST , MIXNUM , MIXP**

Mixture-model variables may be used in AES. For a description, see INFN, above.

E.C Global Output Variables

USE PRMOD_INT , ONLY : **IDEFA**

At ICALL=1, AES may optionally store values in IDEFA in order to improve run time.

IDEFA(1)

IDEFA(1) may optionally be set by AES to indicate how many thetas it uses. See above remarks for IDEFD and DES.

IDEFA(2) is the calling protocol for ADVAN9.
IDEFA(2) applies only when no TIME data item is defined. It is ignored when the TIME data item is defined. AES may set this as follows:
IDEFA(2)=-1 "call with every event record" (the default)
IDEFA(2)=1 "call once per individual record"

When NM-TRAN abbreviated code is used, the reserved variable CALLFL may be used to specify the value for IDEFA(2). E.g., to set IDEFA(2)=1, use:

```
CALLFL=1
```

Alternately, a calling protocol phrase can be used instead of the CALLFL pseudo-statement, e.g.,

```
$AESINIT (ONCE PER INDIVIDUAL RECORD)
```

E.D Displaying AES-Defined Items

A value stored in a variable (or array element) V in AES may be displayed in a table or scatterplot. To accomplish this, module NMPRD4 must be defined in AES, and V must be listed in NMPRD4. The discussion in Section C.D on Displaying DES-defined variables applies also to AES-defined items. There is no warning corresponding to 48, but the warning applies to AES-defined variables as well. See also ISFINL (above).

Module NMPRD4 also provides a convenient place to store values of variables to be shared between AES and other user routines, and it is used thusly when these routines are generated from NM-TRAN abbreviated code (see Guide IV). (INFN-defined and declared variables are also shared between user routines.)

VII. The NONMEM-PREDPP Load Module

A. Introduction

To use PREDPP a number of PREDPP subroutines must be linked with the NONMEM subroutines.† This is described in detail in Guide III. Briefly, the PREDPP subroutines consist of five varieties:

(1) Routines supplied by the NONMEM Project Group (e.g. PRED), each of whose presence in a NONMEM-PREDPP load module is independent of the choice of the pharmacokinetic model.

(2) Routines supplied by the user (i.e. PK and ERROR), each of which must be present in a NONMEM-PREDPP load modudule.

(3) Routines supplied by the NONMEM Project Group (e.g. ADVAN1, TRANS1), each of whose presence in a NONMEM-PREDPP load module depends on the choice of the pharmacokinetic model and the set of basic parameters.

(4) Routines supplied by the user (e.g. DES), each of whose presence in a NONMEM-PREDPP load module depends on the choice of the pharmacokinetic model (these are described in Chapter VI).

(5) Auxilliary subroutines and files.

With NONMEM 7, subroutine MUMODEL2 is required. This subroutine should contain only the code that is needed to compute any MU parameters (MU_1, MU_2, etc.) that are used in PK or PRED. When NM-TRAN is used, subroutine MUMODEL2 is always included in FSUBS. It is an empty subroutine unless MU parameters are defined in $PK or $PRED abbreviated code.

With NONMEM 7, subroutine FSIZESR is required. NONMEM dynamically allocates the sizes of its arrays at run-time, based on the values in FSIZESR. When NM-TRAN is used, FSIZESR is included in FSUBS.

Arrays internal to PREDPP are statically allocated to improve run times. Starting with NONMEM 7.2, module PRSIZES is required to specify the array sizes. When PREDPP is installed, module PRSIZES is found in file resource/SIZES.f90. When nmfe is used, a new version of the module is generated in file prsizes.f90. Some of the constants in PRSIZES are assessed for the current problem; others are copied from default values in SIZES.f90. If nmfe performs "pr recompilation", the appropriate routines from PREDPP are recompiled (in directory temp_dir) using constants from prsizes.f90. (Some of the constants in PRSIZES are also used to compile subroutine PRED when PREDPP is not used. Subroutine PRED may be user-supplied or generated from $PRED abbreviated code.) Note that nmfe options such as -prdefault may be used to override pr recompilation.

The user may override many of the parameter values in FSIZES and prsizes with the $SIZES record.

See Introduction to NONMEM 7, Dynamic Memory Allocation, for information about dynamic and static allocation. See Guide VIII and on-line help for information about nmfe.

Subroutines of category (1) and (3) are present in the pr directory of the installed nonmem directory, although nmfe may use recompiled versions in temp_dir as described above. Subroutines of categories (2) and (4) are not present in pr. They must be in the FSUBS file. Subroutines in FSUBS include those that are generated from NM-TRAN abbreviated code (see Guide IV), and user-written subroutines specified on the $SUBROUTINES record. The latter are copied to FSUBS for compilation.

Subroutines of category (1) are: PRED, PREDI, CHECK, SADVAN, SSS, INFN. The user can substitute a user-written routine for the INFN found on the distribution medium; see section VI.A. Another subroutine of category 1 is subroutine SSS0. Its entry name is SSS, and it may be substituted

† The NONMEM load module is also referred to as the NONMEM executable. Typically, it is named

for the regular SSS routine when there are no steady-state doses in the data set (i.e. no nonzero steady-state data items appearing in dose event records). The advantage in doing this is mainly to save space in computer memory if space is a problem.

Subroutines of category (3) constitute the PREDPP Library. Their use is described in the next section.

B. The PREDPP Library

With every NONMEM-PREDPP load module two subroutines are required, ADVAN and TRANS. Choices for each are found in the PREDPP Library. Some choices for ADVAN require additional subroutines from the Library, as well as certain user-supplied subroutines described in chapter VI.

Choices for ADVAN are named ADVAN1, ADVAN2, etc. These are informal names; the formal name for each routine is ADVAN since this is the entry name used in the calling program in PREDPP. Similarly, choices for TRANS are informally named TRANS1, TRANS2, etc.

In addition to an ADVAN and TRANS subroutine, a NONMEM-PREDPP load module may require an SS subroutine. The SS routine computes steady-state kinetics. If the computation of steady-state kinetics is specified in the data set, then an SS routine is required in the load module. Otherwise, an SS routine is still required, *unless* SSS0 is used (see section VII.A). For the analytic ADVANs, there is a one-to-one correspondence between ADVAN and SS subroutines. (With early versions of PREDPP, it was possible to use SS6 with analytic ADVANS. This is no longer supported.) So, for example, if the user chooses ADVAN3, then SS3 is also chosen. The SS6 subroutine computes steady-state kinetics in a very general way and may be used with the general ADVANs: ADVAN6, ADVAN8, ADVAN10, ADVAN13, ADVAN14. With ADVAN9 and ADVAN15 only SS9 can be used. Some SS routines also require additional subroutines from the Library, as well as certain user-supplied subroutines. NM-TRAN creates a file FREPORT that describes exactly which subroutines are needed for the NONMEM load module.

C. ADVAN

In the next subsections characterisics of each of the various ADVAN routines are tabled, including definitions of the various TRANS routines. With each ADVAN, the basic and additional PK parameters that are allowed with the ADVAN are listed. Variable names used in abbreviated code are used.

To compute the kinetic equations for a kinetic model, PREDPP - but more precisely, the chosen ADVAN - uses an internal set of parameters. For a linear kinetic model these parameters are the micro-constants, i.e. rate constants, of the model. If the model is a general linear model, the MODEL subroutine specifies a numbering of these parameters (see section VI.B). The kinetic equations for a general nonlinear kinetic model PREDPP essentially consist of the differential-algebraic equations given by the user in the DES and AES routines to describe the model (see sections VI.C and VI.E). The parameters of these equations comprise the internal set of parameters.

There are only certain TRANS routines from the PREDPP Library that may be used with each choice of an ADVAN routine. (However, a user-supplied TRANS routine may be used; see section III.M.) It can be seen in the following subsections that with an ADVAN implementing a general linear or general nonlinear model, the only TRANS subroutine that may be used is TRANS1. In this case the set of basic PK parameters for TRANS1 is the same as the set of internal parameters, and TRANS1 translates each basic parameter into itself.

---

nonmem.exe (windows) or nonmem (Linux).

C.1 ADVAN1 One Compartment Linear Model

Compartments:

Compt. No.	Function	Initial Status	On/Off Allowed	Dose Allowed	Default for Dose	Default for Obs.
1	Central	On	No	Yes	Yes	Yes
2	Output	Off	Yes	No	No	No

Maximum number of basic PK parameters: 2

TRANS subroutines that may be used:

	Basic PK Parameters	
TRANS1	1. K	rate constant of elimination
	2. skip	

	Basic PK Parameters	
TRANS2	1. CL	clearance
	2. V	volume of distribution
	relationship:	K = CL/V

Additional PK Parameters:

S1 or SC	scaling parameter for central compartment
S2 or S0	scaling parameter for output compartment
F1	bioavailability for for central compartment
R1	rate for for central compartment
D1	duration for for central compartment
ALAG1	absorption lag time for for central compartment
F0 or F2 or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutines: SS1

Other required user or Library routines: none

C.2 ADVAN2 One Compartment Linear Model with First Order Absorption

Compartments:

Compt. No.	Function	Initial Status	On/Off Allowed	Dose Allowed	Default for Dose	Default for Obs.
1	Drug Depot	Off	Yes	Yes	Yes	No
2	Central	On	No	Yes	No	Yes
3	Output	Off	Yes	No	No	No

Maximum number of basic PK parameters: 3

TRANS subroutines that may be used:

	Basic PK Parameters	
TRANS1	1. K	rate constant of elimination
	2. skip	
	3. Ka	absorption rate constant

	Basic PK Parameters	
TRANS2	1. CL	clearance
	2. V	volume of distribution
	3. Ka	absorption rate constant
	relationship:	K = CL/V
		Ka = Ka

Additional PK Parameters:

S1	scaling parameter for depot compartment
S2 or SC	scaling parameter for central compartment
S3 or S0	scaling parameter for peripheral compartment
F1	bioavailability for drug depot
F2	bioavailability for peripheral compartment
R1	rate for depot compartment
R2	rate for peripheral compartment
D1	duration for depot compartment
D2	duration for peripheral compartment
ALAG1	absorption lag time for depot compartment
ALAG2	absorption lag time for central compartment
F0 or F3 or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutines: SS2

Other required user or Library routines: none

C.3 ADVAN3 Two Compartment Linear Mammillary Model

Compartments:

Compt. No.	Function	Initial Status	On/Off Allowed	Dose Allowed	Default for Dose	Default for Obs.
1	Central	On	No	Yes	Yes	Yes
2	Peripheral	On	No	Yes	No	No
3	Output	Off	Yes	No	No	No

Maximum number of basic PK parameters: 4

TRANS subroutines that may be used:

TRANS1

Basic PK Parameters
1. K rate constant of elimination
2. K12 rate constant from central to peripheral
3. K21 rate constant from peripheral to central
4. skip

TRANS3

Basic PK Parameters
1. CL clearance
2. V central volume
3. Q intercompartmental clearance
4. VSS volume of distribution at steady-state

relationship: K = CL/V
K12 = Q/V
K21 = Q/(VSS-V)

TRANS4

Basic PK Parameters
1. CL clearance
2. V1 central volume
3. Q intercompartmental clearance
4. V2 peripheral volume

relationship: K = CL/V1
K12 = Q/V1
K21 = Q/V2

TRANS5

Basic PK Parameters
1. AOB A/B
2. ALPHA α
3. BETA β
4. skip

relationship: K21=(AOB * BETA + ALPHA)/(AOB + 1)
K = (ALPHA * BETA)/K21
K12 = ALPHA + BETA - K21 - K

Basic PK Parameters

TRANS6

1. ALPHA α
2. BETA β
3. K21 rate constant peripheral to central
4. skip

relationship: K = (ALPHA * BETA)/K21
K12 = ALPHA + BETA - K21 - K

constraint: ALPHA < K21 < BETA
The roles of ALPHA and BETA are exchangeable.

Additional PK Parameters:

S1 or SC	scaling parameter for central compartment
S2	scaling parameter for peripheral compartment
S3 or S0	scaling parameter for output compartment
F1	bioavailability for central compartment
F2	bioavailability for peripheral compartment
R1	rate for central compartment
R2	rate for peripheral compartment
D1	duration for central compartment
D2	duration for peripheral compartment
ALAG1	absorption lag time for central compartment
ALAG2	absorption lag time for peripheral compartment
F0 or F3 or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS3

Other required user or Library routines: none

C.4 ADVAN4 Two Compartment Linear Mammillary Model with First Order Absorption

Compartments:

Compt. No.	Function	Initial Status	On/Off Allowed	Dose Allowed	Default for Dose	Default for Obs.
1	Drug Depot	Off	Yes	Yes	Yes	No
2	Central	On	No	Yes	No	Yes
3	Peripheral	On	No	Yes	No	No
4	Output	Off	Yes	No	No	No

Maximum number of basic PK parameters: 5

TRANS subroutines that may be used:

TRANS1

Basic PK Parameters
1. K rate constant of elimination
2. K23 rate constant from central to peripheral
3. K32 rate constant from peripheral to central
4. skip
5. Ka absorption rate constant

TRANS3

Basic PK Parameters
1. CL clearance
2. V central volume
3. Q intercompartmental clearance
4. VSS volume of distribution at steady-state
5. Ka absorption rate constant

relationship: K = CL/V
K23 = Q/V
K32 = Q/(VSS-V)
Ka = Ka

TRANS4

Basic PK Parameters
1. CL clearance
2. V2 central volume
3. Q intercompartmental clearance
4. V3 peripheral volume
5. Ka absorption rate constant

relationship: K = CL/V2
K23 = Q/V2
K32 = Q/V3

	Basic PK Parameters	
TRANS5	1. AOB	A/B
	2. ALPHA	α
	3. BETA	β
	4. skip	
	5. Ka	absorption rate constant
	relationship:	K32 = (AOB * BETA + ALPHA)/(AOB + 1) K = (ALPHA * BETA)/K32 K23 = ALPHA + BETA - K32 - K Ka = Ka

	Basic PK Parameters	
TRANS6	1. ALPHA	α
	2. BETA	β
	3. K32	rate constant peripheral to central
	4. skip	
	5. Ka	absorption rate constant
	relationship:	K = (ALPHA * BETA)/K32 K23 = ALPHA + BETA - K32 - K Ka = Ka
	constraint:	ALPHA < K32 < BETA The roles of ALPHA and BETA are exchangeable.

Additional PK Parameters:

S1	scaling parameter for depot compartment
S2 or SC	scaling parameter for central compartment
S3	scaling parameter for peripheral compartment
S4 or S0	scaling parameter for output compartment
F1	bioavailability for depot compartment
F2	bioavailability for central compartment
F3	bioavailability for peripheral compartment
R1	rate for depot compartment
R2	rate for central compartment
R3	rate for peripheral compartment
D1	duration for depot compartment
D2	duration for central compartment
D3	duration for peripheral compartment
ALAG1	absorption lag time for depot compartment
ALAG2	absorption lag time for central compartment
ALAG3	absorption lag time for peripheral compartment
F0 or F4 or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS4

Other required user or library routines: none

C.5 ADVAN5 General Linear Model

Compartment attributes set by subroutine MODEL. Suppose there are m compartments in the system, including the output compartment.

TRANS subroutines that may be used:

		Basic PK Parameters
TRANS1	Kij or KiTj	rate constant from compartment i to compartment j
	Ki0 or KiT0	rate constant from comparment i to output compartment
	Kin+1 or KiTn+1	rate constant from comparment i to output compartment

Additional PK parameters:

	For each comparment n in the system:
Sn	scaling parameter for compt. n
S0	alternate name for scaling parameter for compt. m (output compt.)
	For each dosable comparment n in the system:
Fn	bioavailability for compt. n
Rn	rate for compt. n
Dn	duration for compt. n
ALAGn	absorption lag time for compt. n
	Other additional PK parameters:
F0 or Fm or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS5

Other required user or library routines:
User: MODEL
Library: REXPON, RXSUBS

Notes:

1. ADVAN5 and SS5 may be used when the eigenvalues of the rate constant matrix are complex, e.g. with a physiolgical-based pharmacokinetic flow model. If the eigenvalues are known to be real (which is true for many pharmacokinetic systems), then it is more efficient to use ADVAN7 and SS7.
2. ADVAN5 and SS5 are based on a method involving the matrix exponential. The algorithm is described in "Contributions to the computation of the matrix exponential" by K.C. Ng. Report 212 of the Center for Pure and Applied Mathematics, University of California, Berkeley (1984). See also "A recurrence among the elements of functions of triangular matrices" by B.N. Parlett. Linear Algebra and Its Applications 14, 117-121 (1976).
3. The Laplacian method is not directly supported with ADVAN5. The $ESTIMATION record must include the NUMERICAL option with LAPLACIAN.
4. Use of the letter T as a separator between the two compartment numbers of a basic PK parameter is optional unless there are two possible interpretations of the numbers that follow K. E.g., with 12 compartments, K111 is ambiguous. It should be coded K1T11 or K11T1, depending if it symbolizes the rate constant from compartment 1 to compartment 11 or from compartment 11 to compartment 1.

C.6 ADVAN6 General Nonlinear Model

Compartment attributes set by subroutine MODEL. Suppose there are m compartments in the system, including the output compartment.

TRANS subroutines that may be used:

TRANS1		Explicit Basic PK Parameters
	P(i)	ith. basic PK parameter
		Implicit Basic PK parameters
	variable	A $PK-defined variable also used in the $DES block

Additional PK parameters:

	For each comparment n in the system:
Sn	scaling parameter for compt. n
S0	alternate name for scaling parameter for compt. m (output compt.)
	For each dosable comparment n in the system:
Fn	bioavailability for compt. n
Rn	rate for compt. n
Dn	duration for compt. n
ALAGn	absorption lag time for compt. n
	Other additional PK parameters:
F0 or Fm or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS5

Other required user or library routines:
User: MODEL, DES, TOL
Library: DVERK1, FCN1

Notes:

1. ADVAN6 uses IMSL's DVERK, a Runge-Kutta-Verner fifth and sixth order of method of integration, for nonstiff problems.
2. If SS6 is used, Library routines ZSPOW1 and FCN2 are also required.
3. See Chapter VI Section D.B for specification of relative and absolute tolerance. With ADVAN6, the specifications for compartment 1 apply to all compartments.
4. There is a limit on the number of times the differential equations are computed during any one integration over a state interval. NONMEM may terminate when this limit is reached, in which case an appropriate message appears. The limit is given by MAXFCN in SIZES. (With NONMEM 74, this is 1000000.) The $SIZES record may be used to supply a different value for MAXFCN.

C.7 ADVAN7 General Linear Model with Real Eigenvalues

Compartment attributes set by subroutine MODEL. Suppose there are m compartments in the system, including the output compartment.

TRANS subroutines that may be used:

TRANS1		Basic PK Parameters
	Kij or KiTj	rate constant from compartment i to compartment j
	Ki0 or KiT0	rate constant from comparment i to output compartment
	Kin+1 or KiTn+1	rate constant from comparment i to output compartment

Additional PK parameters:

	For each comparment n in the system:
Sn	scaling parameter for compt. n
S0	alternate name for scaling parameter for compt. m (output compt.)
	For each dosable comparment n in the system:
Fn	bioavailability for compt. n
Rn	rate for compt. n
Dn	duration for compt. n
ALAGn	absorption lag time for compt. n
	Other additional PK parameters:
F0 or Fm or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS7

Other required user or library routines:
User: MODEL
Library: RRXPON, RXSUBS

Notes:

1. ADVAN7 and SS7 may be used when the eigenvalues of the rate constant matrix are known to be real (which is true for many pharmacokinetic systems). It is more efficient to use ADVAN7 than ADVAN5 in this case.
2. ADVAN7 and SS7 are based on a method involving the matrix exponential. The algorithm is a modification of that described in "Contributions to the computation of the matrix exponential" by K.C. Ng. Report 212 of the Center for Pure and Applied Mathematics, University of California, Berkeley (1984). See also "A recurrence among the elements of functions of triangular matrices" by B.N. Parlett. Linear Algebra and Its Applications 14, 117-121 (1976).
3. The Laplacian method is not directly supported with ADVAN7. The $ESTIMATION record must include the NUMERICAL option with LAPLACIAN.
4. Use of the letter T as a separator between the two compartment numbers of a basic PK parameter is optional unless there are two possible interpretations of the numbers that follow K. E.g., with 12 compartments, K111 is ambiguous. It should be coded K1T11 or K11T1, depending if it symbolizes the rate constant from compartment 1 to compartment 11 or from compartment 11 to compartment 1.

C.8 ADVAN8 General Nonlinear Model with Stiff Differential Equations

Compartment attributes set by subroutine MODEL. Suppose there are m compartments in the system, including the output compartment.

TRANS subroutines that may be used:

TRANS1		Explicit Basic PK Parameters
	P(i)	ith. basic PK parameter
		Implicit Basic PK parameters
	variable	A $PK-defined variable also used in the $DES block

Additional PK parameters:

	For each comparment n in the system:
Sn	scaling parameter for compt. n
S0	alternate name for scaling parameter for compt. m (output compt.)
	For each dosable comparment n in the system:
Fn	bioavailability for compt. n
Rn	rate for compt. n
Dn	duration for compt. n
ALAGn	absorption lag time for compt. n
	Other additional PK parameters:
F0 or Fm or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS6

Other required user or library routines:
User: MODEL, DES, TOL
Library: DGEAR1, FCN1, FCN3
See Note 2 regarding use of SS6

Notes:

1. ADVAN8 uses IMSL's DGEAR, the Gear method of numerical integration, for stiff problems.
2. If SS6 is used, Library routines ZSPOW1 and FCN2 are also required.
3. See Chapter VI Section D.B for specification of relative and absolute tolerance. With ADVAN8, the specifications for compartment 1 apply to all compartments.
4. There is a limit on the number of times the differential equations are computed during any one integration over a state interval. NONMEM may terminate when this limit is reached, in which case an appropriate message appears. The limit is given by MAXFCN in SIZES. (With NONMEM 74, this is 1000000.) The $SIZES record may be used to supply a different value for MAXFCN.

C.9 ADVAN9 General Nonlinear Model with Equilibrium Compartments

Compartment attributes set by subroutine MODEL. Suppose there are m compartments in the system, including the output compartment.

TRANS subroutines that may be used:

TRANS1		Explicit Basic PK Parameters
	P(n)	nth. basic PK parameter
		Implicit Basic PK parameters
	variable	A $PK-defined variable also used in the $AES or $DES block

Additional PK parameters:

	For each comparment n in the system:
Sn	scaling parameter for compt. n
S0	alternate name for scaling parameter for compt. m (output compt.)
	For each dosable comparment n in the system:
Fn	bioavailability for compt. n
Rn	rate for compt. n
Dn	duration for compt. n
ALAGn	absorption lag time for compt. n
	Other additional PK parameters:
F0 or Fm or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS9

Other required user or library routines:
User: MODEL, DES, AES, TOL
Library: ADDA, FCN5, JAC, LEQT22, LSODI1, RES, ZSPOW2

Notes:

1. ADVAN9 uses the Livermore solver for ordinary differential equations, implicit form (LSODI), using the backward differentiation formulas (BDF) for stiff problems. ADVAN9 may be used with a system of differential-algebraic equations specifying the kinetics of a system with equilibrium compartments. It may be used with a system consisting of only algebraic equations. It may also be useful when the system consists of only stiff differential equations, and there are no equilibrium compartments.
2. If SS9 is used, Library routines ZSPOW1, FCN2, and FCN4 are also required.
3. See Chapter VI Section D.B for specification of relative and absolute tolerance. As of NM74, for ADVAN9 **only**, ATOL=99 specifies that variable, calculated ATOL values are to be derived, in accordance with an algorithm that is present in ADVAN9.
4. When there are no equilibrium compartments, the AES routine is not called, but a dummy routine may need to be included in the load module if the loader considers an unresolved reference to be a fatal error.
5. The Laplacian method is not directly supported with ADVAN9. The $ESTIMATION record must include the NUMERICAL option with LAPLACIAN.
6. There is a limit on the number of times the differential equations are computed during any one integration over a state interval. NONMEM may terminate when this limit is reached, in which

case an appropriate message appears. The limit is given by MAXFCN in SIZES. (With NONMEM 74, this is 1000000.) The $SIZES record may be used to supply a different value for MAXFCN.

7. There is a limit on the maximum number of integration steps. For ADVAN9, it is the largest possible integer value, Reserved variable MXSTEP may be set in $PK to a smaller value.

C.10 ADVAN10 One Compartment Model with Michaelis-Menten Elimination

Compartments:

Compt. No.	Function	Initial Status	On/Off Allowed	Dose Allowed	Default for Dose	Default for Obs.
1	Central	On	No	Yes	Yes	Yes
2	Output	Off	Yes	No	No	No

Maximum number of basic PK parameters: 3

TRANS subroutines that may be used:

	Basic PK Parameters	
TRANS1	1. VM	maximum rate
	2. KM	Michaelis constant
	3. skip	

Additional PK Parameters:

S1 or SC	scaling parameter for central compartment
S2 or S0	scaling parameter for output compartment
F1	bioavailability for central compartment
R1	rate for central compartment
D1	duration for central compartment
ALAG1	absorption lag time for central compartment
F0 or F2 or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutines: SS6

Other required user or library routines:
User: TOL
Library: MMPHI
See Note 2 regarding use of SS6

Notes:

1 The computations are based on: Beal, S.L. (1983), "Computation of the explicit solution to the Michaelis-Menten equation", Journal of Pharmacokinetics and Biopharmaceutics 11, 641-657.

C.11 ADVAN11 Three Compartment Linear Model

Compartments:

Compt. No.	Function	Initial Status	On/Off Allowed	Dose Allowed	Default for Dose	Default for Obs.
1	Central	On	No	Yes	Yes	Yes
2	Peripheral 1	On	No	Yes	No	No
3	Peripheral 2	On	No	Yes	No	No
4	Output	Off	Yes	No	No	No

Maximum number of basic PK parameters: 6

TRANS subroutines that may be used:

	Basic PK Parameters	
TRANS1	1. K	rate constant of elimination
	2. K12	rate constant from central to peripheral 1
	3. K21	rate constant from peripheral 1 to central
	4. K13	rate constant from central to peripheral 2
	5. K31	rate constant from peripheral 2 to central
	6. skip	

	Basic PK Parameters	
TRANS4	1. CL	clearance
	2. V1	central volume
	3. Q2	intercompartmental clearance (central and periph. 1)
	4. V2	peripheral 1 volume
	5. Q3	intercompartmental clearance (central and periph. 2)
	6. V3	peripheral 2 volume
	relationship:	K = CL/V1
		K12=Q2/V1
		K21=Q2/V2
		K13=Q3/V1
		K31=Q3/V3

	Basic PK Parameters	
TRANS6	1. ALPHA	α
	2. BETA	β
	3. GAMMA	γ
	4. K21	rate constant peripheral 1 to central
	5. K31	rate constant peripheral 2 to central
	6. skip	
	relationship:	K = (ALPHA * BETA * GAMMA)/(K21 * K31)
		K13 = (P + K31 * K31 - K31 * S - K * K21) / (K21 - K31)

K12 = S -K -K13 -K21 - K31

where:

S = ALPHA + BETA + GAMMA
P = ALPHA * BETA + ALPHA * GAMMA + BETA * GAMMA

constraint: ALPHA<K21<BETA<K31<GAMMA
or: ALPHA<K31<BETA<K21<GAMMA
The roles of ALPHA, BETA, GAMMA, K21, K32, K12, and K13 are symmetric and are exchangeable.

Additional PK Parameters:

S1 or SC	scaling parameter for central compartment
S2	scaling parameter for peripheral 1 compartment
S3	scaling parameter for peripheral 2 compartment
S4 or S0	scaling parameter for output compartment
F1	bioavailability for central compartment
F2	bioavailability for peripheral 1 compartment
F3	bioavailability for peripheral 2 compartment
R1	rate for central compartment
R2	rate for peripheral 1 compartment
R3	rate for peripheral 2 compartment
D1	duration for central compartment
D2	duration for peripheral 1 compartment
D3	duration for peripheral 2 compartment
ALAG1	absorption lag time for central compartment
ALAG2	absorption lag time for peripheral 1 compartment
ALAG3	absorption lag time for peripheral 2 compartment
F0 or F4 or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutines: SS11 or SS6

Other required user or Library routines: none
See Note 2 regarding use of SS6

1. The Laplacian method is not directly supported with ADVAN11. The $ESTIMATION record must include the NUMERICAL option with LAPLACIAN.

C.12 ADVAN12 Three Compartment Linear Model with First Order Absorption

Compartments:

Compt. No.	Function	Initial Status	On/Off Allowed	Dose Allowed	Default for Dose	Default for Obs.
1	Depot	Off	Yes	Yes	Yes	No
2	Central	On	No	Yes	No	Yes
3	Peripheral 1	On	No	Yes	No	No
4	Peripheral 2	On	No	Yes	No	No
5	Output	Off	Yes	No	No	No

Maximum number of basic PK parameters: 7

TRANS subroutines that may be used:

	Basic PK Parameters	
TRANS1	1. K	rate constant of elimination
	2. K23	rate constant from central to peripheral 1
	3. K32	rate constant from peripheral 1 to central
	4. K24	rate constant from central to peripheral 2
	5. K42	rate constant from peripheral 2 to central
	6. skip	
	7. Ka	absorption rate constant

	Basic PK Parameters	
TRANS4	1. CL	clearance
	2. V2	central volume
	3. Q3	intercompartmental clearance (central and periph. 1)
	4. V3	peripheral 1 volume
	5. Q4	intercompartmental clearance (central and periph. 2)
	6. V4	peripheral 2 volume
	7. Ka	absorption rate constant
	relationship:	K = CL/V2
		K23=Q3/V2
		K32=Q3/V3
		K24=Q4/V2
		K42=Q4/V4
		Ka = Ka

	Basic PK Parameters	
TRANS6	1. ALPHA	α
	2. BETA	β
	3. GAMMA	γ

4. K32	rate constant peripheral 1 to central
5. K42	rate constant peripheral 2 to central
6. skip	
7. Ka	absorption rate constant

relationship:
K = (ALPHA * BETA * GAMMA)/(K32 * K42)
K24 = (P + K42 * K42 - K42 * S - K * K32) / (K32 - K42)
K23 = S -K -K24 -K32 - K42
Ka = Ka

where:
S = ALPHA + BETA + GAMMA
P = ALPHA * BETA + ALPHA * GAMMA + BETA * GAMMA

constraint: ALPHA<K32<BETA<K42<GAMMA
or: ALPHA<K42<BETA<K32<GAMMA
The roles of ALPHA, BETA, GAMMA, K32, K42, K23, and K24 are symmetric and are exchangeable.

Additional PK Parameters:

S1	scaling parameter for depot compartment
S2 or SC	scaling parameter for central compartment
S3	scaling parameter for peripheral 1 compartment
S4	scaling parameter for peripheral 2 compartment
S5 or S0	scaling parameter for output compartment
F1	bioavailability depot compartment
F2	bioavailability central compartment
F3	bioavailability for peripheral 1 compartment
F4	bioavailability for peripheral 2 compartment
R1	rate depot compartment
R2	rate central compartment
R3	rate for peripheral 1 compartment
R4	rate for peripheral 2 compartment
D1	duration depot compartment
D2	duration central compartment
D3	duration for peripheral 1 compartment
D4	duration for peripheral 2 compartment
ALAG1	absorption lag time depot compartment
ALAG2	absorption lag time central compartment
ALAG3	absorption lag time for peripheral 1 compartment
ALAG4	absorption lag time for peripheral 2 compartment
F0 or F5 or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutines: SS12 or SS6

Other required user or Library routines: none
See Note 2 regarding use of SS6

1. The Laplacian method is not directly supported with ADVAN12. The $ESTIMATION record must include the NUMERICAL option with LAPLACIAN.

C.13 ADVAN13 General Nonlinear Model With Stiff/Nonstiff Equations (NM 71)

Compartment attributes set by subroutine MODEL. Suppose there are m compartments in the system, including the output compartment.

TRANS subroutines that may be used:

TRANS1		Explicit Basic PK Parameters
	P(n)	nth. basic PK parameter
		Implicit Basic PK parameters
	variable	A $PK-defined variable also used in the $DES block

Additional PK parameters:

	For each comparment n in the system:
Sn	scaling parameter for compt. n
S0	alternate name for scaling parameter for compt. m (output compt.)
	For each dosable comparment n in the system:
Fn	bioavailability for compt. n
Rn	rate for compt. n
Dn	duration for compt. n
ALAGn	absorption lag time for compt. n
	Other additional PK parameters:
F0 or Fm or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS6

Other required user or library routines:
User: MODEL, DES, TOL
Library: LSODA, FCN1, JAC2†

Notes:

1. ADVAN13 uses the Livermore solver for ordinary differential equations (LSODA), with automatic method switching for stiff (BDF) and non-stiff (Adams method) problems.
2. If SS6 is used, Library routines ZSPOW1 and FCN2 are also required.
3. See Chapter VI Section D.B for specification of relative and absolute tolerance.
4. There is a limit on the number of times the differential equations are computed during any one integration over a state interval. NONMEM may terminate when this limit is reached, in which case an appropriate message appears. The limit is given by MAXFCN in SIZES. (With NONMEM 74, this is 1000000.) The $SIZES record may be used to supply a different value for MAXFCN.
5. There is a limit on the maximum number of integration steps. For ADVAN13, it is 10000. Reserved variable MXSTEP may be set in $PK to a larger value.
6. With NONMEM 7.4, the Laplacian Method is supported with ADVAN13. With NONMEM 7.1 through 7.3, the Laplacian method is not directly supported with ADVAN13 and the $ESTIMATION record must include the NUMERICAL option with LAPLACIAN.

† With NONMEM 7.1 through NONMEM 7.3, the required subroutines are ZSPOW2, LEQT22, LSODA, RES, ADDA, JAC, FCN5

C.14 ADVAN14 General Nonlinear Model With Stiff/Nonstiff Equations (NM 74)

Compartment attributes set by subroutine MODEL. Suppose there are m compartments in the system, including the output compartment.

TRANS subroutines that may be used:

TRANS1		Explicit Basic PK Parameters
	P(n)	nth. basic PK parameter
		Implicit Basic PK parameters
	variable	A $PK-defined variable also used in the $DES block

Additional PK parameters:

	For each comparment n in the system:
Sn	scaling parameter for compt. n
S0	alternate name for scaling parameter for compt. m (output compt.)
	For each dosable comparment n in the system:
Fn	bioavailability for compt. n
Rn	rate for compt. n
Dn	duration for compt. n
ALAGn	absorption lag time for compt. n
	Other additional PK parameters:
F0 or Fm or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS6

Other required user or library routines:
User: MODEL, DES, TOL
Library: CVODE, FCN1, JAC2
See Note 2 regarding use of SS6

Notes:

1. ADVAN14 uses CVODES from the LLNL SUNDIALS system for ordinary differential equations and is a descendent of LSODA (ADVAN13). The user may choose to make modifications to ..\pr\CVODEU.f90.
   See ..ides
2. If SS6 is used, Library routines ZSPOW1 and FCN2 are also required.
3. See Chapter VI Section D.B for specification of relative and absolute tolerance.
4. There is a limit on the number of times the differential equations are computed during any one integration over a state interval. NONMEM may terminate when this limit is reached, in which case an appropriate message appears. The limit is given by MAXFCN in SIZES. (With NONMEM 74, this is 1000000.) The $SIZES record may be used to supply a different value for MAXFCN.
5. There is a limit on the maximum number of integration steps. For ADVAN14, it is 10000. Reserved variable MXSTEP may be set in $PK to a larger value.
6. ADVAN14 also has a root-finding algorithm. See the comments in CVODEU.f90 for its use.

C.15 ADVAN15 General Nonlinear Model with Equilibrium Compartments (NM 74)

Compartment attributes set by subroutine MODEL. Suppose there are m compartments in the system, including the output compartment.

TRANS subroutines that may be used:

TRANS1		Explicit Basic PK Parameters
	P(n)	nth. basic PK parameter
		Implicit Basic PK parameters
	variable	A $PK-defined variable also used in the $DES block

Additional PK parameters:

	For each comparment n in the system:
Sn	scaling parameter for compt. n
S0	alternate name for scaling parameter for compt. m (output compt.)
	For each dosable comparment n in the system:
Fn	bioavailability for compt. n
Rn	rate for compt. n
Dn	duration for compt. n
ALAGn	absorption lag time for compt. n
	Other additional PK parameters:
F0 or Fm or FO	output fraction
XSCALE	time scale parameter
MTIME(i)	model event time parameters

Steady-state subroutine: SS9

Other required user or library routines:
User: MODEL, DES, TOL
Library: ADDA, FCN5, JAC, LEQT22, IDA, RES, ZSPOW2

Notes:

1. ADVAN15 uses IDA from the LLNL SUNDIALS system for differential-algebraic equations and is a descendent of the LSODI1 (ADVAN9) system. The user may choose to make modifications to parameters in ..rIDAU.f90.
   See ..idesidas_guide.pdf and ..idesida_guide.pdf.
   See INTRODUCTION TO NONMEM 7, section "$SUBROUTINES: Yet Another New Differential Equation Solving Method: IDAS (ADVAN15)"
   ADVAN15 may be used with a system of differential-algebraic equations specifying the kinetics of a system with equilibrium compartments. It may also be useful when the system consists of only stiff differential equations, and there are no equilibrium compartments.
2. If SS9 is used, Library routines ZSPOW1, FCN2, and FCN4 are also required.
3. See Chapter VI Section D.B for specification of relative and absolute tolerance.
4. When there are no equilibrium compartments, the AES routine is not called, but a dummy routine may need to be included in the load module if the loader considers an unresolved reference to be a fatal error.
5. The Laplacian method is not directly supported with ADVAN15. The $ESTIMATION record must include the NUMERICAL option with LAPLACIAN.

6. There is a limit on the number of times the differential equations are computed during any one integration over a state interval. NONMEM may terminate when this limit is reached, in which case an appropriate message appears. The limit is given by MAXFCN in SIZES. (With NONMEM 74, this is 1000000.) The $SIZES record may be used to supply a different value for MAXFCN.
7. There is a limit on the maximum number of integration steps. For ADVAN15, it is 10000. Reserved variable MXSTEP may be set in $PK to a larger value.
8. ADVAN15 also has a root-finding algorithm. See the comments in IDAU.f90 for its use.

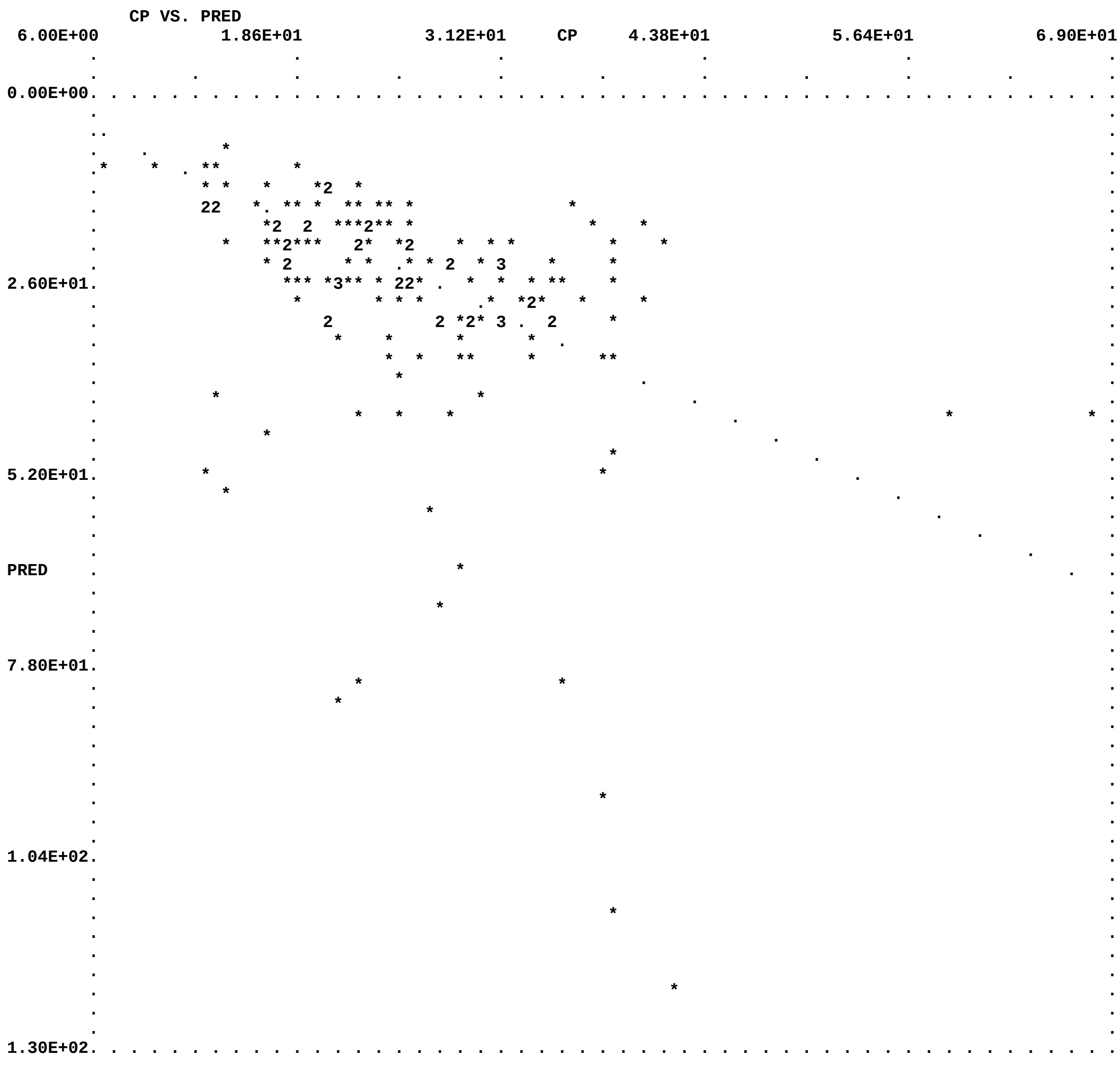
FIGURES

```
! PK SUBROUTINE FOR THE PHENOBARB POPULATION DATA
! TYPICAL VALUES RETURNED
! USED WITH ADVAN1 AND TRANS2
! CLEARANCE AND VOLUME PROPORTIONAL TO WEIGHT
! PROPORTIONALITY CONSTANT FOR VOLUME DEPENDS ON APGAR
!
      SUBROUTINE PK(ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,IRGG,GG,NETAS)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS,IRGG,NETAS
      DIMENSION :: IDEF(7,*),THETA(*),EVTREC(IREV,*),INDXS(*),GG(IRGG,*)
      IF (ICALL.GT.1) GO TO 1000
! SET UP IDEF ARRAY:
      IDEF(1,1)=-9
! ROW INDEX OF SCALING PARAMETER
      IDEF(3,1)=3
! CALL PK ONCE PER INDIV. REC.
      IDEF(1,2)=1
      RETURN
 1000 CONTINUE
! REGULAR CALLS TO PK:
! WEIGHT
      WT=EVTREC(1,4)
! APGAR
      APGR=EVTREC(1,5)
! CLEARANCE
      TVCL=THETA(1)*WT
      GG(1,1)=TVCL
      GG(1,2)=TVCL
! VOLUME
      TVVD=THETA(2)*WT
      IF (APGR.LE.2) TVVD=THETA(3)*TVVD
      GG(2,1)=TVVD
      GG(2,3)=TVVD
! SCALING
      GG(3,1)=TVVD
      GG(3,3)=TVVD
      RETURN
      END
```

```
! PK SUBROUTINE FOR THE PHENOBARB POPULATION DATA
! ILLUSTRATING DATA SIMULATION
! USED WITH ADVAN1 AND TRANS2
! CLEARANCE AND VOLUME PROPORTIONAL TO WEIGHT
! PROPORTIONALITY CONSTANT FOR VOLUME DEPENDS ON APGAR
!
      SUBROUTINE PK(ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,IRGG,GG,NETAS)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE NMPRD_REAL,ONLY: ETA
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS,IRGG,NETAS
      DIMENSION :: IDEF(7,*),THETA(*),EVTREC(IREV,*),INDXS(*),GG(IRGG,*)
      IF (ICALL.GT.1) GO TO 1000
! SET UP IDEF ARRAY:
      IDEF(1,1)=-9
! ROW INDEX OF SCALING PARAMETER
      IDEF(3,1)=3
      RETURN
 1000 CONTINUE
! REGULAR CALLS TO PK:
! WEIGHT
      WT=EVTREC(1,4)
! APGAR
      APGR=EVTREC(1,5)
      IF (ICALL.EQ.4) GO TO 2000
! DATA ANALYTIC CALL:
! CLEARANCE
      TVCL=THETA(1)*WT
      GG(1,1)=TVCL
      GG(1,2)=TVCL
! VOLUME
      TVVD=THETA(2)*WT
      IF (APGR.LE.2) TVVD=THETA(3)*TVVD
      GG(2,1)=TVVD
      GG(2,3)=TVVD
! SCALING
      GG(3,1)=TVVD
      GG(3,3)=TVVD
      RETURN
 2000 CONTINUE
! SIMULATION CALL:
      CALL SIMETA (ETA)
! CLEARANCE
      CL=THETA(1)*WT*EXP(ETA(1))
      GG(1,1)=CL
! VOLUME AND APFL
```

```
      VD=THETA(2)*WT*EXP(ETA(2))
      EVTREC(1,7)=1
      IF (APGR.LE.2) THEN
         EVTREC(1,7)=2
         VD=THETA(3)*VD
      ENDIF
      GG(2,1)=VD
! SCALING
      GG(3,1)=VD
      RETURN
      END
```

```
! PK SUBROUTINE FOR THE PHENOBARB POPULATION DATA
! SUBJECT-SPECIFIC VALUES RETURNED
! USED WITH ADVAN1 AND TRANS2
! CLEARANCE AND VOLUME PROPORTIONAL TO WEIGHT
! PROPORTIONALITY CONSTANT FOR VOLUME DEPENDS ON APGAR
!
      SUBROUTINE PK(ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,IRGG,GG,NETAS)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE NMPRD_REAL,ONLY: ETA
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS,IRGG,NETAS
      DIMENSION :: IDEF(7,*),THETA(*),EVTREC(IREV,*),INDXS(*),GG(IRGG,*)
      IF (ICALL.GT.1) GO TO 1000
! SET UP IDEF ARRAY:
      IDEF(1,1)=-9
! ROW INDEX OF SCALING PARAMETER
      IDEF(3,1)=3
! CALL PK ONCE PER INDIV. REC.
      IDEF(1,2)=1
! INITIALIZE GETETA
      CALL GETETA (ETA)
      RETURN
 1000 CONTINUE
! REGULAR CALLS TO PK:
! WEIGHT
      WT=EVTREC(1,4)
! APGAR
      APGR=EVTREC(1,5)
! GETETA
      CALL GETETA (ETA)
! CLEARANCE
      CL=THETA(1)*WT*EXP(ETA(1))
      GG(1,1)=CL
      GG(1,2)=CL
! VOLUME
      VD=THETA(2)*WT*EXP(ETA(2))
      IF (APGR.LE.2) VD=THETA(3)*VD
      GG(2,1)=VD
      GG(2,3)=VD
! SCALING
      GG(3,1)=VD
      GG(3,3)=VD
      RETURN
      END
```


CP VS. PRED
6.00E+00
1.86E+01
3.12E+01
CP
4.38E+01
5.64E+01
6.90E+01
0.00E+00
2.60E+01
5.20E+01
PRED
7.80E+01
1.04E+02
1.30E+02

```
! PK SUBROUTINE FOR THE PHENOBARB POPULATION DATA
! SUBJECT-SPECIFIC VALUES RETURNED        MIXTURE MODEL
! USED WITH ADVAN1 AND TRANS2
!
      SUBROUTINE PK(ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,IRGG,GG,NETAS)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE ROCM_INT,  ONLY: MIXNUM=>MIXCALL,MIXEST=>IMIXEST
      USE NMPRD4,ONLY: COM=>VRBL
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS,IRGG,NETAS
      DIMENSION :: IDEF(7,*),THETA(*),EVTREC(IREV,*),INDXS(*),GG(IRGG,*)
      DIMENSION ETA(4)
      DOUBLE PRECISION THETA,GG,ETA
      DOUBLE PRECISION CL,VD,EST
      IF (ICALL.GT.1) GO TO 1000
! SET UP IDEF ARRAY:
      IDEF(1,1)=-9
! ROW INDEX OF SCALING PARAMETER
      IDEF(3,1)=3
! CALL PK ONCE PER INDIV. REC.
      IDEF(1,2)=1
! INITIALIZE GETETA
      CALL GETETA (ETA)
      RETURN
 1000 CONTINUE
! REGULAR CALLS TO PK:
! GETETA
      CALL GETETA (ETA)
! MIXTURE MIXNUM ESTIMATE
      EST=MIXEST
      COM(1)=EST
! CLEARANCE
      IF (MIXNUM.EQ.1) THEN
         CL=THETA(1)*EXP(ETA(1))
         GG(1,1)=CL
         GG(1,2)=CL
      ELSE
         CL=THETA(2)*THETA(1)*EXP(ETA(3))
         GG(1,1)=CL
         GG(1,4)=CL
      ENDIF
! VOLUME
      IF (MIXNUM.EQ.1) THEN
         VD=THETA(3)*EXP(ETA(2))
         GG(2,1)=VD
         GG(2,3)=VD
```

```
      ELSE
         VD=THETA(4)*THETA(3)*EXP(ETA(4))
         GG(2,1)=VD
         GG(2,5)=VD
      ENDIF
! SCALING
      GG(3,1)=VD
      IF (MIXNUM.EQ.1) THEN
         GG(3,3)=VD
      ELSE
         GG(3,5)=VD
      ENDIF
      RETURN
      END
```

```
SUBROUTINE MIX (ICALL,NSPOP,P)
USE SIZES,     ONLY: DPSIZE,ISIZE
USE ROCM_REAL, ONLY: THETA=>THETAC,DATA=>RDATA
IMPLICIT REAL(KIND=DPSIZE) (A-Z)
INTEGER(KIND=ISIZE) :: ICALL,NSPOP
REAL(KIND=DPSIZE) :: P(*)
NSPOP=2
P(1)=THETA(5)
P(2)=1.-THETA(5)
RETURN
END
```

```
! PK SUBROUTINE FOR SINGLE-SUBJECT DATA
! USED WITH ADVAN2 AND TRANS1
!
      SUBROUTINE PK(ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,IRGG,GG,NETAS)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS,IRGG,NETAS
      DIMENSION :: IDEF(7,*),THETA(*),EVTREC(IREV,*),INDXS(*),GG(IRGG,*)
      IF (ICALL.GT.1) GO TO 1000
! SET UP IDEF ARRAY:
      IDEF(1,1)=-9
! ROW INDEX FOR SCALING PARAMETER FOR COMPT. 2
      IDEF(3,2)=4
! CALL PK ONCE PER INDIV. REC.
      IDEF(1,2)=1
      RETURN
 1000 CONTINUE
! REGULAR CALL TO PK:
! ELIMINATION RATE CONSTANT
      GG(1,1)=THETA(2)
! ABSORPTION RATE CONSTANT
      GG(3,1)=THETA(1)
! SCALING
      GG(4,1)=THETA(3)
      RETURN
      END
```

```
! PK ROUTINE FOR SINGLE-SUBJECT PHARMACODYNAMIC DATA
! USED WITH ADVAN7 AND TRANS1
!
      SUBROUTINE PK(ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,IRGG,GG,NETAS)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS,IRGG,NETAS
      DIMENSION :: IDEF(7,*),THETA(*),EVTREC(IREV,*),INDXS(*),GG(IRGG,*)
      IF (ICALL.GT.1) GO TO 1000
! SET UP IDEF ARRAY:
      IDEF(1,1)=-9
! ROW INDEX OF SCALING PARAMETER
      IDEF(3,3)=5
! CALL PK ONCE PER INDIV. REC.
      IDEF(1,2)=1
      RETURN
 1000 CONTINUE
! REGULAR CALLS TO PK:
! K12
      K12=1.94
      GG(1,1)=K12
! K20
      K20=.102
      GG(2,1)=K20
! K23
      K23=.001*K20
      GG(3,1)=K23
! K30 (KEO)
      K30=THETA(1)
      GG(4,1)=K30
! SCALING
      VD=32
      VE=VD*K23/K30
      GG(5,1)=VE
      RETURN
      END
```

```
! ERROR SUBROUTINE FOR THE PHENOBARB POPULATION DATA
! EXPONENTIAL ERROR MODEL
!
      SUBROUTINE ERROR (ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,F,G,HH)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE PRDIMS,    ONLY: GERD
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS
      DIMENSION :: IDEF(*),THETA(*),EVTREC(IREV,*),INDXS(*)
      REAL(KIND=DPSIZE) :: G(GERD,*),HH(*)
      HH(1)=F
      RETURN
      END
```

```
! ERROR ROUTINE FOR THE PHENOBARB POPULATION DATA
! ILLUSTRATING DATA SIMULATION
! EXPONENTIAL ERROR MODEL
!
      SUBROUTINE ERROR (ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,F,G,HH)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE PRDIMS,    ONLY: GERD
      USE NMPRD_REAL,ONLY: EPS
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS
      DIMENSION :: IDEF(*),THETA(*),EVTREC(IREV,*),INDXS(*)
      REAL(KIND=DPSIZE) :: G(GERD,*),HH(*)
      IF (ICALL.EQ.2) HH(1)=F
      IF (ICALL.EQ.4) THEN
         CALL SIMEPS (EPS)
         F=F*EXP(EPS(1))
      ENDIF
      RETURN
      END
```

```
! ERROR ROUTINE FOR SINGLE-SUBJECT DATA
! ADDITIVE ERROR MODEL
!
      SUBROUTINE ERROR (ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,F,G,HH)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS
      DIMENSION :: IDEF(*),THETA(*),EVTREC(IREV,*),INDXS(*)
      REAL(KIND=DPSIZE) :: HH(*)
! CALL ERROR ONLY ONCE
      IDEF(2)=2
      HH(1)=1.
      RETURN
      END
```

```
! ERROR ROUTINE FOR SINGLE-SUBJECT DATA
! ILLUSTRATING DATA SIMULATION
! ADDITIVE ERROR MODEL
!
      SUBROUTINE ERROR (ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,F,G,HH)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE NMPRD_REAL,ONLY: ETA
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS
      DIMENSION :: IDEF(*),THETA(*),EVTREC(IREV,*),INDXS(*)
      REAL(KIND=DPSIZE) :: HH(*)
      IF (ICALL.EQ.4) THEN
         CALL SIMETA (ETA)
         F=F+ETA(1)
         RETURN
      ENDIF
! CALL ERROR ONLY ONCE FOR DATA ANALYSIS
      IDEF(2)=2
      HH(1)=1.
      RETURN
      END
```

```
! ERROR ROUTINE FOR SINGLE-SUBJECT PHARMACODYNAMIC DATA
! CONSTANT CV ERROR MODEL
!
      SUBROUTINE ERROR (ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,F,G,HH)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE PRDIMS,    ONLY: GERD
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS
      DIMENSION :: IDEF(*),THETA(*),EVTREC(IREV,*),INDXS(*)
      REAL(KIND=DPSIZE) :: G(GERD,*),HH(*)
      IF (ICALL.EQ.1) RETURN
! EMAX
      EMAX=THETA(2)
! C50
      C50=THETA(3)
! EFFECT
      E=EMAX*F/(F+C50)
      F=E
! CONSTANT CV ERROR MODEL
      HH(1)=F
      RETURN
      END
```

```
! ERROR ROUTINE FOR SINGLE-SUBJECT PHARMACODYNAMIC DATA
! ILLUSTRATING ERROR-DEFINED ITEMS
! CONSTANT CV ERROR MODEL
!
      SUBROUTINE ERROR (ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,F,G,HH)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE NMPRD4,ONLY: COM=>VRBL
      USE PROCM_REAL,ONLY: A=>AMNT,DAETA,D2AETA
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS
      DIMENSION :: IDEF(*),THETA(*),EVTREC(IREV,*),INDXS(*)
      REAL(KIND=DPSIZE) :: HH(*)
      IF (ICALL.EQ.1) RETURN
! CP AND CE
      CP=A(2)/32.
      CE=F
! EMAX
      EMAX=THETA(2)
! save for display
      COM(1)=CP
      COM(2)=CE
! C50
      C50=THETA(3)
! EFFECT
      E=EMAX*F/(F+C50)
      F=E
! CONSTANT CV ERROR MODEL
      HH(1)=F
      RETURN
      END
```

1	0.	25.0	1.4	7		1	1
1	2.0		1.4	7	17.3	0	0
1	12.5	3.5	1.4	7		1	1
1	24.5	3.5	1.4	7		1	1
1	37.0	3.5	1.4	7		1	1
1	48.0	3.5	1.4	7		1	1
1	60.5	3.5	1.4	7		1	1
1	72.5	3.5	1.4	7		1	1
1	85.3	3.5	1.4	7		1	1
1	96.5	3.5	1.4	7		1	1
1	108.5	3.5	1.4	7		1	1
1	112.5		1.4	7	31.0	0	0
2	0.	15.0	1.5	9		1	1
2	2.0		1.5	9	9.7	0	0
2	4.0	3.8	1.5	9		1	1
2	16.0	3.8	1.5	9		1	1
2	27.8	3.8	1.5	9		1	1
2	40.0	3.8	1.5	9		1	1
2	52.0	3.8	1.5	9		1	1
2	63.5		1.5	9	24.6	0	0
2	64.0	3.8	1.5	9		1	1
2	76.0	3.8	1.5	9		1	1
2	88.0	3.8	1.5	9		1	1
2	100.0	3.8	1.5	9		1	1
2	112.0	3.8	1.5	9		1	1
2	124.0	3.8	1.5	9		1	1
2	135.5		1.5	9	33.0	0	0
3	0.	30.0	1.5	6		1	1
3	1.5		1.5	6	18.0	0	0
3	11.5	3.7	1.5	6		1	1
3	23.5	3.7	1.5	6		1	1
3	35.5	3.7	1.5	6		1	1
3	47.5	3.7	1.5	6		1	1
3	59.3	3.7	1.5	6		1	1
3	73.0	3.7	1.5	6		1	1
3	83.5		1.5	6	23.8	0	0
3	84.0	3.7	1.5	6		1	1
3	96.5	3.7	1.5	6		1	1
3	108.5	3.7	1.5	6		1	1
3	120.0	3.7	1.5	6		1	1
3	132.0	3.7	1.5	6		1	1
3	134.3		1.5	6	24.3	0	0

```
FILE    FSTREAM
PROB    PHENOBARB POPULATION DATA
DATA       1   0 744   8   0   0   0   0
ITEM       1   6   8  11   1   0   0   0   0   0   0   0   0   0   0
INDX       7   2   3   0   0   0   0   0   0   0   0
XVID       0   0   0   0   0
MSDT     0.00000000000000E+000
LABL                        ID,                          TIME
                           AMT,                            WT
                          APGR,                            CP
                          EVID,                           MDV
LRES SIGMA
LTHT
LETA
LEPS
FORM
(6F7.0,2F2.0)
STRC       3   2   1   0   0   1   0   1   0
THCN       1   0   0   0
THTA     4.700000000000000E-03, 9.900000000000000E-01, 1.000000000000000E+00
LOWR     0.000000000000000E+00, 0.000000000000000E+00, 0.000000000000000E+00
UPPR     1000000, 1000000, 1000000
DIAG     5.000000000000000E-02, 3.000000000000000E-02
DIAG     2.000000000000000E-02
DFLT      -1
ESTM       0 500   3   5   0   0   0   0   0   0   0   0   0   0   0   0   0   0
           0   0   0   0  -1  -1   0   0   0
ORDR
PFIL            -100
FFIL               0
COVR       0   0   0   0   0   0   1   0   0
COVT      -1  -1  -1  -1   0   0          -1     -1     -1     -1     -1     -1     -1-1-1TOS
              -1  -1  -1-0.10000E+03-0.10000E+03-0.10000E+03

CPAR              -1
TABL       1   1
PPAR              -1
TABL   1   5     1 0     2 0     3 0     4 0     5 0
           0   0   0   0   0           0           0            0-1 0
FRML
FRMR
FETA
SCAT       1   7
SCAT           9       6       0       0       0       1       0       0       0
               0       0       0      -1
SCAT           9      10       0       0       0       0       0       0       0
               0       0       0      -1
```

```
SCAT                4       10        0        0        0        0        0        0        0
                    0        0        0       -1
SCAT                5       10        0        0        0        0        0        0        0
                    0        0        0       -1
SCAT                9       11        0        0        0        0        0        0        0
                    0        0        0       -1
SCAT                4       11        0        0        0        0        0        0        0
                    0        0        0       -1
SCAT                5       11        0        0        0        0        0        0        0
                    0        0        0       -1
```

```
$PROB    PHENOBARB POPULATION DATA
$DATA  DATA1 (6F7.0)
$INPUT  ID TIME AMT WT APGR CP=DV
$SUBROUTINES ADVAN1 TRANS2
$PK
;CLEARANCE AND VOLUME PROPORTIONAL TO WEIGHT
;PROPORTIONALITY CONSTANT FOR VOLUME DEPENDS ON APGAR
      CALLFL=1
      CL=THETA(1)*WT*EXP(ETA(1))
      TVVD=THETA(2)*WT
      IF (APGR.LE.2) TVVD=THETA(3)*TVVD
      V=TVVD*EXP(ETA(2))
      S1=V
$ERROR
      Y=F*EXP(EPS(1))
$THETA   (0,.0047)  (0,.99)  (0,1.0)
$OMEGA  .05  .03
$SIGMA  .02
$ESTIM   MAXEVAL=500   PRINT=5
$COVAR
$TABLE  ID TIME AMT WT APGR
$SCAT  CP VS PRED   UNIT
$SCAT  RES VS PRED
$SCAT  RES VS WT
$SCAT  RES VS APGR
$SCAT  WRES VS PRED
$SCAT  WRES VS WT
$SCAT  WRES VS APGR
```

```
$PROB    PHENOBARB POPULATION DATA WITH SIMULATION AND TRANSGENERATION
$DATA  DATA1 (7F7.0)
; apfl and cp are set during simulation
$INPUT  ID TIME AMT WT APGR CP=DV APFL
$SUBROUTINES ADVAN1 TRANS2
$PK
;CLEARANCE AND VOLUME PROPORTIONAL TO WEIGHT
;PROPORTIONALITY CONSTANT FOR VOLUME DEPENDS ON APGAR
; By default, call with every event record.
; This allows apfl to be set for every record.
      CL=THETA(1)*WT*EXP(ETA(1))
      TVVD=THETA(2)*WT
      IF (APGR.LE.2) TVVD=THETA(3)*TVVD
      V=TVVD*EXP(ETA(2))
      S1=V
; apfl is set only with simulation
      IF (ICALL==4) THEN
       APFL=1
       IF (APGR.LE.2) APFL=2
      ENDIF
$ERROR
      Y=F*EXP(EPS(1))
$THETA   (0,.0047)  (0,.99)  (0,1.0)
$OMEGA  .05  .03
$SIGMA  .02
; required for simulation step
$SIML (1111)
$ESTIM   MAXEVAL=500   PRINT=5
$COVAR
$TABLE  ID TIME AMT WT APGR
$TABLE ID TIME APGR APFL FILE=apfla NOAPPEND NOPRINT
; use of APFL for partitioning a scatter plot
$SCAT  CP VS PRED   UNIT BY APFL
$SCAT  RES VS PRED
$SCAT  RES VS WT
$SCAT  RES VS APGR
$SCAT  WRES VS PRED
$SCAT  WRES VS WT
$SCAT  WRES VS APGR
```

```
NONLINEAR MIXED EFFECTS MODEL PROGRAM (NONMEM) VERSION 7.4.0
ORIGINALLY DEVELOPED BY STUART BEAL, LEWIS SHEINER, AND ALISON BOECKMANN
CURRENT DEVELOPERS ARE ROBERT BAUER, ICON DEVELOPMENT SOLUTIONS,
AND ALISON BOECKMANN. IMPLEMENTATION, EFFICIENCY, AND STANDARDIZATION
PERFORMED BY NOUS INFOSYSTEMS.

PROBLEM NO.:         1
PHENOBARB POPULATION DATA
DATA CHECKOUT RUN:              NO
DATA SET LOCATED ON UNIT NO.:    2
THIS UNIT TO BE REWOUND:        NO
NO. OF DATA RECS IN DATA SET:      744
NO. OF DATA ITEMS IN DATA SET:   8
ID DATA ITEM IS DATA ITEM NO.:   1
DEP VARIABLE IS DATA ITEM NO.:   6
MDV DATA ITEM IS DATA ITEM NO.:  8
INDICES PASSED TO SUBROUTINE PRED:
  7   2   3   0   0   0   0   0   0   0   0
LABELS FOR DATA ITEMS:
ID TIME AMT WT APGR CP EVID MDV
FORMAT FOR DATA:
(6F7.0,2F2.0)

TOT. NO. OF OBS RECS:      155
TOT. NO. OF INDIVIDUALS:       59
LENGTH OF THETA:   3
DEFAULT THETA BOUNDARY TEST OMITTED:    NO
OMEGA HAS SIMPLE DIAGONAL FORM WITH DIMENSION:   2
DEFAULT OMEGA BOUNDARY TEST OMITTED:    NO
SIGMA HAS SIMPLE DIAGONAL FORM WITH DIMENSION:   1
DEFAULT SIGMA BOUNDARY TEST OMITTED:    NO
INITIAL ESTIMATE OF THETA:
LOWER BOUND    INITIAL EST    UPPER BOUND
 0.0000E+00     0.4700E-02     0.1000E+07
 0.0000E+00     0.9900E+00     0.1000E+07
 0.0000E+00     0.1000E+01     0.1000E+07
INITIAL ESTIMATE OF OMEGA:
0.5000E-01
0.0000E+00   0.3000E-01
INITIAL ESTIMATE OF SIGMA:
0.2000E-01
COVARIANCE STEP OMITTED:        NO
EIGENVLS. PRINTED:              NO
SPECIAL COMPUTATION:            NO
COMPRESSED FORMAT:              NO
GRADIENT METHOD USED:     NOSLOW
SIGDIGITS ETAHAT (SIGLO):                  -1
SIGDIGITS GRADIENTS (SIGL):                -1
EXCLUDE COV FOR FOCE (NOFCOV):              NO
TURN OFF Cholesky Transposition of R Matrix (CHOLROFF): NO
KNUTHSUMOFF:                                -1
RESUME COV ANALYSIS (RESUME):               NO
SIR SAMPLE SIZE (SIRSAMPLE):              -1
NON-LINEARLY TRANSFORM THETAS DURING COV (THBND): 1
PRECONDTIONING CYCLES (PRECOND):        0
PRECONDTIONING TYPES (PRECONDS):        TOS
FORCED PRECONDTIONING CYCLES (PFCOND):0
PRECONDTIONING TYPE (PRETYPE):        0
FORCED POS. DEFINITE SETTING: (FPOSDEF):0
TABLES STEP OMITTED:    NO
NO. OF TABLES:           1
```

```
SEED NUMBER (SEED):     11456
RANMETHOD:              3U
MC SAMPLES (ESAMPLE):     300
WRES SQUARE ROOT TYPE (WRESCHOL): EIGENVALUE
-- TABLE   1 --
RECORDS ONLY:    ALL
4 COLUMNS APPENDED:    YES
PRINTED:                YES
FORMAT:                 S1PE11.4
LFORMAT:
RFORMAT:
FIXED_EFFECT_ETAS:
USER-CHOSEN ITEMS:
ID TIME AMT WT APGR
SCATTERPLOT STEP OMITTED:    NO
FAMILIES OF SCATTERPLOTS:     7
-- SCATTERPLOT   1 --
UNIT SLOPE LINE:            YES
ITEMS TO BE SCATTERED:    PRED    CP
-- SCATTERPLOT   2 --
UNIT SLOPE LINE:             NO
ITEMS TO BE SCATTERED:    PRED    RES
-- SCATTERPLOT   3 --
UNIT SLOPE LINE:             NO
ITEMS TO BE SCATTERED:    WT    RES
-- SCATTERPLOT   4 --
UNIT SLOPE LINE:             NO
ITEMS TO BE SCATTERED:    APGR    RES
-- SCATTERPLOT   5 --
UNIT SLOPE LINE:             NO
ITEMS TO BE SCATTERED:    PRED    WRES
-- SCATTERPLOT   6 --
UNIT SLOPE LINE:             NO
ITEMS TO BE SCATTERED:    WT    WRES
-- SCATTERPLOT   7 --
UNIT SLOPE LINE:             NO
ITEMS TO BE SCATTERED:    APGR    WRES
```

```
DOUBLE PRECISION PREDPP VERSION 7.4.0

ONE COMPARTMENT MODEL (ADVAN1)
MAXIMUM NO. OF BASIC PK PARAMETERS:   2
BASIC PK PARAMETERS (AFTER TRANSLATION):
  ELIMINATION RATE (K) IS BASIC PK PARAMETER NO.:  1

TRANSLATOR WILL CONVERT PARAMETERS
CLEARANCE (CL) AND VOLUME (V) TO K (TRANS2)
COMPARTMENT ATTRIBUTES
COMPT. NO.   FUNCTION   INITIAL    ON/OFF      DOSE      DEFAULT    DEFAULT
                        STATUS     ALLOWED    ALLOWED    FOR DOSE   FOR OBS.
   1         CENTRAL      ON         NO         YES        YES        YES
   2         OUTPUT       OFF        YES        NO         NO         NO

ADDITIONAL PK PARAMETERS - ASSIGNMENT OF ROWS IN GG
COMPT. NO.                             INDICES
             SCALE      BIOAVAIL.   ZERO-ORDER  ZERO-ORDER  ABSORB
                        FRACTION    RATE        DURATION    LAG
   1            3           *           *           *           *
   2            *           -           -           -           -
            - PARAMETER IS NOT ALLOWED FOR THIS MODEL
            * PARAMETER IS NOT SUPPLIED BY PK SUBROUTINE;
              WILL DEFAULT TO ONE IF APPLICABLE
ERROR IN LOG Y IS MODELED
DATA ITEM INDICES USED BY PRED ARE:
  EVENT ID DATA ITEM IS DATA ITEM NO.:      7
  TIME DATA ITEM IS DATA ITEM NO.:          2
  DOSE AMOUNT DATA ITEM IS DATA ITEM NO.:   3

PK SUBROUTINE CALLED ONCE PER INDIVIDUAL RECORD.
PK SUBROUTINE NOT CALLED AT NONEVENT (ADDITIONAL OR LAGGED) DOSE TIMES.
DURING SIMULATION, ERROR SUBROUTINE CALLED WITH EVERY EVENT RECORD.
OTHERWISE, ERROR SUBROUTINE CALLED ONCE IN THIS PROBLEM.
```

```
FILE    FSTREAM
PROB    PHENOBARB POPULATION DATA
DATA       1   0 744   8   0   0   0   0
ITEM       1   6   8  11   1   0   0   0   0   0   0   0   0   0   0
INDX       7   2   3   0   0   0   0   0   0   0   0
XVID       0   0   0   0   0
MSDT     0.00000000000000E+000
LABL                        ID,                         TIME
                           AMT,                           WT
                          APGR,                           CP
                          EVID,                          MDV
LRES SIGMA
LTHT
LETA
LEPS
FORM
(6F7.0,2F2.0)
STRC       3   2   1   0   0   1   0   1   0
THCN       1   0   0   0
THTA     4.700000000000000E-03, 9.900000000000000E-01, 1.000000000000000E+00
LOWR     0.000000000000000E+00, 0.000000000000000E+00, 0.000000000000000E+00
UPPR     1000000, 1000000, 1000000
DIAG     5.000000000000000E-02, 3.000000000000000E-02
DIAG     2.000000000000000E-02
DFLT      -1
ESTM       0 500   3   5   0   0   0   1   0   0   0   0   0   0   0   0   0   0
           0   0   0   0  -1  -1   0   0   0
COVR       0   0   0   0   0   0   1   0   0
COVT      -1  -1  -1  -1   0   0          -1     -1     -1     -1     -1     -1    -1-1-1TOS
             -1  -1  -1-0.10000E+03-0.10000E+03-0.10000E+03

CPAR               -1
TABL       1   1
PPAR               -1
TABL   1   7     1 0     2 0     3 0     4 0     5 0    12 0    13 0
           0   0   0   0   0           0           0            0-1 0
FRML
FRMR
FETA
SCAT       1   7
SCAT           9       6       0       0       0       1       0       0       0
               0       0       0      -1
SCAT           9      10       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           9      11       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           4      12       0       0       0       0       0       0       0
```

```
                 0        0        0       -1
SCAT             5       12        0        0        0        0        0        0        0
                 0        0        0       -1
SCAT             4       13        0        0        0        0        0        0        0
                 0        0        0       -1
SCAT             5       13        0        0        0        0        0        0        0
                 0        0        0       -1
```

```
$PROB    PHENOBARB POPULATION DATA
$DATA  DATA2  (6F7.0)
$INPUT  ID TIME AMT WT APGR CP=DV
$SUBROUTINES ADVAN1 TRANS2
$PK
;CLEARANCE AND VOLUME PROPORTIONAL TO WEIGHT
;PROPORTIONALITY CONSTANT FOR VOLUME DEPENDS ON APGAR
      CALLFL=1
      CL=THETA(1)*WT*EXP(ETA(1))
      TVVD=THETA(2)*WT
      IF (APGR.LE.2) TVVD=THETA(3)*TVVD
      V=TVVD*EXP(ETA(2))
      S1=V
$ERROR
      Y=F*EXP(EPS(1))
$THETA   (0,.0047)  (0,.99)  (0,1.0)
$OMEGA  .05  .03
$SIGMA  .02
$ESTIM   MAXEVAL=500   PRINT=5   POSTHOC
$COVAR
$TABLE  ID TIME AMT WT APGR ETA1 ETA2
$SCAT  CP VS PRED   UNIT
$SCAT  RES VS PRED
$SCAT  WRES VS PRED
$SCAT  ETA1 VS WT
$SCAT  ETA1 VS APGR
$SCAT  ETA2 VS WT
$SCAT  ETA2 VS APGR
```

```
FILE    FSTREAM
PROB    PHENOBARB POPULATION DATA    MIXTURE MODEL
DATA       1   0 744   8   0   0   0   0
ITEM       1   6   8  11   1   0   0   0   0   0   1   0   0   0   0
INDX       7   2   3   0   0   0   0   0   0   0   0
XVID       0   0   0   0   0
MSDT     0.00000000000000E+000
LABL                             ID,                         TIME
                                AMT,                           WT
                               APGR,                           CP
                               EVID,                          MDV
                                EST
LRES SIGMA
LTHT
LETA
LEPS
FORM
(6F7.0,2F2.0)
STRC       5   4   1   0   0   0   1   1   0
STRC       2   2
THCN       1   0   0   0
THTA     4.700000000000000E-03, 2.000000000000000E+00, 9.900000000000000E-01
         2.000000000000000E+00, 5.000000000000000E-01
LOWR     0.000000000000000E+00, 0.000000000000000E+00, 0.000000000000000E+00
         0.000000000000000E+00, 0.000000000000000E+00
UPPR     1000000              , 1000000              , 1000000
         1000000              , 1.000000000000000E+00
BLST     5.000000000000000E-02, 1.000000000000000E-02, 3.000000000000000E-02
DIAG     2.000000000000000E-02
DFLT      -1
ESTM       0 500   3   5   0   0   1   0   1   0   0   0   0   0   0   0   0   0
           0   0   0   0  -1  -1   0   0   0
SCAT       1   8
SCAT           9       6       0       0       0       1       0       0       0
               0       0       0      -1
SCAT           9      10       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           4      10       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           5      10       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           9      11       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           4      11       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           5      11       0       0       0       0       0       0       0
               0       0       0      -1
```

```
SCAT                 4       12        0        0        0        0        0        0        0
                     0        0        0       -1
```

```
$PROB    PHENOBARB POPULATION DATA    MIXTURE MODEL
$DATA  DATA2 (6F7.0)
$INPUT  ID TIME AMT WT APGR CP=DV
$SUBROUTINES ADVAN1 TRANS2
$MIX
       NSPOP=2
       P(1)=THETA(5)
       P(2)=1.-THETA(5)
$PK
       CALLFL=1
       EST=MIXEST
       CL1=THETA(1)*EXP(ETA(1))
       V1=THETA(3)*EXP(ETA(2))
       CL2=THETA(2)*THETA(1)*EXP(ETA(3))
       V2=THETA(4)*THETA(3)*EXP(ETA(4))
       IF (MIXNUM.EQ.1) THEN
         CL=CL1
         V=V1
       ELSE
         CL=CL2
         V=V2
       ENDIF
       S1=V
$ERROR
      Y=F*EXP(EPS(1))
$THETA  (0,.0047)  (0,2.0)  (0,.99)  (0,2.0)  (0,.5,1)
$OMEGA  BLOCK(2) .05  .01  .03
$OMEGA  BLOCK(2) SAME
$SIGMA  .02
$ESTM   MAXEVAL=500   PRINT=5  INTERACTION METHOD=1
$SCAT  CP VS PRED   UNIT
$SCAT  RES VS PRED
$SCAT  RES VS WT
$SCAT  RES VS APGR
$SCAT  WRES VS PRED
$SCAT  WRES VS WT
$SCAT  WRES VS APGR
$SCAT  EST VS WT
```

```
           EST VS.   WT
 9.00E-01            1.14E+00            1.38E+00    EST     1.62E+00            1.86E+00            2.10E+00
        .                   .                   .                   .                   .                   .
        .         .         .         .         .         .         .         .         .         .         .
5.00E-01. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
        .                                                                                                   .
        .       3                                                                                           .
        .       X                                                                                           .
        .                                                                                                   .
        .       C                                                                                           .
        .       $                                                                                           .
        .                                                                                                   .
        .       $                                                                                           .
        .       $                                                                                           .
1.14E+00.                                                                                                   ..
        .       $                                                                                           .
        .       $                                                                                           .
        .                                                                                                   .
        .       $                                                                                           .
        .                                                                                                   .
        .       $                                                                                           .
        .                                                                                                   .
        .                                                                                                   .
        .       Z                                                                                           .
1.78E+00.       $                                                                                           ..
        .                                                                                                   .
        .       O                                                                                           .
        .                                                                                                   .
        .                                                                                                   .
  WT    .       C                                                                                           .
        .                                                                                                   .
        .                                                                                                   .
        .                                                                                                   .
        .                                                                                                   .
2.42E+00.                                                                                                   ..
        .       B                                                                                           .
        .                                                                                                   .
        .       6                                                                                           .
        .                                                                                                   .
        .                                                                                                   .
        .                                                                                          A        .
        .                                                                                                   .
        .                                                                                                   .
        .                                                                                                   .
3.06E+00.                                                                                                   ..
        .                                                                                          F        .
        .                                                                                          K        .
        .                                                                                                   .
        .                                                                                                   .
        .                                                                                                   .
        .                                                                                                   .
        .                                                                                          C        .
        .                                                                                          B        .
        .                                                                                                   .
3.70E+00. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
        .         .         .         .         .         .         .         .         .         .         .
```

```
FILE    NULL
PROB    THEOPHYLLINE   SINGLE SUBJECT DATA
DATA       0   0  11   6   0   0   0   0
ITEM       6   3   5  11   1   0   0   0   0   0   0   0   0   0   0
INDX       4   2   1   0   0   0   0   0   0   0   0
XVID       0   0   0   0   0
MSDT     0.00000000000000E+000
LABL                          DOSE,                          TIME
                                CP,                          EVID
                               MDV,                          .ID.
LRES
LTHT
LETA
LEPS
FORM
(4F7.0,1X,F1.0,1X,F2.0)
    320    .0             1 1  1
           .27   1.71     0 0  1
           .52   7.91     0 0  2
          1.     8.31     0 0  3
          1.92   8.33     0 0  4
          3.5    6.85     0 0  5
          5.02   6.08     0 0  6
          7.03   5.4      0 0  7
          9.     4.55     0 0  8
         12.     3.01     0 0  9
         24.3     .90     0 0 10
STRC       3   1   0   0   0   1   0   0   0
THCN       1   0   0   0
THTA     1.700000000000000E+00, 1.020000000000000E-01, 2.900000000000000E+01
LOWR     0.000000000000000E+00, 0.000000000000000E+00, 0.000000000000000E+00
UPPR     1000000, 1000000, 1000000
DIAG   2
DFLT      -1
ESTM       0 240   3   2   0   0   0   0   0   0   0   0   0   0   0   0   0   0
           0   0   0   0  -1  -1   0   0   0
COVR       0   0   0   0   1   0   1   0   0
COVT      -1  -1  -1  -1   0   0          -1     -1     -1     -1     -1     -1     -1-1-1TOS
              -1  -1  -1-0.10000E+03-0.10000E+03-0.10000E+03

CPAR                 -1
TABL       1   1
PPAR                 -1
TABL   1   1     2 0
           0   0   0   0   0          0           0            0-1 0
FRML
FRMR
```

```
FETA
SCAT       1   4
SCAT           2     3     0      0     0     0     0     0     0
               0     0     0     -1
SCAT           2     7     0      0     0     0     0     0     0
               0     0     0     -1
SCAT           2     8     0      0     0     0     0     0     0
               0     0     0     -1
SCAT           3     7     0      0     0     1     0     0     0
               0     0     0     -1
```

```
$PROBLEM  THEOPHYLLINE   SINGLE SUBJECT DATA
$INPUT  DOSE=AMT TIME CP=DV
$DATA  DATA3  (3F7.0)
$SUBROUTINES  ADVAN2

$PK
CALLFL=1
KA=THETA(1)
K=THETA(2)
SC=THETA(3)

$ERROR
Y=F+ERR(1)

$THETA  (0,1.7)  (0,.102)  (0,29)

$ESTIMATION  MAXEVAL=240  PRINT=2
$COVR
$TABLE TIME
$SCAT    CP VS TIME
$SCAT    PRED VS TIME
$SCAT    RES VS TIME
$SCAT    PRED VS CP  UNIT
```

```
NONLINEAR MIXED EFFECTS MODEL PROGRAM (NONMEM) VERSION 7.4.0
ORIGINALLY DEVELOPED BY STUART BEAL, LEWIS SHEINER, AND ALISON BOECKMANN
CURRENT DEVELOPERS ARE ROBERT BAUER, ICON DEVELOPMENT SOLUTIONS,
AND ALISON BOECKMANN. IMPLEMENTATION, EFFICIENCY, AND STANDARDIZATION
PERFORMED BY NOUS INFOSYSTEMS.

PROBLEM NO.:         1
THEOPHYLLINE   SINGLE SUBJECT DATA
DATA CHECKOUT RUN:              NO
DATA SET LOCATED ON UNIT NO.:    2
THIS UNIT TO BE REWOUND:        NO
NO. OF DATA RECS IN DATA SET:       11
NO. OF DATA ITEMS IN DATA SET:   6
ID DATA ITEM IS DATA ITEM NO.:   6
DEP VARIABLE IS DATA ITEM NO.:   3
MDV DATA ITEM IS DATA ITEM NO.:  5
INDICES PASSED TO SUBROUTINE PRED:
  4   2   1   0   0   0   0   0   0   0   0
LABELS FOR DATA ITEMS:
DOSE TIME CP EVID MDV .ID.
FORMAT FOR DATA:
(3F7.0,3F2.0)

TOT. NO. OF OBS RECS:       10
TOT. NO. OF INDIVIDUALS:       10
LENGTH OF THETA:   3
DEFAULT THETA BOUNDARY TEST OMITTED:    NO
OMEGA HAS SIMPLE DIAGONAL FORM WITH DIMENSION:   1
DEFAULT OMEGA BOUNDARY TEST OMITTED:    NO
INITIAL ESTIMATE OF THETA:
LOWER BOUND    INITIAL EST    UPPER BOUND
 0.0000E+00     0.1700E+01     0.1000E+07
 0.0000E+00     0.1020E+00     0.1000E+07
 0.0000E+00     0.2900E+02     0.1000E+07
COVARIANCE STEP OMITTED:        NO
EIGENVLS. PRINTED:              NO
SPECIAL COMPUTATION:           YES
COMPRESSED FORMAT:              NO
GRADIENT METHOD USED:     NOSLOW
SIGDIGITS ETAHAT (SIGLO):                  -1
SIGDIGITS GRADIENTS (SIGL):                -1
EXCLUDE COV FOR FOCE (NOFCOV):              NO
TURN OFF Cholesky Transposition of R Matrix (CHOLROFF): NO
KNUTHSUMOFF:                                -1
RESUME COV ANALYSIS (RESUME):               NO
SIR SAMPLE SIZE (SIRSAMPLE):              -1
NON-LINEARLY TRANSFORM THETAS DURING COV (THBND): 1
PRECONDTIONING CYCLES (PRECOND):        0
PRECONDTIONING TYPES (PRECONDS):        TOS
FORCED PRECONDTIONING CYCLES (PFCOND):0
PRECONDTIONING TYPE (PRETYPE):        0
FORCED POS. DEFINITE SETTING: (FPOSDEF):0
TABLES STEP OMITTED:    NO
NO. OF TABLES:           1
SEED NUMBER (SEED):    11456
RANMETHOD:             3U
MC SAMPLES (ESAMPLE):    300
WRES SQUARE ROOT TYPE (WRESCHOL): EIGENVALUE
-- TABLE   1 --
RECORDS ONLY:    ALL
4 COLUMNS APPENDED:    YES
```

```
PRINTED:                YES
FORMAT:                 S1PE11.4
LFORMAT:
RFORMAT:
FIXED_EFFECT_ETAS:
USER-CHOSEN ITEMS:
TIME
SCATTERPLOT STEP OMITTED:    NO
FAMILIES OF SCATTERPLOTS:     4
-- SCATTERPLOT   1 --
UNIT SLOPE LINE:             NO
ITEMS TO BE SCATTERED:    TIME    CP
-- SCATTERPLOT   2 --
UNIT SLOPE LINE:             NO
ITEMS TO BE SCATTERED:    TIME    PRED
-- SCATTERPLOT   3 --
UNIT SLOPE LINE:             NO
ITEMS TO BE SCATTERED:    TIME    RES
-- SCATTERPLOT   4 --
UNIT SLOPE LINE:            YES
ITEMS TO BE SCATTERED:    CP    PRED
```

```
DOUBLE PRECISION PREDPP VERSION 7.4.0

ONE COMPARTMENT MODEL WITH FIRST-ORDER ABSORPTION (ADVAN2)
MAXIMUM NO. OF BASIC PK PARAMETERS:   3
BASIC PK PARAMETERS (AFTER TRANSLATION):
  ELIMINATION RATE (K) IS BASIC PK PARAMETER NO.:  1
  ABSORPTION RATE (KA) IS BASIC PK PARAMETER NO.:  3

COMPARTMENT ATTRIBUTES
COMPT. NO.   FUNCTION   INITIAL    ON/OFF      DOSE      DEFAULT    DEFAULT
                        STATUS     ALLOWED    ALLOWED    FOR DOSE   FOR OBS.
   1         DEPOT        OFF        YES        YES        YES        NO
   2         CENTRAL      ON         NO         YES        NO         YES
   3         OUTPUT       OFF        YES        NO         NO         NO

ADDITIONAL PK PARAMETERS - ASSIGNMENT OF ROWS IN GG
COMPT. NO.                             INDICES
             SCALE      BIOAVAIL.   ZERO-ORDER  ZERO-ORDER  ABSORB
                        FRACTION    RATE        DURATION    LAG
   1            *           *           *           *           *
   2            4           *           *           *           *
   3            *           -           -           -           -
            - PARAMETER IS NOT ALLOWED FOR THIS MODEL
            * PARAMETER IS NOT SUPPLIED BY PK SUBROUTINE;
              WILL DEFAULT TO ONE IF APPLICABLE
DATA ITEM INDICES USED BY PRED ARE:
  EVENT ID DATA ITEM IS DATA ITEM NO.:      4
  TIME DATA ITEM IS DATA ITEM NO.:          2
  DOSE AMOUNT DATA ITEM IS DATA ITEM NO.:   1

PK SUBROUTINE CALLED ONCE PER INDIVIDUAL RECORD.
PK SUBROUTINE NOT CALLED AT NONEVENT (ADDITIONAL OR LAGGED) DOSE TIMES.
DURING SIMULATION, ERROR SUBROUTINE CALLED WITH EVERY EVENT RECORD.
OTHERWISE, ERROR SUBROUTINE CALLED ONCE IN THIS PROBLEM.
```

```
FILE    NULL
PROB    THEOPHYLLINE   SINGLE SUBJECT DATA  PHARMACODYNAMICS
DATA       0   0  11   6   0   0   0   0
ITEM       6   3   5  11   1   0   0   0   0   0   0   0   0   0   0
INDX       4   2   1   0   0   0   0   0   0   0   0
XVID       0   0   0   0   0
MSDT     0.00000000000000E+000
LABL                          DOSE,                          TIME
                               EFF,                          EVID
                               MDV,                          .ID.
FORM
(4F7.0,1X,F1.0,1X,F2.0)
    320    .0              1 1  1
           .27   .094      0 0  1
           .52   .163      0 0  2
          1.     .317      0 0  3
          1.92   .544      0 0  4
          3.5    .689      0 0  5
          5.02   .473      0 0  6
          7.03   .733      0 0  7
          9.     .667      0 0  8
         12.     .327      0 0  9
         24.3    .151      0 0 10
STRC       3   1   0   0   0   1   0   0   0
THCN       1   0   0   0
THTA     1.000000000000000E+00, 1.000000000000000E+00, 5.000000000000000E+00
LOWR     0.000000000000000E+00, 0.000000000000000E+00, 0.000000000000000E+00
UPPR     1000000, 1000000, 1000000
DIAG   2
DFLT      -1
ESTM       0 240   3   2   0   0   0   0   0   0   0   0   0   0   0   0   0   0
           0   0   0   0  -1  -1   0   0   0
COVR       0   0   0   0   1   0   1   0   0
COVT      -1  -1  -1  -1   0   0          -1    -1    -1    -1    -1    -1    -1-1-1TOS
              -1  -1  -1-0.10000E+03-0.10000E+03-0.10000E+03

CPAR              -1
TABL       1   1
PPAR              -1
TABL   1   1     2 0
           0   0   0   0   0           0           0             0-1 0
FRML
FRMR
FETA
SCAT       1   4
SCAT           2       3       0       0       0       0       0       0       0
               0       0       0      -1
```

```
SCAT                2       7       0        0       0       0       0       0       0
                    0       0       0       -1
SCAT                2       8       0        0       0       0       0       0       0
                    0       0       0       -1
SCAT                3       7       0        0       0       1       0       0       0
                    0       0       0       -1
```

```
$PROBLEM  THEOPHYLLINE   SINGLE SUBJECT DATA  PHARMACODYNAMICS
$INPUT  DOSE=AMT TIME EFF=DV
$DATA  DATA4  (3F7.0)
$SUBROUTINES  ADVAN7
$MODEL COMP=(DEPOT,DEFDOSE)  COMP=(CENTRAL)  COMP=(EFFECT,DEFOBS)

$PK
CALLFL=1
K12=1.94
K20=.102
K23=.001*K20
K30=THETA(1)
VD=32
S3=VD*K23/K30

$ERROR
EMAX=THETA(2)
C50=THETA(3)
E=EMAX*F/(F+C50)
Y=E*(1+ETA(1))

$THETA  (0,1)  (0,1)  (0,5)

$ESTIMATION  MAXEVALS=240   PRINT=2
$COVR
$TABLE TIME
$SCAT    EFF VS TIME
$SCAT    PRED VS TIME
$SCAT    RES VS TIME
$SCAT    PRED VS EFF  UNIT
```

```
FILE    FSTREAM
PROB    THEOPHYLLINE   SINGLE SUBJECT DATA  PHARMACODYNAMICS
DATA       1   0  11   6   0   0   0   0
ITEM       6   3   5  11   1   0   0   0   0   0   8   0   0   0   0
INDX       4   2   1   0   0   0   0   0   0   0   0
XVID       0   0   0   0   0
MSDT     0.00000000000000E+000
LABL                         DOSE,                          TIME
                              EFF,                          EVID
                              MDV,                          .ID.
     ,
     ,
     ,
                               CP,                            CE
LRES
LTHT
LETA
LEPS
FORM
(3F7.0,3F2.0)
STRC       3   1   0   0   0   1   0   0   0
THCN       1   0   0   0
THTA     1.000000000000000E+00, 1.000000000000000E+00, 5.000000000000000E+00
LOWR     0.000000000000000E+00, 0.000000000000000E+00, 0.000000000000000E+00
UPPR     1000000, 1000000, 1000000
DIAG   2
DFLT      -1
ESTM       0 240   3   2   0   0   0   0   0   0   0   0   0   0   0   0   0   0
           0   0   0   0  -1  -1   0   0   0
COVR       0   0   0   0   1   0   1   0   0
COVT      -1  -1  -1  -1   0   0          -1    -1    -1    -1    -1    -1    -1-1-1TOS
              -1  -1  -1-0.10000E+03-0.10000E+03-0.10000E+03

CPAR               -1
TABL       1   1
PPAR               -1
TABL   1   3     2 0    16 0    17 0
           0   0   0   0   0          0          0            0-1 0
FRML
FRMR
FETA
SCAT       1   8
SCAT           2       3       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           2       7       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           2       8       0       0       0       0       0       0       0
```

```
               0       0       0      -1
SCAT           3       7       0       0       0       1       0       0       0
               0       0       0      -1
SCAT          17       7       0       0       0       0       0       0       0
               0       0       0      -1
SCAT          16      17       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           2      16       0       0       0       0       0       0       0
               0       0       0      -1
SCAT           2      17       0       0       0       0       0       0       0
               0       0       0      -1
```

```
$PROBLEM  THEOPHYLLINE   SINGLE SUBJECT DATA  PHARMACODYNAMICS
$INPUT  DOSE=AMT TIME EFF=DV
$DATA  DATA4  (3F7.0)
$SUBROUTINES  ADVAN7
$MODEL COMP=(DEPOT,DEFDOSE)  COMP=(CENTRAL)  COMP=(EFFECT,DEFOBS)

$PK
CALLFL=1
K12=1.94
K20=.102
K23=.001*K20
K30=THETA(1)
VD=32
S3=VD*K23/K30

$ERROR
CP=A(2)/VD
CE=F
EMAX=THETA(2)
C50=THETA(3)
E=EMAX*F/(F+C50)
Y=E*(1+ETA(1))

$THETA  (0,1)  (0,1)  (0,5)

$ESTIMATION  MAXEVALS=240   PRINT=2
$COVR
$TABLE TIME CP CE
$SCAT    EFF VS TIME
$SCAT    PRED VS TIME
$SCAT    RES VS TIME
$SCAT    PRED VS EFF  UNIT
$SCAT    PRED VS CE
$SCAT    CE VS CP
$SCAT    CP VS TIME
$SCAT    CE VS TIME
```

```
           PRED VS.   CE
 -1.00E-02            1.22E-01            2.54E-01   PRED     3.86E-01            5.18E-01            6.50E-01
         .                   .                   .                   .                   .                   .
         .         .         .         .         .         .         .         .         .         .         .
-2.00E-01. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
         . *                                                                                                 .
         .                                                                                                   .
         .                                                                                                   .
         .            *                                                                                      .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                         *                                                                         .
         .                                                                                                   .
 1.32E+00.                              *                                                                    ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 2.84E+00.                                                                                                   ..
         .                                                                                                   .
         .                                                            *                                      .
         .                                                                                                   .
         .                                                                                                   .
   CE    .                                                                 *                                 .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 4.36E+00.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                              *                    .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 5.88E+00.                                                                                       2           ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                               *   .
         .                                                                                                   .
         .                                                                                                 * .
 7.40E+00. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
         .         .         .         .         .         .         .         .         .         .         .
```

```
             CE VS.   CP
 -1.00E-01            1.40E+00            2.90E+00     CE     4.40E+00            5.90E+00            7.40E+00
         .                   .                   .                   .                   .                   .
         .         .         .         .         .         .         .         .         .         .         .
-2.00E-01. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
         .*                                                                                                  .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .              *                                                                                    .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 1.56E+00.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                *                                                  .
 3.32E+00.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .      *                                                                                            .
   CP    .                                                                 *                                 .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 5.08E+00.                                                                              *                    ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                 *                                                                                 .
         .                                                                                            *      .
         .                                                                                                   .
         .                                                                                                   .
 6.84E+00.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                 * .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                           *                                                       .
         .                                                                                                   .
         .                                                                              *                    .
 8.60E+00. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
         .         .         .         .         .         .         .         .         .         .         .
```

```
             CP VS. TIME
 -1.00E-01            1.64E+00            3.38E+00     CP     5.12E+00            6.86E+00            8.60E+00
         .                   .                   .                   .                   .                   .
         .         .         .         .         .         .         .         .         .         .         .
-1.00E+00. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
         .                                                                                                   .
         .*                                             *                                                    .
         .                                                                       *                           .
         .                                                                                            *      .
         .                                                                                                   .
         .                                                                                                 * .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                    *              .
 4.20E+00.                                                                                                   ..
         .                                                                                                   .
         .                                                                         *                         .
         .                                                                                                   .
         .                                                                                                   .
         .                                                           *                                       .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                 *                                                 .
 9.40E+00.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 TIME    .                                    *                                                              .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 1.46E+01.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 1.98E+01.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .          *                                                                                        .
 2.50E+01. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
         .         .         .         .         .         .         .         .         .         .         .
```

```
             CE VS. TIME
 -1.00E-01            1.40E+00            2.90E+00     CE     4.40E+00            5.90E+00            7.40E+00
         .                   .                   .                   .                   .                   .
         .         .         .         .         .         .         .         .         .         .         .
-1.00E+00. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
         .                                                                                                   .
         .*     *                                                                                            .
         .                 *                                                                                 .
         .                                           *                                                       .
         .                                                                                                   .
         .                                                                              *                    .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                 * .
 4.20E+00.                                                                                                   ..
         .                                                                                                   .
         .                                                                                           *       .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                              *                    .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                 *                                 .
 9.40E+00.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 TIME    .                                                *                                                  .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 1.46E+01.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
 1.98E+01.                                                                                                   ..
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .                                                                                                   .
         .              *                                                                                    .
 2.50E+01. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
         .         .         .         .         .         .         .         .         .         .         .
```

```
! THIS EXAMPLE IS IDENTICAL TO HELP FILE infn1.exa
! INFN ROUTINE FOR COMPUTING LINEARLY INTERPOLATED VALUES
! OF AN INDEPENDENT VARIABLE V.    ILLUSTRATES USE OF ROUTINE PASS.
! USE FOR SINGLE-SUBJECT OR POPULATION DATA
! WORKS WITH 1 OR MORE NON-MISSING VALUES.
! A SINGLE SUCH VALUE MAY BE ANYWHERE IN DATA SET.
! DATREC(UI)=TIME DATA ITEM
! DATREC(VI)=INDEPENDENT VARIABLE DATA ITEM
! DATREC(MI)=MISSING INDEPENDENT VARIABLE DATA ITEM
!   =0 INDEP VAR NOT MISSING
!   >0 IF THIS DATA RECORD HAS MISSING INDEP VAR AND:
!   =1    THIS DATA RECORD PRECEDES FIRST NON-MISSING VALUE
!   =3    THIS DATA RECORD FOLLOWS LAST NON-MISSING VALUE
!   =2    THIS DATA RECORD IS BETWEEN NON-MISSING VALUES
!
       SUBROUTINE INFN (ICALL,THETA,DATREC,INDXS,NEWIN)
       USE SIZES, ONLY: ISIZE,DPSIZE,NO,MAXIDS
! NOTE THAT NEWIND IS NOT OBTAINED FROM THE SUBROUTINE ARGUMENT
       USE NMPRD_INT, ONLY: NEWIND=>NWIND
       INTEGER(KIND=ISIZE), INTENT(IN) :: ICALL,INDXS(*),NEWIN
       REAL(KIND=DPSIZE),   INTENT(IN) :: THETA(*)
       REAL(KIND=DPSIZE),   INTENT(IN OUT) :: DATREC(*)
       REAL(KIND=DPSIZE) :: U,V
       DIMENSION U(NO,MAXIDS),V(NO,MAXIDS),DEPVAR(MAXIDS),LASTI(MAXIDS)
       INTEGER UI,VI,MI,IS,LASTI
! IMPORTANT: CHANGE UI, VI, MI ACCORDING TO LAYOUT OF DATA SET
       UI=3
       VI=5
       MI=6
!
       IF (ICALL.NE.1) RETURN
       I=0
       IS=0
 ! INITIALIZE PASS
       MODE=0
       CALL PASS (MODE)
       MODE=2
 ! PASS THROUGH DATA
     5 CALL PASS (MODE)
       IF (MODE.EQ.0) GO TO 10
       IF (NEWIND<2) THEN
         I=0
         IS=IS+1
       ENDIF
 ! IF INDEP VAR IS PRESENT, STORE TIME AND VALUE
       IF (DATREC(MI).EQ.0.) THEN
          I=I+1
```

```
          U(IS,I)=DATREC(UI)
          V(IS,I)=DATREC(VI)
          LASTI(IS)=I
       ENDIF
       IF (I == 1) THEN   ! SAVE VALUE IN CASE ONLY RECORDED ONCE
          DEPVAR(IS)=V(IS,1)
       ENDIF
       GO TO 5
 ! INITIALIZE PASS A SECOND TIME
    10 I=0
       IS=0
       MODE=0
       CALL PASS (MODE)
       MODE=2
 ! PASS THROUGH DATA A SECOND TIME
    15 CALL PASS (MODE)
       IF (MODE.EQ.0) RETURN
       IF (NEWIND<2) THEN
         I=0
         IS=IS+1
       ENDIF
! IF INDEP VAR IS MISSING AND ONLY RECORDED ONCE, COPY IT
       IF (DATREC(MI).NE.0. .AND. LASTI(IS) == 1 ) THEN
        DATREC(VI)=DEPVAR(IS)
        GO TO 15
       ENDIF
 ! IF INDEP VAR IS MISSING, STORE INTERPOLATED VALUE
       IF (DATREC(MI).EQ.0.) THEN
         I=I+1
       ELSE
       IF (DATREC(MI).EQ.1.) THEN   ! EXTRAPOL. FROM FIRST 2 VALUES
         K=1
         L=2
       ELSEIF (DATREC(MI).EQ.2.) THEN  ! INTERPOL. FROM BEFORE AND AFTER
         K=I
         L=I+1
       ELSEIF (DATREC(MI).EQ.3.) THEN   ! EXTRAPOL. FROM LAST 2 VALUES
       K=I-1
       L=I
       ENDIF
       B=(V(IS,K)-V(IS,L))/(U(IS,K)-U(IS,L))
       DATREC(VI)=V(IS,K)+B*(DATREC(UI)-U(IS,K))
       ENDIF
       GO TO 15
       END
```

```
! DEFINES A 1 COMPARTMENT LINEAR MODEL WITH FIRST-ORDER ABSORPTION
! COMPT1: DRUG DEPOT   COMPT2: CENTRAL COMPT
!
      SUBROUTINE MODEL (IDNO,NCM,NPAR,IR,IATT,LINK)
      USE PRMOD_CHAR, ONLY: NAME
      USE SIZES,     ONLY: DPSIZE,ISIZE,SD
      USE PRDIMS,    ONLY: GPRD,HPRD,GERD,HERD,GPKD
      INTEGER(KIND=ISIZE) :: IDNO,NCM,NPAR,IR,IATT,LINK,I,J
      DIMENSION :: IATT(IR,*),LINK(IR,*)
      SAVE
      INTEGER(KIND=ISIZE), DIMENSION (2,7) :: MOD
      CHARACTER(LEN=SD), DIMENSION(2) :: CMOD
      DATA (MOD(I,  1),I=  1,  2)/&
! INITIAL STATUS: OFF ON
      0,1 /
      DATA (MOD(I,  2),I=  1,  2)/&
! ON/OFF ALLOWED: YES NO
      1,0 /
      DATA (MOD(I,  3),I=  1,  2)/&
! DOSE ALLOWED: YES YES
      1,1 /
      DATA (MOD(I,  4),I=  1,  2)/&
! DEFAULT FOR OBSERVATIONS: NO YES
      0,1 /
      DATA (MOD(I,  5),I=  1,  2)/&
! DEFAULT FOR DOSES: YES NO
      1,0 /
      DATA (MOD(I,  6),I=  1,  2)/&
      0,0 /
      DATA (MOD(I,  7),I=  1,  2)/&
      0,0 /
      DATA (CMOD(I),I=  1,  2) &
      /'DEPOT','CENTRAL'/
      FORALL (I=1:2) NAME(I)=CMOD(I)
      FORALL (I=1:2,J=1:7) IATT(I,J)=MOD(I,J)
      IDNO=9999
      NCM=  2
      NPAR=002
      RETURN
      END
```

```
! DEFINES A 3 COMPARTMENT MODEL
! COMPT1: DRUG DEPOT   COMPT2: CENTRAL COMPT   COMPT3: EFFECT COMPT
!
      SUBROUTINE MODEL (IDNO,NCM,NPAR,IR,IATT,LINK)
      USE PRMOD_CHAR, ONLY: NAME
      USE SIZES,     ONLY: DPSIZE,ISIZE,SD
      USE PRDIMS,    ONLY: GPRD,HPRD,GERD,HERD,GPKD
      INTEGER(KIND=ISIZE) :: IDNO,NCM,NPAR,IR,IATT,LINK,I,J
      DIMENSION :: IATT(IR,*),LINK(IR,*)
      SAVE
      INTEGER(KIND=ISIZE), DIMENSION (3,7) :: MOD
      CHARACTER(LEN=SD), DIMENSION(3) :: CMOD
      DATA (MOD(I,  1),I=  1,  3)/&
! INITIAL STATUS: ON ON ON
      1,1,1 /
      DATA (MOD(I,  2),I=  1,  3)/&
! ON/OFF ALLOWED: NO NO NO
      0,0,0 /
      DATA (MOD(I,  3),I=  1,  3)/&
! DOSE ALLOWED: YES NO NO
      1,0,0 /
      DATA (MOD(I,  4),I=  1,  3)/&
! DEFAULT FOR OBSERVATIONS: NO NO YES
      0,0,1 /
      DATA (MOD(I,  5),I=  1,  3)/&
! DEFAULT FOR DOSES: YES NO NO
      1,0,0 /
      DATA (MOD(I,  6),I=  1,  3)/&
      0,0,0 /
      DATA (MOD(I,  7),I=  1,  3)/&
      0,0,0 /
      DATA (CMOD(I),I=  1,  3) &
      /'DEPOT','CENTRAL','EFFECT'/
      FORALL (I=1:3) NAME(I)=CMOD(I)
      FORALL (I=1:3,J=1:7) IATT(I,J)=MOD(I,J)
      IDNO=9999
      NCM=  3
      NPAR=004
! SET LINK:
! K12
    LINK(  1,002)=001
! K23
    LINK(  2,003)=003
! K20
    LINK(  2,004)=002
! K30 (KEO)
      LINK(  3,004)=004
```

```
      RETURN
      END
```

```
      SUBROUTINE DES (A,P,T,DADT,IR,DA,DP,DT)
      USE NMPRD4P
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE PRDIMS,    ONLY: GPRD,HPRD,GERD,HERD,GPKD
      USE NMPRD_INT, ONLY: MSEC=>ISECDER,MFIRST=>IFRSTDER,IFIRSTEM,IFIRSTEMJAC
      USE PRCOM_INT, ONLY: MITER
      USE NMPRD_INT, ONLY: MDVRES,ETASXI,NPDE_MODE,NOFIRSTDERCODE
      USE NMPRD_REAL, ONLY: DV_LOQ,CDF_L
      USE PRMOD_INT, ONLY: ICALL=>ICALLD,IDEFD,IDEFA
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      SAVE
      INTEGER(KIND=ISIZE) :: IR
      DIMENSION :: A(*),P(*),DADT(*),DA(IR,*),DP(IR,*),DT(*)
      INTEGER(KIND=ISIZE) :: FIRSTEM,IFIRSTEMJACIN
      IF(MITER==1.OR.MITER==4) IFIRSTEM=1
      FIRSTEM=IFIRSTEM
      IFIRSTEMJACIN=IFIRSTEMJAC
      IF(NOFIRSTDERCODE/=1) THEN
      IFIRSTEMJAC=FIRSTEM
      ELSE
      IFIRSTEMJAC=0
      ENDIF
      IF(IFIRSTEMJACIN==-2) RETURN
      IF (ICALL == 1) THEN
      CALL ASSOCNMPRD4
      IDEFD(1)=  0
      RETURN
      ENDIF
 !  level            0
 !  level            0
      DADT(1)=-P(002)*A(1)
      DADT(2)=P(002)*A(1)-P(001)*A(2)
      IF (FIRSTEM == 1) THEN ! 1
!                      E000004 = DERIVATIVE OF DADT(1) W.R.T. A(001)
      E000004=-P(002)
!                      F000081 = DERIVATIVE OF DADT(1) W.R.T. P(002)
      F000081=-A(1)
!                      E000005 = DERIVATIVE OF DADT(2) W.R.T. A(001)
      E000005=P(002)
!                      E000006 = DERIVATIVE OF DADT(2) W.R.T. A(002)
      E000006=-P(001)
!                      F000082 = DERIVATIVE OF DADT(2) W.R.T. P(002)
      F000082=A(1)
!                      F000083 = DERIVATIVE OF DADT(2) W.R.T. P(001)
      F000083=-A(2)
      ENDIF !1
      IF (FIRSTEM == 1) THEN !3
```

```
DA(  1,  1)=E000004
DA(  2,  1)=E000005
DA(  2,  2)=E000006
DP(  1,002)=F000081
DP(  2,001)=F000083
DP(  2,002)=F000082
ENDIF !3
RETURN
END
```

```
! TOL SUBROUTINE for NONMEM 7.3 and earler
      SUBROUTINE TOL (NRD)
      DIMENSION NRD(*)
      NRD(1)=4
      RETURN
      END
```

```
! TOL SUBROUTINE for NONMEM 7.4 and later
      SUBROUTINE TOL(NRD,ANRD,NRDC,ANRDC)
      USE SIZES,     ONLY: ISIZE
      INTEGER(KIND=ISIZE) :: NRD(0:*), ANRD(0:*), NRDC(0:*), ANRDC(0:*)
      NRD(1)=5
      RETURN
      END
```

```
! AES FOR COMPARTMENT (3) IN EQUILLIBRIUM WITH COMPARTMENT (2)
!
      SUBROUTINE AES (INIT,A,P,T,E,IR,DA,DP,DT)
      USE SIZES,     ONLY: DPSIZE,ISIZE
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      INTEGER(KIND=ISIZE) :: IR,INIT
      DIMENSION :: A(*),P(*),E(*),DA(IR,*),DP(IR,*),DT(*)
      IF (INIT.EQ.1) THEN
! SOLUTION FOR EQUILLIBRIUM COMPARTMENT
         A(3)=P(4)*A(2)
      ELSE
! EQUATIONS FOR EQUILLIBRIUM COMPARTMEMT
         E(3)=A(3)-P(4)*A(2)
         DA(3,2)=-P(4)
         DA(3,3)=1.
         DP(3,3)=-A(2)
      ENDIF
      RETURN
      END
```

```
SUBROUTINE MODEL (IDNO,NCM,NPAR,IR,IATT,LINK)
USE PRMOD_CHAR, ONLY: NAME
USE SIZES,     ONLY: ISIZE
INTEGER(KIND=ISIZE) :: IDNO,NCM,NPAR,IR,IATT,LINK
DIMENSION :: IATT(IR,*),LINK(IR,*)
INTEGER(KIND=ISIZE) :: I,J
INTEGER(KIND=ISIZE), DIMENSION (3,9) :: MOD
CHARACTER(LEN=8), DIMENSION(3) :: CMOD
DATA MOD/&
1,1,1,&
1,1,1,&
1,1,0,&
0,1,0,&
1,0,0,&
0,0,0,&
0,0,0,      &
0,0,1,      &
0,0,0/
DATA CMOD/ &
'DEPOT   ',&
'CENTRAL ',&
'A3      '/
FORALL (I=1:3) NAME(I)=CMOD(I)
FORALL (I=1:3,J=1:9) IATT(I,J)=MOD(I,J)
IDNO=2
NCM=  3
NPAR=4
RETURN
END
```

Appendix I. The PREDPP Defined Data Items

1. Event Identification (EVID) Data Item

Value	Meaning
0	observation event
1	dose event
2	other-type event
3	reset event
4	reset-dose event

See Chapter V for a discussion of "extra" EVID's:
special names XVID1 XVID2 XVID3 XVID4 XVID5

2. Time (TIME) Data Item

Time of event

3. Dose Amount (AMT) Data Item

Value	Meaning
0	steady-state infusion
positive no.	amount of either bolus dose or infusion

Note: AMT data item must be 0 in a non-dose or non-reset-dose event record.

4. Dose Rate (RATE) Data Item

Value	Meaning
0	instantaneous bolus dose
-1	zero-order bolus dose, rate modeled in PK steady-state infusion, rate modeled in PK
-2	zero-order bolus dose, duration modeled in PK
positive no.	rate of infusion

Note: RATE data item must be 0 in a non-dose or non-reset-dose event record.

5. Steady-State (SS) Data Item

Value	Meaning
0	not a steady-state dose
1	a steady-state dose
2	a superimposed steady-state dose
3	Same as SS=1 but existing values of A are used as initial estimates (SS6, SS9, SS13)

Note: SS data item must be 0 in a non-dose or non-reset-dose event record.

6. Interdose Interval (II) Data Item

Value	Meaning
0	single non-steady-state dose or steady-state infusion
positive no.	interdose interval

Note: II data item must be 0 in a non-dose or non-reset-dose event record.

7. Compartment (CMT) Data Item

Value	Meaning
0	• event (dose, reset-dose, observation) applies to the default compartment • has no effect with an other-type or reset event
integer	• event (dose, reset-dose, observation) applies to the compartment with this number • for other-type or reset event, compartment with this number is turned on
-integer	• for observation event, and if output compartment has this number, output compartment is observed and then turned off • for other-type or reset event, compartment with this number is turned off • not allowed for dose or reset-dose event

Note: With a reset event, a compartment is turned on or off after all compartments are reset to their initial status.

When the integer is 1000, it designates the default compartment for output.

When the integer is 100 and the number of compartments is <=99, it is another way of designating the default compartment for output.

8. Prediction Compartment (PCMT) Data Item

Value	Meaning
0	prediction is the one that would result for an observation of the default observation compartment
integer	prediction is the one that would result for an observation of the compartment with this number

Note: PCMT data item is ignored in an observation event record.

When the integer is 1000, it designates the default compartment for output.

When the integer is 100 and the number of compartments is <=99, it is another way of designating the default compartment for output.

9. Call (CALL) Data Item

Value	Meaning
0	has no effect
1	call to ERROR with event record is forced
2	call to PK with event record is forced
3	calls to both PK and ERROR with event record are forced
10	Call to ADVAN9 is forced.

10. Continuation (CONT) Data Item

Value	Meaning
0	data record ends event record
1	event record is continued on next data record

11. Additional Dose (ADDL) Data Item

Value	Meaning
0	no additional doses, or dose is a steady-state infusion
integer	number of additional doses of same type

Note: ADDL data item must be 0 in a non-dose or non-reset-dose event record. Also, when ADDL data item is nonzero, II data item must be nonzero.

Appendix II. Another Example: Theophylline

The population theophylline example of Guide I, section F, can be specified using PREDPP, as shown in this appendix. A NONMEM control stream, and PK and ERROR routines, are given on the following pages. A NM-TRAN control stream, using abbreviated PK and ERROR codes, is also given. The results are essentially identical to those given in Guide I, figures 76-83. Unlike the example of Guide I, the data set is not embedded in the control stream; it is found in a separate file, also given in this appendix. (This is unrelated to the fact that PREDPP is used.) A file stream, pointing to the data file, is also given.

This example uses ADVAN2, which implements an analytic computation for the one compartment linear model with first-order absorption. The same example can be handled using ADVAN6 and ADVAN7, although it is less efficient to use subroutines implementing "numerical" computations for general models when an analytic ADVAN for a specific model can be used. However, it may be instructive for the user to examine how all three ADVAN's can be used for the same example, and to examine additional user-supplied codes to be used with these ADVAN's. Therefore, this appendix also gives PK, MODEL, DES, and TOL routines to be used with ADVAN6, and PK and MODEL routines to be used with ADVAN7.

Finally, NM-TRAN control streams corresponding to the use of ADVAN6 and ADVAN7 with this example are also given. These use the very same data set as does the NONMEM control stream. However, the last two data item types, EVID and MDV, are ignored. Though NM-TRAN can always properly process these data item types, it can ignore them with this example.

FCON Stream

```
FILE    FSTREAM
PROB    THEOPHYLLINE   POPULATION DATA
DATA       1   0 144   7   0   0   0   0
ITEM       1   4   7  11   1   0   0   0   0   0   0   0   0   0   0
INDX       6   3   2   0   0   0   0   0   0   0   0
XVID       0   0   0   0   0
MSDT     0.00000000000000E+000
LABL                           ID,                          DOSE
                             TIME,                            CP
                               WT,                          EVID
                              MDV
LBW1 IWRS,IPRD
     IRS,NPRED
     NRES,NWRES
     NIWRES,NIPRED
     NIRES,CPRED
     CRES,CWRES
     CIWRES,CIPRED
     CIRES,PREDI
     RESI,WRESI
     IWRESI,IPREDI
     IRESI,CPREDI
     CRESI,CWRESI
     CIWRESI,CIPREDI
     CIRESI,EPRED
     ERES,EWRES
     EIWRES,EIPRED
     EIRES,NPDE
     ECWRES,NPD
     OBJI
LRES SIGMA
LTHT
LETA
LEPS
FORM
(5F10.0,2F2.0)
STRC       3   3   1   0   0   0   1   1   0
STRC       1   3
THCN       1   0   0   0
THTA     3.000000000000000E+00, 8.000000000000000E-02, 4.000000000000000E-02
LOWR     1.000000000000000E-01, 8.000000000000000E-03, 4.000000000000000E-03
UPPR     5.000000000000000E+00, 5.000000000000000E-01, 9.000000000000000E-01
BLST     6.000000000000000E+00, 5.000000000000000E-03, 3.000000000000000E-01
         2.000000000000000E-04, 6.000000000000000E-03, 4.000000000000000E-01
DIAG     4.000000000000000E-01
DFLT      -1
ESTM       0 450   3   5   0   0   0   0   0   0   0   0   0   0   0   0   0   0
           0   0   0   0  -1  -1   0   0   0
```

```
BEST        -100        -100        -100        -100  -100.00000        -100
            -100        -100  -100.00000 -2147483647        -100        -100
            -100  -100.00000        -100        -100-0.10000E+03        -100
            -100        -100           0        -100        -100  -100.00000
            -100        -100        -100        -100-0.10000E+03-0.10000E+03
            -100        -100           1        -100        -100
            -100        -100 -2147483647        -100        -100-0.10000E+03
            -100-0.10000E+03-0.10000E+03        -100           0  -100
      -100  -100  -100-0.10000E+03-0.10000000E+03  -100
            -100        -100        -100        -100-0.10000000E+03-0.10000000E+03
    -0.10000000E+03-0.10000000E+03-0.10000000E+03-0.10000000E+03-0.10000000E+03-0.10000000E+03
    -0.10000000E+03-0.10000000+301-0.10000000+301
                                -100-100-100-100-100-100
            -100        -100        -100-0.10000000E+03-0.10000000E+03-0.10000000E+03
            -100        -100        -100-0.10000000E+03-0.10000000E+03-100
            -100        -100        -100        -100
BFIL
BDLM
BMUM
BGRD
ORDR
PFIL            -100
FFIL               0
COVR       0   0   0   0   0   0   1   0   0
COVT      -1  -1  -1  -1   0   0          -1    -1    -1    -1    -1    -1    -1-1-1TOS
              -1  -1  -1-0.10000E+03-0.10000E+03-0.10000E+03

CPAR              -1
TABL       1   1
PPAR              -1
TABL   1   4     1 0     2 0     5 0     3 0
           0   0   0   0   0           0           0             0-1 0
FRML
FRMR
FETA
SCAT       1   2
SCAT           3       9       1       1       0       0       0       0       0
               0       0       0      -1
SCAT           3      10       1       1       0       0       0       0       0
               0       0       0      -1
```

File Stream

```
DATA     FDATA
****
```

Data Set

1	4.02	0.		79.6	1	1
1		0.	.74		0	0
1		0.25	2.84		0	0
1		0.57	6.57		0	0
1		1.12	10.5		0	0
1		2.02	9.66		0	0
1		3.82	8.58		0	0
1		5.1	8.36		0	0
1		7.03	7.47		0	0
1		9.05	6.89		0	0
1		12.12	5.94		0	0
1		24.37	3.28		0	0
2	4.4	0.		72.4	1	1
2		0.	0.		0	0
2		.27	1.72		0	0
2		.52	7.91		0	0
2		1.	8.31		0	0
2		1.92	8.33		0	0
2		3.5	6.85		0	0
2		5.02	6.08		0	0
2		7.03	5.4		0	0
2		9.	4.55		0	0
2		12.	3.01		0	0
2		24.3	.90		0	0
3	4.53	0.		70.5	1	1
3		0.	0.		0	0
3		.27	4.4		0	0
3		.58	6.9		0	0
3		1.02	8.2		0	0
3		2.02	7.8		0	0
3		3.62	7.5		0	0
3		5.08	6.2		0	0
3		7.07	5.3		0	0
3		9.	4.9		0	0
3		12.15	3.7		0	0
3		24.17	1.05		0	0
4	4.4	0.		72.7	1	1
4		0.	0.		0	0
4		.35	1.89		0	0
4		.6	4.6		0	0
4		1.07	8.6		0	0
4		2.13	8.38		0	0
4		3.5	7.54		0	0
4		5.02	6.88		0	0
4		7.02	5.78		0	0
4		9.02	5.33		0	0
4		11.98	4.19		0	0
4		24.65	1.15		0	0

```
5      5.86      0.                 54.6  1 1
5                0.        0.             0 0
5                 .3       2.02           0 0
5                 .52      5.63           0 0
5                1.       11.4            0 0
5                2.02      9.33           0 0
5                3.5       8.74           0 0
5                5.02      7.56           0 0
5                7.02      7.09           0 0
5                9.1       5.9            0 0
5               12.        4.37           0 0
5               24.35      1.57           0 0
6      4.        0.                 80.   1 1
6                0.        0.             0 0
6                 .27      1.29           0 0
6                 .58      3.08           0 0
6                1.15      6.44           0 0
6                2.03      6.32           0 0
6                3.57      5.53           0 0
6                5.        4.94           0 0
6                7.        4.02           0 0
6                9.22      3.46           0 0
6               12.1       2.78           0 0
6               23.85       .92           0 0
7      4.95      0.                 64.6  1 1
7                0.         .15           0 0
7                 .25       .85           0 0
7                 .5       2.35           0 0
7                1.02      5.02           0 0
7                2.02      6.58           0 0
7                3.48      7.09           0 0
7                5.        6.66           0 0
7                6.98      5.25           0 0
7                9.        4.39           0 0
7               12.05      3.53           0 0
7               24.22      1.15           0 0
8      4.53      0.                 70.5  1 1
8                0.        0.             0 0
8                 .25      3.05           0 0
8                0.52      3.05           0 0
8                 .98      7.31           0 0
8                2.02      7.56           0 0
8                3.53      6.59           0 0
8                5.05      5.88           0 0
8                7.15      4.73           0 0
8                9.07      4.57           0 0
8               12.1       3.             0 0
8               24.12      1.25           0 0
9      3.1       0.                 86.4  1 1
9                0.        0.             0 0
```

9		.3	7.37		0	0
9		.63	9.03		0	0
9		1.05	7.14		0	0
9		2.02	6.33		0	0
9		3.53	5.66		0	0
9		5.02	5.67		0	0
9		7.17	4.24		0	0
9		8.8	4.11		0	0
9		11.6	3.16		0	0
9		24.43	1.12		0	0
10	5.5	0.		58.2	1	1
10		0.	.24		0	0
10		.37	2.89		0	0
10		.77	5.22		0	0
10		1.02	6.41		0	0
10		2.05	7.83		0	0
10		3.55	10.21		0	0
10		5.05	9.18		0	0
10		7.08	8.02		0	0
10		9.38	7.14		0	0
10		12.1	5.68		0	0
10		23.7	2.42		0	0
11	4.92	0.		65.	1	1
11		0.	0.		0	0
11		.25	4.86		0	0
11		.5	7.24		0	0
11		.98	8.		0	0
11		1.98	6.81		0	0
11		3.6	5.87		0	0
11		5.02	5.22		0	0
11		7.03	4.45		0	0
11		9.03	3.62		0	0
11		12.12	2.69		0	0
11		24.08	.86		0	0
12	5.3	0.		60.5	1	1
12		0.	0.		0	0
12		.25	1.25		0	0
12		.5	3.96		0	0
12		1.	7.82		0	0
12		2.	9.72		0	0
12		3.52	9.75		0	0
12		5.07	8.57		0	0
12		7.07	6.59		0	0
12		9.03	6.11		0	0
12		12.05	4.57		0	0
12		24.15	1.17		0	0

PK Routine To Be Used With ADVAN2

```
      SUBROUTINE PK(ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,IRGG,GG,NETAS)
!  PK SUBROUTINE FOR THE THEOPHYLLINE POPULATION DATA
!  USED WITH ADVAN2 AND TRANS1
!  CLEARANCE PROPORTIONAL TO WEIGHT
!  SCALING PARAMETER IS VOLUME/WEIGHT SINCE DOSE IS WEIGHT-ADJUSTED
      USE NMPRD4P
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE PRDIMS,    ONLY: GPRD,HPRD,GERD,HERD,GPKD
      USE NMPRD_REAL,ONLY: ETA,EPS
      USE NMPRD_INT, ONLY: MSEC=>ISECDER,MFIRST=>IFRSTDER,COMACT,COMSAV,IFIRSTEM
      USE NMPRD_INT, ONLY: MDVRES,ETASXI,NPDE_MODE,NOFIRSTDERCODE
      USE NMPRD_REAL, ONLY: DV_LOQ
      USE NMPRD_INT, ONLY: IQUIT
      USE PROCM_INT, ONLY: NEWIND=>PNEWIF
      USE NMBAYES_REAL, ONLY: LDF
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      SAVE
      INTEGER(KIND=ISIZE) :: FIRSTEM
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS,IRGG,NETAS
      DIMENSION :: IDEF(7,*),THETA(*),EVTREC(IREV,*),INDXS(*),GG(IRGG,GPKD+1,*)
      FIRSTEM=IFIRSTEM
      IF (ICALL <= 1) THEN
      CALL ASSOCNMPRD4
!  SET UP IDEF ARRAY:
      IDEF(   1,0001)=  -9
!  CALL PK ONCE PER INDIV. REC.
      IDEF(   1,0002)=   1
      IDEF(   1,0003)=   0
      IDEF(   1,0004)=   0
      IDEF(   2,0003)=   0
      IDEF(   2,0004)=   0
!  ROW INDEX FOR SCALING PARAMETER FOR COMPT. 2
      IDEF(   3,0002)=   4
      CALL GETETA(ETA)
      IF (IQUIT == 1) RETURN
      RETURN
      ENDIF
      IF (NEWIND /= 2) THEN
      IF (ICALL == 4) THEN
      CALL SIMETA(ETA)
      ELSE
      CALL GETETA(ETA)
      ENDIF
      IF (IQUIT == 1) RETURN
      ENDIF
 !  level            0
!  REGULAR CALL TO PK:
```

```
!  WEIGHT
      WT=EVTREC(NVNT,005)
      KA=THETA(001)+ETA(001)
      K=THETA(002)+ETA(002)
      CL=THETA(003)*WT+ETA(003)
      SC=CL/K/WT
      IF (FIRSTEM == 1) THEN
!                        A000031 = DERIVATIVE OF KA W.R.T. ETA(001)
      A000031=1.D0
!                        A000032 = DERIVATIVE OF K W.R.T. ETA(002)
      A000032=1.D0
      B000001=1.D0/K/WT
!                        A000034 = DERIVATIVE OF SC W.R.T. ETA(003)
      A000034=B000001
      B000002=-CL/K/K/WT
!                        A000035 = DERIVATIVE OF SC W.R.T. ETA(002)
      A000035=B000002*A000032
!  ELIMINATION RATE CONSTANT
      GG(0001,1,1)=K
      GG(0001,0003,1)=A000032
!  ABSORPTION RATE CONSTANT
      GG(0003,1,1)=KA
      GG(0003,0002,1)=A000031
!  SCALING
      GG(0004,1,1)=SC
      GG(0004,0003,1)=A000035
      GG(0004,0004,1)=A000034
      ELSE
      GG(0001,1,1)=K
      GG(0003,1,1)=KA
      GG(0004,1,1)=SC
      ENDIF
      IF (MSEC == 1) THEN
      B000004=-1.D0/K/K/WT
!                        A000037 = DERIVATIVE OF B000001 W.R.T. ETA(002)
      A000037=B000004*A000032
      B000005=-1.D0/K/K/WT
      B000006=CL/K/K/K/WT
!                        A000039 = DERIVATIVE OF B000002 W.R.T. ETA(002)
      A000039=B000006*A000032
      B000007=CL/K/K/K/WT
!                        A000040 = DERIVATIVE OF B000002 W.R.T. ETA(002)
      A000040=B000007*A000032+A000039
!                        A000041 = DERIVATIVE OF A000035 W.R.T. ETA(002)
      A000041=A000032*A000040
!                        A000042 = DERIVATIVE OF A000035 W.R.T. ETA(003)
      A000042=A000032*B000005
      GG(0004,0003,0003)=A000041
      GG(0004,0004,0003)=A000042
      ENDIF
```

```
      RETURN
      END
```

ERROR Routine To Be Used With ADVAN2, ADVAN6, and ADVAN7

```
!  ERROR SUBROUTINE FOR THE THEOPHYLLINE POPULATION DATA
      SUBROUTINE ERROR (ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,F,G,HH)
      USE NMPRD4P
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE PRDIMS,    ONLY: GPRD,HPRD,GERD,HERD,GPKD
      USE NMPRD_REAL,ONLY: ETA,EPS
      USE NMPRD_INT, ONLY: MSEC=>ISECDER,MFIRST=>IFRSTDER,IQUIT,IFIRSTEM
      USE NMPRD_INT, ONLY: MDVRES,ETASXI,NPDE_MODE,NOFIRSTDERCODE
      USE NMPRD_REAL, ONLY: DV_LOQ
      USE NMPRD_INT, ONLY: NEWL2
      USE PROCM_INT, ONLY: NEWIND=>PNEWIF
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      SAVE
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS
      DIMENSION :: IDEF(*),THETA(*),EVTREC(IREV,*),INDXS(*)
      REAL(KIND=DPSIZE) :: G(GERD,*),HH(HERD,*)
      INTEGER(KIND=ISIZE) :: FIRSTEM
      FIRSTEM=IFIRSTEM
      IF (ICALL <= 1) THEN
      CALL ASSOCNMPRD4
!  CALL ERROR ONLY ONCE
      IDEF(2)=2
      HH(1,1)=1.0D0
      IDEF(3)=000
      RETURN
      ENDIF
      IF (ICALL == 4) THEN
      IF (NEWL2 == 1) THEN
      CALL SIMEPS(EPS)
      IF (IQUIT == 1) RETURN
      ENDIF
      ENDIF
!  level            0
      Y=F+EPS(001)
      IF (FIRSTEM == 1) THEN !1
      ENDIF !1
      F=Y
      RETURN
      END
```

NM-TRAN Control Stream To Be Used With ADVAN2

```
$PROB  THEOPHYLLINE   POPULATION DATA
$INPUT      ID DOSE=AMT TIME CP=DV WT
$DATA       DATAFILE  (5F10.0)

$SUBROUTINES  ADVAN2

$PK
;THETA(1)=MEAN ABSORPTION RATE CONSTANT (1/HR)
;THETA(2)=MEAN ELIMINATION RATE CONSTANT (1/HR)
;THETA(3)=SLOPE OF CLEARANCE VS WEIGHT RELATIONSHIP (LITERS/HR/KG)
;SCALING PARAMETER=VOLUME/WT SINCE DOSE IS WEIGHT-ADJUSTED
   CALLFL=1
   KA=THETA(1)+ETA(1)
   K=THETA(2)+ETA(2)
   CL=THETA(3)*WT+ETA(3)
   SC=CL/K/WT

$THETA  (.1,3,5) (.008,.08,.5) (.004,.04,.9)
$OMEGA BLOCK(3)  6 .005 .0002 .3 .006 .4

$ERROR
   Y=F+EPS(1)

$SIGMA  .4

$EST     MAXEVAL=450  PRINT=5
$COV
$TABLE          ID DOSE WT TIME
$SCAT           (RES WRES) VS TIME BY ID
```

PK Routine To Be Used With ADVAN6 and ADVAN7

```
      SUBROUTINE PK(ICALL,IDEF,THETA,IREV,EVTREC,NVNT,INDXS,IRGG,GG,NETAS)
!  PK SUBROUTINE FOR THE THEOPHYLLINE POPULATION DATA
!  USED WITH ADVAN6 AND ADVAN7, AND TRANS1
!  CLEARANCE PROPORTIONAL TO WEIGHT
!  SCALING PARAMETER IS VOLUME/WEIGHT SINCE DOSE IS WEIGHT-ADJUSTED
      USE NMPRD4P
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE PRDIMS,    ONLY: GPRD,HPRD,GERD,HERD,GPKD
      USE NMPRD_REAL,ONLY: ETA,EPS
      USE NMPRD_INT, ONLY: MSEC=>ISECDER,MFIRST=>IFRSTDER,COMACT,COMSAV,IFIRSTEM
      USE NMPRD_INT, ONLY: MDVRES,ETASXI,NPDE_MODE,NOFIRSTDERCODE
      USE NMPRD_REAL, ONLY: DV_LOQ
      USE NMPRD_INT, ONLY: IQUIT
      USE PROCM_INT, ONLY: NEWIND=>PNEWIF
      USE NMBAYES_REAL, ONLY: LDF
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      REAL(KIND=DPSIZE) :: EVTREC
      SAVE
      INTEGER(KIND=ISIZE) :: FIRSTEM
      INTEGER(KIND=ISIZE) :: ICALL,IDEF,IREV,NVNT,INDXS,IRGG,NETAS
      DIMENSION :: IDEF(7,*),THETA(*),EVTREC(IREV,*),INDXS(*),GG(IRGG,GPKD+1,*)
      FIRSTEM=IFIRSTEM
      IF (ICALL <= 1) THEN
      CALL ASSOCNMPRD4
!  SET UP IDEF ARRAY:
      IDEF(   1,0001)=  -9
!  CALL PK ONCE PER INDIV. REC.
      IDEF(   1,0002)=   1
!  ROW INDEX FOR SCALING PARAMETER FOR COMPT. 2
      IDEF(   1,0003)=   0
      IDEF(   1,0004)=   0
      IDEF(   2,0003)=   0
      IDEF(   2,0004)=   0
      IDEF(   3,0002)=   3
      CALL GETETA(ETA)
      IF (IQUIT == 1) RETURN
      RETURN
      ENDIF
      IF (NEWIND /= 2) THEN
      IF (ICALL == 4) THEN
      CALL SIMETA(ETA)
      ELSE
      CALL GETETA(ETA)
      ENDIF
      IF (IQUIT == 1) RETURN
      ENDIF
 !  level           0
      WT=EVTREC(NVNT,005)
```

```
      KA=THETA(001)+ETA(001)
      KE=THETA(002)+ETA(002)
      CL=THETA(003)*WT+ETA(003)
      S2=CL/KE/WT
      P000001=KA
      P000002=KE
!  REGULAR CALL TO PK:
!  WEIGHT
      IF (FIRSTEM == 1) THEN
      B000001=1.D0/KE/WT
!                      A000034 = DERIVATIVE OF S2 W.R.T. ETA(003)
      A000034=B000001
      B000002=-CL/KE/KE/WT
!                      A000035 = DERIVATIVE OF S2 W.R.T. ETA(002)
      A000035=B000002
!                      A000046 = DERIVATIVE OF P000001 W.R.T. ETA(001)
      A000046=1.D0
!                      A000047 = DERIVATIVE OF P000002 W.R.T. ETA(002)
      A000047=1.D0
!  ABSORPTION RATE CONSTANT
      GG(0001,1,1)=P000001
      GG(0001,0002,1)=A000046
!  ELIMINATION RATE CONSTANT
      GG(0002,1,1)=P000002
      GG(0002,0003,1)=A000047
!  SCALING
      GG(0003,1,1)=S2
      GG(0003,0003,1)=A000035
      GG(0003,0004,1)=A000034
      ELSE
      GG(0001,1,1)=P000001
      GG(0002,1,1)=P000002
      GG(0003,1,1)=S2
      ENDIF
      IF (MSEC == 1) THEN
      B000005=-1.D0/KE/KE/WT
      B000006=-1.D0/KE/KE/WT
      B000007=CL/KE/KE/KE/WT
      B000008=CL/KE/KE/KE/WT
!                      A000041 = DERIVATIVE OF B000002 W.R.T. ETA(002)
      A000041=B000008+B000007
!                      A000042 = DERIVATIVE OF A000035 W.R.T. ETA(002)
      A000042=A000041
!                      A000043 = DERIVATIVE OF A000035 W.R.T. ETA(003)
      A000043=B000006
      GG(0003,0003,0003)=A000042
      GG(0003,0004,0003)=A000043
      ENDIF
      RETURN
      END
```

MODEL Routine To Be Used With ADVAN6

```
      SUBROUTINE MODEL (IDNO,NCM,NPAR,IR,IATT,LINK)
! DEFINES A 1 COMPARTMENT MODEL WITH DRUG DEPOT COMPARTMENT
! COMPT1: DRUG DEPOT   COMPT2: CENTRAL COMPT
      USE PRMOD_CHAR, ONLY: NAME
      USE SIZES,     ONLY: DPSIZE,ISIZE,SD
      USE PRDIMS,    ONLY: GPRD,HPRD,GERD,HERD,GPKD
      INTEGER(KIND=ISIZE) :: IDNO,NCM,NPAR,IR,IATT,LINK,I,J
      DIMENSION :: IATT(IR,*),LINK(IR,*)
      SAVE
      INTEGER(KIND=ISIZE), DIMENSION (2,7) :: MOD
      CHARACTER(LEN=SD), DIMENSION(2) :: CMOD
      DATA (MOD(I,  1),I=  1,  2)/&
! INITIAL STATUS: OFF ON
      0,1 /
      DATA (MOD(I,  2),I=  1,  2)/&
! ON/OFF ALLOWED: YES NO
      1,0 /
      DATA (MOD(I,  3),I=  1,  2)/&
! DOSE ALLOWED: YES YES
      1,1 /
      DATA (MOD(I,  4),I=  1,  2)/&
! DEFAULT FOR OBSERVATIONS: NO YES
      0,1 /
      DATA (MOD(I,  5),I=  1,  2)/&
! DEFAULT FOR DOSES: YES NO
      1,0 /
      DATA (MOD(I,  6),I=  1,  2)/&
      0,0 /
      DATA (MOD(I,  7),I=  1,  2)/&
      0,0 /
      DATA (CMOD(I),I=  1,  2) &
      /'DEPOT','CENTRAL'/
      FORALL (I=1:2) NAME(I)=CMOD(I)
      FORALL (I=1:2,J=1:7) IATT(I,J)=MOD(I,J)
      IDNO=9999
      NCM=  2
      NPAR=002
      RETURN
      END
```

DES Routine To Be Used With ADVAN6

```
      SUBROUTINE DES (A,P,T,DADT,IR,DA,DP,DT)
!  DES SUBROUTINE FOR THE THEOPHYLLINE POPULATION DATA
      USE NMPRD4P
      USE SIZES,     ONLY: DPSIZE,ISIZE
      USE PRDIMS,    ONLY: GPRD,HPRD,GERD,HERD,GPKD
      USE NMPRD_INT, ONLY: IERPRD,NETEXT,IQUIT
      USE NMPRD_CHAR,ONLY: ETEXT
      USE NMPRD_INT, ONLY: MSEC=>ISECDER,MFIRST=>IFRSTDER,IFIRSTEM,IFIRSTEMJAC
      USE PRCOM_INT, ONLY: MITER
      USE NMPRD_INT, ONLY: MDVRES,ETASXI,NPDE_MODE,NOFIRSTDERCODE
      USE NMPRD_REAL, ONLY: DV_LOQ
      USE PRMOD_INT, ONLY: ICALL=>ICALLD,IDEFD,IDEFA
      IMPLICIT REAL(KIND=DPSIZE) (A-Z)
      SAVE
      INTEGER(KIND=ISIZE) :: IR
      DIMENSION :: A(*),P(*),DADT(*),DA(IR,*),DP(IR,*),DT(*)
      INTEGER(KIND=ISIZE) :: FIRSTEM,IFIRSTEMJACIN
      IF(MITER==1.OR.MITER==4) IFIRSTEM=1
      FIRSTEM=IFIRSTEM
      IFIRSTEMJACIN=IFIRSTEMJAC
      IF(NOFIRSTDERCODE/=1) THEN
      IFIRSTEMJAC=FIRSTEM
      ELSE
      IFIRSTEMJAC=0
      ENDIF
      IF(IFIRSTEMJACIN==-2) RETURN
      IF (ICALL == 1) THEN
      CALL ASSOCNMPRD4
      IDEFD(1)=  0
      IDEFD(2)=0
      DA(   1,1)=0014280
      DA(   2,1)=0028441
      DA(   3,1)=0028560
      DA(   4,1)=0014311
      DA(   5,1)=0028472
      DA(   6,1)=0028592
      DA(   7,1)=0000000
      DP(   1,1)=0014280
      DP(   2,1)=0028441
      DP(   3,1)=0028560
      DP(   4,1)=0000000
      DT(   1)=0000000
      RETURN
      ENDIF
 !  level            0
 !  level            0
      DADT(1)=-P(001)*A(1)
      DADT(2)=P(001)*A(1)-P(002)*A(2)
```

```
      IF (FIRSTEM == 1) THEN ! 1
!                        E000004 = DERIVATIVE OF DADT(1) W.R.T. A(001)
      E000004=-P(001)
!                        F000081 = DERIVATIVE OF DADT(1) W.R.T. P(001)
      F000081=-A(1)
!                        E000006 = DERIVATIVE OF DADT(2) W.R.T. A(001)
      E000006=P(001)
!                        E000007 = DERIVATIVE OF DADT(2) W.R.T. A(002)
      E000007=-P(002)
!                        F000082 = DERIVATIVE OF DADT(2) W.R.T. P(001)
      F000082=A(1)
!                        F000083 = DERIVATIVE OF DADT(2) W.R.T. P(002)
      F000083=-A(2)
      ENDIF !1
      IF (MSEC == 1) THEN
!                        E000005 = DERIVATIVE OF F000081 W.R.T. A(001)
      E000005=-1.D0
!                        E000008 = DERIVATIVE OF F000082 W.R.T. A(001)
      E000008=1.D0
!                        E000009 = DERIVATIVE OF F000083 W.R.T. A(002)
      E000009=-1.D0
      ENDIF !msec
      IF (FIRSTEM == 1) THEN !2
      DA(   1,1)=E000004
      DA(   2,1)=E000006
      DA(   3,1)=E000007
      DP(   1,1)=F000081
      DP(   2,1)=F000082
      DP(   3,1)=F000083
      ENDIF !2
      IF (MSEC == 1) THEN
      DA(   4,1)=E000005
      DA(   5,1)=E000008
      DA(   6,1)=E000009
      ENDIF
      RETURN
      END
```

TOL Routine To Be Used With ADVAN6

```
!  TOL SUBROUTINE FOR THE THEOPHYLLINE POPULATION DATA
      SUBROUTINE TOL(NRD,ANRD,NRDC,ANRDC)
      USE SIZES,     ONLY: ISIZE
      INTEGER(KIND=ISIZE) :: NRD(0:*), ANRD(0:*), NRDC(0:*), ANRDC(0:*)
      NRD(1)=5
      RETURN
      END
```

NM-TRAN Control Stream To Be Used With ADVAN6

```
$PROB  THEOPHYLLINE   POPULATION DATA
$INPUT      ID DOSE=AMT TIME CP=DV WT
$DATA       DATAFILE  (5F10.0)

$SUBROUTINES  ADVAN6  TOL=5
$MODEL COMP=(DEPOT,INITIALOFF,DEFDOSE) COMP=(CENTRAL,DEFOBS,NOOFF)

$PK
;THETA(1)=MEAN ABSORPTION RATE CONSTANT (1/HR)
;THETA(2)=MEAN ELIMINATION RATE CONSTANT (1/HR)
;THETA(3)=SLOPE OF CLEARANCE VS WEIGHT RELATIONSHIP (LITERS/HR/KG)
;SCALING PARAMETER=VOLUME/WT SINCE DOSE IS WEIGHT-ADJUSTED
   CALLFL=1
   KA=THETA(1)+ETA(1)
   KE=THETA(2)+ETA(2)
   CL=THETA(3)*WT+ETA(3)
   S2=CL/KE/WT

$THETA  (.1,3,5) (.008,.08,.5) (.004,.04,.9)
$OMEGA BLOCK(3)  6 .005 .0002 .3 .006 .4

$DES
   DADT(1)=-KA*A(1)
   DADT(2)= KA*A(1)-KE*A(2)

$ERROR
   Y=F+EPS(1)

$SIGMA  .4

$EST     MAXEVAL=450  PRINT=5
$COV
$TABLE          ID DOSE WT TIME
$SCAT           (RES WRES) VS TIME BY ID
```

MODEL Routine To Be Used With ADVAN7

```
      SUBROUTINE MODEL (IDNO,NCM,NPAR,IR,IATT,LINK)
! DEFINES A 1 COMPARTMENT LINEAR MODEL WITH FIRST-ORDER ABSORPTION
! COMPT1: DRUG DEPOT   COMPT2: CENTRAL COMPT
      USE PRMOD_CHAR, ONLY: NAME
      USE SIZES,     ONLY: DPSIZE,ISIZE,SD
      USE PRDIMS,    ONLY: GPRD,HPRD,GERD,HERD,GPKD
      INTEGER(KIND=ISIZE) :: IDNO,NCM,NPAR,IR,IATT,LINK,I,J
      DIMENSION :: IATT(IR,*),LINK(IR,*)
      SAVE
      INTEGER(KIND=ISIZE), DIMENSION (2,7) :: MOD
      CHARACTER(LEN=SD), DIMENSION(2) :: CMOD
      DATA (MOD(I,  1),I=  1,  2)/&
! INITIAL STATUS: OFF ON
      0,1 /
      DATA (MOD(I,  2),I=  1,  2)/&
! ON/OFF ALLOWED: YES NO
      1,0 /
      DATA (MOD(I,  3),I=  1,  2)/&
! DOSE ALLOWED: YES YES
      1,1 /
      DATA (MOD(I,  4),I=  1,  2)/&
! DEFAULT FOR OBSERVATIONS: NO YES
      0,1 /
      DATA (MOD(I,  5),I=  1,  2)/&
! DEFAULT FOR DOSES: YES NO
      1,0 /
      DATA (MOD(I,  6),I=  1,  2)/&
      0,0 /
      DATA (MOD(I,  7),I=  1,  2)/&
      0,0 /
      DATA (CMOD(I),I=  1,  2) &
! COMPARTMENT NAMES
      /'DEPOT','CENTRAL'/
      FORALL (I=1:2) NAME(I)=CMOD(I)
      FORALL (I=1:2,J=1:7) IATT(I,J)=MOD(I,J)
      IDNO=9999
      NCM=  2
      NPAR=002
! SET LINK:
! K12
      LINK(  1,002)=001
! K20
      LINK(  2,003)=002
      RETURN
      END
```

NM-TRAN Control Stream To Be Used With ADVAN7

```
$PROB  THEOPHYLLINE   POPULATION DATA
$INPUT      ID DOSE=AMT TIME CP=DV WT
$DATA       DATAFILE  (5F10.0)

$SUBROUTINES  ADVAN7
$MODEL COMP=(DEPOT,INITIALOFF,DEFDOSE) COMP=(CENTRAL,DEFOBS,NOOFF)

$PK
;THETA(1)=MEAN ABSORPTION RATE CONSTANT (1/HR)
;THETA(2)=MEAN ELIMINATION RATE CONSTANT (1/HR)
;THETA(3)=SLOPE OF CLEARANCE VS WEIGHT RELATIONSHIP (LITERS/HR/KG)
;SCALING PARAMETER=VOLUME/WT SINCE DOSE IS WEIGHT-ADJUSTED
   CALLFL=1
   K12=THETA(1)+ETA(1)
   K20=THETA(2)+ETA(2)
   CL=THETA(3)*WT+ETA(3)
   S2=CL/K20/WT

$THETA  (.1,3,5) (.008,.08,.5) (.004,.04,.9)
$OMEGA BLOCK(3)  6 .005 .0002 .3 .006 .4

$ERROR
   Y=F+EPS(1)

$SIGMA  .4

$EST     MAXEVAL=450  PRINT=5
$COV
$TABLE          ID DOSE WT TIME
$SCAT           (RES WRES) VS TIME BY ID
```

Appendix III. MTIME Details

III.A. GENERAL DESCRIPTION

Model event times are additional PK parameters defined in the PK routine or $PK block. A model event time is not associated with any compartment, but, like an absorption lag time, defines a time to which the system is advanced. When the time is reached, certain new indicator variables are set and a call to PK is made. At this call (and/or subsequent to this call) PK or DES or AES or ERROR can use the indicator variables to change some aspect of the system, e.g., a term in a differential equation, or the rate of an infusion. This feature may be used with any ADVAN routine and does not require that extra compartments be added to the model or that dummy doses be included in the data set. There may be up to PCT model event times, where PCT is defined in PSIZES and TSIZES. Its current value is 30.

Model times are independent of non-event dose times. See below (Rules for coincidence of times) when model times happen to coincide with other times.

In abbreviated code, the model event times have reserved names MTIME(i), where i is an integer from 1 to PCT.

The indicator variables currently implemented have reserved names MNEXT(i) and MPAST(i), each corresponding to the MTIME(i) variable with the same index i. An indicator variable may be used in the PK, ERROR, DES, or AES routine (as a right-hand quantity in abbreviated codes). An indicator variable may not be used on the right (other than in a write/ print statement) unless the corresponding MTIME has already been defined. It is not required that one or more indicator variables be used.

There is no requirement that the MTIME(i) be in ascending order. That is, MTIME(i) may be less than, equal to, or greater than MTIME(i+1). Any MTIME(i) may be negative or have the value 0, in which case MPAST(i)=1 always and MNEXT(i)=0 always.

There may be gaps in the sequence of any length. E.g., PK may define MTIME(i) and MTIME(i+2) but not MTIME(i+1). This has the same effect as including the code MTIME(i+1)=0.

PK may redefine MTIME(i). See "Example 3" below.

An ETA may be used in the definition of MTIME(i).

PREDPP sets all indicator variables to 0 at start of an IR and at RESET events.

MNEXT(i)=1 when PREDPP calls PK prior to the advance to MTIME(i), and remains 1 during the advance from the previous time to MTIME(i). Otherwise, MNEXT(i)=0. (The previous time may be an event time, non-event time, or model event time).

MPAST(i)=0 until the call to PK subsequent to the one for which MNEXT(i)=1. At that call MPAST(i)=1. It then retains this value, unless MTIME is redefined, in which case MPAST will be appropriately redefined as another step function.

Another related variable is MNOW. MNOW=i if this is a call to PK such that MNEXT(i)=1 for some i. MNOW=0 otherwise.

Model event time MTIME is not a dose-related feature and has no effect on steady-state doses. See Note 3 of chapter V.

III.B. EXAMPLES

Example of indicator variable values

Suppose there are three MTIMES, and MTIME(1)<MTIME(2)<MTIME(3).

```
           MTIME(1)      MTIME(2)      MTIME(3)
 ............|(.............|(.............|(................time axis...
```

```
  MNEXT(1)=1     MNEXT(2)=1    MNEXT(3)=1
                 MPAST(1)=1
                               MPAST(2)=1
                                              MPAST(3)=1
```

Another way to draw the same picture:

```
          MTIME(1)      MTIME(2)      MTIME(3)
 ............|..............|..............|..................
 011111111111100000000000000000000000000000000000000   MNEXT(1)
 000000000000011111111111111100000000000000000000000   MNEXT(2)
 000000000000000000000000000011111111111111100000000   MNEXT(3)
 000000000000011111111111111111111111111111111111111   MPAST(1)
 000000000000000000000000000011111111111111111111111   MPAST(2)
 000000000000000000000000000000000000000000011111111   MPAST(3)
```

If the relationship MTIME(1)<MTIME(2)<MTIME(3) does not hold, imagine the picture above with indices i, j, k (a permutation of 1, 2, 3) such that MTIME(i)<MTIME(j)<MTIME(k)

Event times (both dose and non-dose) can occur anywhere above. See below (Rules for coincidence of times).

If model event times depend on ETAS, so do the values of indicator variables. However, the effect of the model event times or of the indicator variables on the eta-derivatives of the state vector is handled by PREDPP in such a way that the indicator variables are *not* random variables, and their eta-derivatives are always 0.

Examples of use of model time parameters

Example 1. Flexible input

There is an infusion with modeled rate.
The rate is 400*EXP(ETA(1)) from time 0 to 1.5
The rate is 300*EXP(ETA(2)) from time 1.5 to 2.5
The rate is 200*EXP(ETA(3)) from time 2.5 till end of infusion.

Code in $PK:

```
      MTIME(1)=1.5
      MTIME(2)=2.5
      R1=   400*EXP(ETA(1))*(1-MPAST(1))
      R1=R1+300*EXP(ETA(2))*(MPAST(1)-MPAST(2))
      R1=R1+200*EXP(ETA(3))*MPAST(2)
      WRITE (82,*) TIME,MNOW,MNEXT(1),MNEXT(2),MPAST(1),MPAST(2),R1,TSTATE
```

Events at times 0, 1, 3.
Values in fort.82 are

TIME	MNOW	MNEXT1	MNEXT2	MPAST1	MPAST2	R1	TSTATE
0.00	0.00	0.00	0.00	0.00	0.00	400.00	0.00
1.00	0.00	0.00	0.00	0.00	0.00	400.00	0.00
3.00	1.00	1.00	0.00	0.00	0.00	400.00	1.00
3.00	2.00	0.00	1.00	1.00	0.00	300.00	1.50
3.00	0.00	0.00	0.00	1.00	1.00	200.00	2.50

Example 2. Here are three versions of an EHC control stream.
(EHC stands for Enterohepatic Recycling.)
All give identical results. The transfer of drug from

compt. 4 to 1 is controlled by "FLAG":

```
DADT(1)=-KA*A(1)+K41*A(4)*FLAG
DADT(4)=K1G*A(2)-K41*A(4)*FLAG
```

1) No use of model event times.
Define 2 dummy compts. (5 and 6) and dummy doses with lag times

```
CID DOSE TIME DV EVID CMT
101 10    0    0   1    2
101 1     0    0   1    5
101 1     0    0   1    6
101 0   2.0    0   0    2
```

In $PK

```
ALAG5=THETA(8)*EXP(ETA(8))
DELTA=THETA(9)*EXP(ETA(9))
ALAG6=ALAG5+DELTA
```

1a) Define FLAG in $DES

```
$DES
 FLAG=0.0
 IF (T.GE.ALAG5) FLAG=1
 IF (T.GE.ALAG5.AND.DOSTIM.EQ.ALAG5) FLAG=0
 IF (T.GE.ALAG5.AND.DOSTIM.EQ.0.AND.TIME.EQ.ALAG5) FLAG=0
 IF (T.GE.ALAG6) FLAG=0
 IF (T.GE.ALAG6.AND.DOSTIM.EQ.ALAG6) FLAG=1
 IF (T.GE.ALAG6.AND.DOSTIM.EQ.0.AND.TIME.EQ.ALAG6) FLAG=1
```

1b) Define FLAG in $PK

```
$PK
 IF (NEWIND.LT.2) THEN
 FLAG=0.0
 OLDTIM=0
 ENDIF
 IF (OLDTIM.EQ.ALAG5) FLAG=1
 IF (OLDTIM.EQ.ALAG6) FLAG=0
 OLDTIM=DOSTIM
```

2) Use of model event times.
No dummy compartments, no dummy doses

```
CID DOSE TIME DV EVID CMT
101 10    0    0   1    2
101 0   2.0    0   0    2
```

2a) Define MTIMES and FLAG in $PK

```
$PK
  MTIME(1)=THETA(8)*EXP(ETA(8))
  DELTA=THETA(9)*EXP(ETA(9))
```

```
MTIME(2)=MTIME(1)+DELTA
FLAG=MPAST(1)-MPAST(2)
```

2b) Define FLAG in $DES.
Might give better run time than 2a.
In 2a, FLAG is considered by NMTRAN to be an implicit basic PK parameter and hence its eta derivatives (which are always 0) are included in the chain rule calculations. In 2b, FLAG is known not to have derivatives because it is DES-defined.

```
$PK
   MTIME(1)=THETA(8)*EXP(ETA(8))
   DELTA=THETA(9)*EXP(ETA(9))
   MTIME(2)=MTIME(1)+DELTA
$DES
   FLAG=MPAST(1)-MPAST(2)
```

Example 3. MTIME's may be changed.

Suppose there are events at times 0 and 10 but one wants to advance in increments of 1 with stops at times 1, 2, 3, ... , 9, 10.
This code does not work:

```
IF (TIME.EQ.0) THEN
 MTIME(1)=0
ELSE
 MTIME(1)=MTIME(1)+1
ENDIF
```

Instead, use

```
IF (TIME.EQ.0) THEN
 TEMP=0
ELSE
 TEMP=TEMP+1
ENDIF
MTIME(1)=TEMP
```

Reason: If a basic or additional PK parameters is set conditionally in the $PK block, NM-TRAN inserts statements setting it to 0 so that, if it is not set by the $PK block, its value is 0. In the code that does not work, MTIME(1) does not retain its value from one call to the next. In the code that does work, TEMP (being a PK-defined item that does not depend on etas) does retain its value.

Other Examples of the Use of Model Time Parameters

"Model Time Examples".

Some of the above examples are also found in Guide VIII and in on-line help entry for examples.

"Using MTIME to Model Periodic Discontinuities in $DES"

Contains important examples of modelling circadian rhythm. In Guide VIII and in on-line help entry for examples. This help entry also describes the following control streams.

Indirect response model

idr_circexa.ctl in the NONMEM examples directory.

Step Function model

step_circexa.ctl in the NONMEM examples directory.

See ehc.exa (Enterhepatic circulation example). This gives a fully worked-out example for EHC. It cannot be used with Steady-State doses. This example also gives an alternative model, a smooth step model using Hill terms in a sigmoid emax model. This may be used with Steady-State doses.

III.C. RULES FOR COINCIDENCE OF TIMES

(1) MTIME(i)=MTIME(j)

There are two calls to PK. Assuming i<j, the calls are described by:

```
        MNEXT(i) MNEXT(j) MPAST(i) MPAST(j)  MNOW

call #1     1        0        0        0      1
call #2     0        1        1        0      2
```

(2) MTIME(i)=ALAGj

MNEXT(i) and DOSTIM are set on separate calls to PK.
First, MNEXT(i). Then, DOSTIM.
E.g., code in $PK is

```
ALAG1=.5
MTIME(1)=ALAG1
WRITE (82,*) TIME,MNOW,MNEXT(1),MPAST(1),DOSTIM,TSTATE
```

Events at times 0, 1, 3.
Values in fort.82 are

```
     TIME   MNOW  MNEXT  MPAST DOSTIM TSTATE
     0.00   0.00   0.00   0.00   0.00     0
     1.00   1.00   1.00   0.00   0.00     0
     1.00   0.00   0.00   1.00   0.50    .5
     1.00   0.00   0.00   1.00   0.00    .5
     3.00   0.00   0.00   1.00   0.00   1.0
```

(3) MTIME(i)=event time

Two calls. First MTIME(1). Then, event time.

E.g., code in $PK is

```
MTIME(1)=1
WRITE (82,*) TIME,MNOW,MNEXT(1),MPAST(1),TSTATE
```

Events at times 0, 1, 3
Values in fort.82 are

```
     TIME   MNOW   MNEXT  MPAST  TSTATE
     0.00   0.00   0.00   0.00   0
     1.00   1.00   1.00   0.00   0
     1.00   0.00   0.00   1.00   1
```

```
      3.00   0.00   0.00   1.00   1
```

compare with

(3a) ALAGj=event time:

Two calls. First, ALAG1. Then, event time.
E.g., code in $PK is

```
   ALAG1=1
   WRITE (72,*) TIME,DOSTIM,TSTATE
```

Events at times 0, 1, 3
Values in fort.72 are

```
      TIME  DOSTIM TSTATE
      0.00   0.00      0
      1.00   1.00      0
      1.00   0.00      1
      3.00   0.00      1
```

(4) MTIME(i)=additional dose time

MNEXT(i) and DOSTIM are set on separate calls to PK.
First, MNEXT(i). Then, DOSTIM.

E.g., code in $PK is

```
   MTIME(1)=.5
   WRITE (82,*) TIME,MNOW,MNEXT(1),MPAST(1),DOSTIM,TSTATE
```

Events at times 0, 1, 3.
Dose at time 0 specifies ADDL=1, II=.5
Values in fort.82 are

```
      TIME   MNOW  MNEXT  MPAST DOSTIM TSTATE
      0.00   0.00   0.00   0.00   0.00 0.00
      1.00   1.00   1.00   0.00   0.00 0.00
      1.00   0.00   0.00   1.00   0.50 0.50
      1.00   0.00   0.00   1.00   0.00 0.50
      3.00   0.00   0.00   1.00   0.00 1.00
```

(5) MTIME(i)=additional dose time = event time

There are three calls to PK.
MNEXT(i) and DOSTIM are set on separate calls to PK.
First, MNEXT(i). Then, DOSTIM. Finally, a call at event time.

E.g., code in $PK is

```
   MTIME(1)=1
   WRITE (82,*) TIME,MNOW,MNEXT(1),MPAST(1),DOSTIM,TSTATE
```

Events at times 0, 1, 3.
Dose at time 0 specifies ADDL=1, II=1
Values in fort.82 are

```
      TIME   MNOW  MNEXT  MPAST DOSTIM TSTATE
      0.00   0.00   0.00   0.00   0.00   0.00
      1.00   1.00   1.00   0.00   0.00   0.00
      1.00   0.00   0.00   1.00   1.00   1.00
      1.00   0.00   0.00   1.00   0.00   1.00
      3.00   0.00   0.00   1.00   0.00   1.00
```

compare with

(5a) MTIME(i)= lagged dose time = event time

There are three calls to PK.
MNEXT(i) and DOSTIM are set on separate calls to PK.
First, MNEXT(i). Then, DOSTIM. Finally, a call at event time.

E.g., code in $PK is

```
MTIME(1)=1
ALAG1=1
WRITE (82,*) TIME,MNOW,MNEXT(1),MPAST(1),DOSTIM,TSTATE
```

Events at times 0, 1, 3.
Dose at time 0 specifies ADDL=1, II=1
Values in fort.82 are

```
TIME   MNOW  MNEXT  MPAST DOSTIM TSTATE
0.00   0.00   0.00   0.00   0.00   0.00
1.00   1.00   1.00   0.00   0.00   0.00
1.00   0.00   0.00   1.00   1.00   1.00
1.00   0.00   0.00   1.00   0.00   1.00
3.00   0.00   0.00   1.00   2.00   1.00
3.00   0.00   0.00   1.00   0.00   2.00
```

III.D. IMPLEMENTATION DETAILS

MTIME parameters are defined in a MODULE. The declaration is:

```
MODULE PKERR_REAL
 USE SIZES, ONLY : DPSIZE,PCT
 REAL(KIND=DPSIZE) :: MTIME(PCT)
END MODULE PKERR_REAL
```

where the size is given in SIZES.f90 by PARAMETER (PCT=30) and PCT can be changed with the $SIZES record.

In FSUBS, the generated PK routine contains

```
USE PKERR_REAL,ONLY: MTIME
```

PK contains code to copy MTIME parameters and their eta derivatives to GG. If there are gaps in the sequence of MTIME parameters, the corresponding positions in GG are left as-is. Since GG is initialized to 0 before calls to PK, the effective value of omitted MTIME parameters is 0.

The MTIME array is never listed in NMPRD4. Therefore, if an MTIME parameter is to be displayed in a table or scatter it must be copied to a PK-defined variable in NMPRD4. E.g.,

```
MTIME(1)=...
MT1=MTIME(1)
 ...
$TABLE MT1
```

Another reason for defining MTIME in a MODULE is to make it available to the ERROR routine via the same USE statement. All other basic and additional parameters are available to ERROR only if listed in NMPRD4. If MTIME is defined in ERROR, then the values of the MTIME parameters are available as well. However, eta derivatives are not included in MTIME, only in GG. (This is done to save space and compute time.) Hence the ERROR routine should not use MTIME parameters in such a way as to influence the value of Y if those parameters have eta derivatives. NM-TRAN gives an appropriate warning if an MTIME variable is used in the $ERROR block.

Values of MTIME may be used in DES and AES, where they are obtained from the subroutine argument P (which is really GG) and may have eta derivatives.

Appendix IV. Compact Arrays in DES

IV.A. FULL ARRAYS AND THE CHAIN RULE

With general non-linear models ADVAN6, ADVAN8, ADVAN9, ADVAN13, ADVAN14, and ADVAN15, a DES subroutine is needed.

```
SUBROUTINE DES (A,P,T,DADT,IR,DA,DP,DT)
DIMENSION :: A(*),P(*),DADT(*),DA(IR,*),DP(IR,*),DT(*)
```

If DES sets IDEFD(2)=1, the arrays are full.

As explained in Chapter VI Section C,

The partial of DADT(i) wrt A(k) is stored in DA(i,k).
The partial of DADT(i) wrt P(k) is stored in DP(i,k).
The partial of DADT(i) wrt T   is stored in DT(i).

These arguments are used by PREDPP subroutines such as FCN1 as follows.

The state vector consists of compartment amounts A_i and their eta partials $\frac{\partial A_i}{\partial \eta_j}$.

The ADVAN routines must integrate over time t both of the following, for each compartment i and eta j:

$\frac{dA_i}{dt}$ and $\frac{d}{dt}\frac{\partial A_i}{\partial \eta_j}$

The latter is not known to PREDPP, but can be computed by changing the order of differentiation:

$$\frac{d}{dt}\frac{\partial A_i}{\partial \eta_j} = \frac{\partial}{\partial \eta_j}\frac{dA_i}{dt}$$

The total derivative $\frac{\partial}{\partial \eta_j}\frac{dA_i}{dt}$ can be obtained by the chain rule by adding the sum over all compartments A_k and parameters P_k and time T

$$\frac{d}{dA_k}\left(\frac{dA_i}{dt}\right)\cdot\frac{\partial A_k}{\partial \eta_j}\ ... \quad + \quad \frac{d}{dP_k}\left(\frac{dA_i}{dt}\right)\cdot\frac{\partial P_k}{\partial \eta_j}\ ... \quad + \quad \frac{d}{dT}\left(\frac{dA_i}{dt}\right)\cdot\frac{\partial T}{\partial \eta_j}$$

But $\frac{d}{dA_k}\left(\frac{dA_i}{dt}\right)$ is simply DA(i,k) and $\frac{d}{dP_k}\left(\frac{dA_i}{dt}\right)$ is DP(i,k) and $\frac{d}{dT}\left(\frac{dA_i}{dt}\right)$ is DT(i).

The derivative $\frac{d}{dt}\frac{\partial A_i}{\partial \eta_j}$ can thus be computed as the sum of quantities that are available to the FCN routine:

$$\mathrm{DA}(i,k)\cdot\frac{\partial A_k}{\partial \eta_j}\ ... \quad + \quad \mathrm{DP}(i,k)\cdot\frac{\partial P_k}{\partial \eta_j}\ ... \quad + \quad \mathrm{DT}(i)\cdot\frac{\partial T}{\partial \eta_j}$$

This can be extended easily to second derivatives.

Note:

(1) For every PK parameter (i.e., every variable defined in PK and used in DES) there is a loop over all the etas. If a given PK parameter does not have partials with respect to etas, the loop does a lot of multiplying by zero's and adding of zero's. This wastes time. It is better simply to define it in $DES.

(2) There is no third dimension of DA or DP or DT for the second partials (e.g., partials of DADT with respect to A(k),A(n)), or cross-partials (e.g., partials of DADT with respect to A(k),P(n)).

Therefore the Laplacian method cannot be used with the FULL arrays,

(3) FCN1 always loops over all elements of DA(i,j), DP(i,j) and DT(i). With a large sparse system of differential equations, many of these elements are zero because they are never computed by DES. With compact arrays (below), the elements that are always zero are not included in the computation.

IV.B. COMPACT ARRAYS

If DES sets IDEFD(2)=0, the arrays are compact. (This is the default). FCN1 uses only those derivatives that are actually computed by DES.† In addition, second partials (Laplace method) can be computed by DES.

DES computes all first and second partials and cross-partials of DADT. These may include (for all compartments and parameters defined in the model):

In DA:
the first partials of DADT(i) wrt A(j)
the second partials of DADT(i) wrt A(j) A(k)
the cross partials of DADT(i) wrt A(j) P(k)
In DP:
the first partials of DADT(i) wrt P(j)
the second partials of DADT(i) wrt P(j) P(k)
In DT:
the first partials of DADT(i) wrt T
the second partials of DADT(i) wrt T T
the cross partials of DADT(i) wrt T A(j)
the cross partials of DADT(i) wrt T P(j)

At ICALL=1, DES uses the first row DA(*,1) to describe each position in DA that will contain a value and describes ("maps") which A and eta it corresponds to (and which P, in the case of second partials). At other values of ICALL, DES puts the values themselves in DA(*,1). Similarly for DP and DT. Thus DA, DP, DT are linear (e.g., only the first row DA(*,1) is used for both the ICALL=1 "map" and the ICALL=2 value.)

The mapping uses two constants:
PG=max. no. of basic and additional PK parameters); may be set by $SIZES PG
PC=max. no. of compartments + 1; may be set by $SIZES PC

From these are computed

MM=MAX(100,PC-1+PG+10)
MMSQ=MM*MM

For DA, the mapping is in position p is:

DA(p,1)=DA(i*MMSQ+j*MM+k,1)

Where:
i tells which DADT(i)
j tells which A(j)
k=0 if this is a first partial. k>0 if this is a second partial.
if k < PC, k tells which A(k)
if k > PC, k tells which P(n), where n=k-PC.

†Some derivatives may happen to be zero for some values of A's and P's and etas, but presumably they are non-zero at other values.

Position p in DA will be used to store the value at ICALL=2

The DA values are terminated by DA(..)=0

Similarly for partials of DADT(i) wrt P:

DP(i*MMSQ+j*MM+k,1)

Where
j tells which P(j)
k=0 if this is a first partial. k>0 if this is a second partial.
if k > 0, k tells which P(k)

Similarly for partials of DADT(i) wrt T:

DT(i*MMSQ+k,1)

Where
k=0 if this is a first partial. k>0 if this is a second partial.
if 0< k < PC, this is partial wrt T, A(k)
if k > PC and k < PC+PG, this is partial wrt T, P(n) where n=k-PC
if k = PC+PG+1, this is second partial wrt T T

IV.C. EXAMPLE

An example is given as app4example.ctl. This is not a realistic model, but serves to demonstrate the mapping. The only differential equation is:

```
DADT(1)=-T*T*A(1)*A(2)*P(1)*P(2)
```

The example contains contains

```
$SIZES PC=5 PG=10
```

Then MM=100 and MMSQ=10000 and the mapping given by the right sides can be read easily.

For DA, right sides have the form 00i0j0k.
The DA values in DES at ICALL=1 are:

```
DA(   1,1)=0010100
DA(   2,1)=0010200
DA(   3,1)=0010201
DA(   4,1)=0010102
DA(   5,1)=0010106
DA(   6,1)=0010107
DA(   7,1)=0010206
DA(   8,1)=0010207
```

1st partial of DADT(1) wrt A(1) will be stored in DA(1,1)
1st partial of DADT(1) wrt A(2) will be stored in DA(2,1)
2nd partial of DADT(1) wrt A(2) A(1) will be stored in DA(3,1)
2nd partial of DADT(1) wrt A(1) A(2) will be stored in DA(4,1)
2nd partial of DADT(1) wrt A(1) P(1) will be stored in DA(5,1)
2nd partial of DADT(1) wrt A(1) P(2) will be stored in DA(6,1)
2nd partial of DADT(1) wrt A(2) P(1) will be stored in DA(7,1)
2nd partial of DADT(1) wrt A(2) P(2) will be stored in DA(8,1)

For DP, right sides have the form 00i0j0k.
The DP values in DES at ICALL=1 are:

```
DP(   1,1)=0010100
DP(   2,1)=0010200
```

```
      DP(   3,1)=0010201
      DP(   4,1)=0010102
      DP(   5,1)=0
```

1st partial of DADT(1) wrt P(1) will be stored in DP(1,1)
1st partial of DADT(1) wrt P(2) will be stored in DP(2,1)
2nd partial of DADT(1) wrt P(2) P(1) will be stored in DP(3,1)
2nd partial of DADT(1) wrt P(1) P(2) will be stored in DP(4,1)

For DT, right sides have the form: 00i000k
The DT values in DES at ICALL=1 are:

```
      DT(   1)=0010000
      DT(   2)=0010001
      DT(   3)=0010002
      DT(   4)=0010016
      DT(   5)=0010006
      DT(   6)=0010007
      DT(   7)=0
```

1st partial of DADT(1) wrt T will be stored in DT(1)
2nd partial of DADT(1) wrt T A(1) will be stored in DT(2)
2nd partial of DADT(1) wrt T A(2) will be stored in DT(3)
2nd partial of DADT(1) wrt T T will be stored in DT(4)
2nd partial of DADT(1) wrt T P(1) will be stored in DT(5)
2nd partial of DADT(1) wrt T P(2) will be stored in DT(6)

Here is the app4example.ctl:

```
$SIZES PC=5 PG=10
$PROB  DEMONSTRATE COMPACT  DES MAPPING
$INPUT      ID DOSE=AMT TIME CP=DV WT
$DATA       THEOPP RECS=1

$SUBROUTINES  ADVAN6  TOL=5
$MODEL NCM=2

$PK
   P(1)=THETA(1)*EXP(ETA(1))
   P(2)=THETA(2)*EXP(ETA(2))

$DES
   DADT(1)=-T*T*A(1)*A(2)*P(1)*P(2)

$ERROR
   Y=F+EPS(1)

$THETA   1 2
$OMEGA   1 2
$SIGMA     1
```