Mlxtran

MonolixSuite Language Reference

www.mlxtran.lixoft.com

Structural model

```
DESCRIPTION:
[LONGITUDINAL]
  input = { }
PK:
EQUATION:
DEFINITION:
OUTPUT:
  output = { }
  table  = { }
```

DESCRIPTION: Optional text describing the model

[LONGITUDINAL] section
Contains the structural model with:

- **input = { } list**
parameters that are estimated or used as regressor variables.
- **PK: block**
permits to define PK models using macros, and to link the administration information of the data set with the model.
- **EQUATION: block**
mathematical equations including ODEs and DDEs.
- **DEFINITION: block**
used to define a random variable and its probability distribution.
- **OUTPUT: block**
contains the [LONGITUDINAL] section outputs.
  - **output = { } list**
  identifies the predictions or the modeled outputs that are fitted against the data set observations.
  - **table = { } list**
  parameters or variables outputted in the result folder of Monolix.

Modeling discrete data with DEFINITION:

Time-to-event model

DEFINITION:
Event = {type=event, maxEventNumber=1, hazard=h}

- **Event:** name of the random variable, can be replaced by any name. It should be listed in the outputs, to be matched to the observed data.
- **hazard**: hazard function, can be defined via an expression in EQUATION:.
- Indicating the maximum number of events in the **maxEventNumber** argument speeds up calculations.

Count model

DEFINITION:
CountNumber = {type=count, P(CountNumber=k) = ...}

- **CountNumber:** name of the random variable, can be replaced by any name. It should be listed in the outputs, to be matched to the observed data.
- **k** is a mandatory name for the values. The probability mass function **P(CountNumber=k)** should be defined as a function of k and individual parameters.
- It is possible to define directly the log of the probability mass function with: log(P(CountNumber=k)) = ...}

Categorical model

DEFINITION:
level = {type = categorical,
categories = {0, 1, 2},
logit(P(level <=0)) = th1
logit(P(level <=1)) = th1 + th2}

- **categories**: list of ordered categories, as increasing successive integers.
- **P(Y=i)**: probability of a given category integer i, for the observation named Y. A transformed probability can be provided instead of a direct one. The transformation can be log, logit, or probit.
- The model is completely defined by the probability mass functions P(Y=i) for each category, or the cumulative probabilities P(Y <= i).
- When the value of a probability can be deduced from others, its definition can be spared.

SimulationsPlus

PK: macros

Arguments

Compartment characteristics	
amount	Variable for drug amount in the compartment
concentration	Variable for drug concentration in the compartment
volume, V	Compartment volume
Administration	
adm	Administration type to map with ADMINISTRATION ID from dataset, otional: default value is 1
Targets	
cmt	Label of compartment (integer)
target	ODE variable
Absorption	
Tk0	Zero-order duration
ka	First-order rate
p	Fraction of absorbed drug
Delays	
Tlag	Lag time
Ktr	Transit rate
Mtt	Mean transit time
Transfers	
kt	Transfer rate from one compartment to another
kij, kji	Transfer rates between compartments i and j
ke0	Transfer rate to an effect compartment
Elimination	
k	Elimination rate
Cl	Clearance
Vm, Km	Michaelis-Menten elimination

pkmodel macro

pkmodel(V, k/Cl/(Vm, Km),
Tlag/(Ktr, Mtt), p, Tk0/ka,
(k12, k21), (k13, k31), ke0)
Defines common PK models with a list of parameters. Single administration type and single elimination only.

Administration macros

Administration macros apply the doses from the dataset to the model. Dose types indicated in the column ADMINISTRATION ID are mapped with the argument adm.

Targeting a compartment

absorption(adm = ..., cmt = ...,
Tlag/(Ktr, Mtt), p, Tk0/ka)
For first-order or zero-order absorption arriving in cmt compartment.

iv(adm = ..., cmt = ..., Tlag, p**)**
For bolus or infusion into cmt.

Targeting an ODE variable

depot(adm = ..., target = ...,
Tlag/(Ktr, Mtt), p, Tk0/ka)
For first-order or zero-order absorption, bolus or infusion. Amount applied to the ODE variable in target.

reset(adm = ..., target = ...**)**
Resets the target variable to its initial value at the corresponding dosing times in the dataset (dose value not used).

empty(adm = ..., target = ...**)**
Sets the target variable to 0 at the corresponding dosing times in the dataset (dose value not used).

Compartment macros

compartment(cmt = ...,
amount/concentration = ...,
volume = ...**)**
Defines a compartment that can be used in other macros. Needs to be defined first.

peripheral(kij, kji,
amount/concentration = ...,
volume = ...**)**
Defines a peripheral compartment of label j with two transfers of drug amount from and toward the compartment of label i.

effect(cmt = ..., ke0,
concentration = ...**)**
Defines an effect compartment with a transfer of drug from compartment cmt.

transfer(from = i, to = j, kt**)**
Unidirectional transfer process from compartment i to compartment j.

Elimination macro

elimination(cmt = ...,
k/Cl/(Vm, Km)**)**
Defines an elimination from compartment cmt.

Legend:
/: Mutually exclusive
( ): Mutually dependent
Mandatory arguments

Syntax for equations and ODEs

Syntax

ddt_x: time derivative of variable x (x can be any name)
t_0: initial value of time
x_0: initial value of variable x
if, elseif, else, end: conditional statements
a*b, a/b, a^b: math operators
a==b, a=<b, a<b, a>=b, a&b , a|b: logical operators
exp, log, log10, sqrt, cos, sin, factln, max, min ...: math functions

Keywords

t: time
tDose: time of the last administered dose
amtDose: amount of the last administered dose
inftDose: infusion time of the last administered dose
delay(x,tau): delay function for DDEs
odeType = stiff: use a stiff ODE solver (add in EQUATION:)
Comments begin with ;

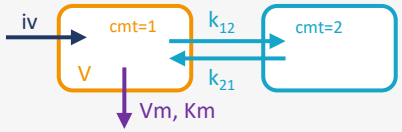
Mlxtran

MonolixSuite Language Reference: examples

SimulationsPlus

www.simulations-plus.com

Two-compartment model with iv bolus/infusion and Michaelis-Menten elimination

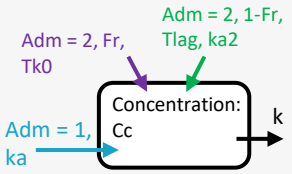


This model is available in the PK models library.

```
[LONGITUDINAL]
input = {V, Vm, Km, k12, k21}

EQUATION:
Cc = pkmodel(V, Vm, Km, k12, k21)

OUTPUT:
output = Cc
```

Two formulations: 1st order absorption and mixed 0-order/ 1st order absorption

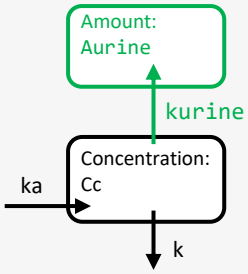
```
[LONGITUDINAL]
input = {V, ka, Tk0, ka2, Fr, k}

PK:
compartment(cmt = 1, volume = V, concentration = Cc)
absorption(adm = 1, cmt = 1, ka)
absorption(adm = 2, cmt = 1, Tk0, p = Fr)
absorption(adm = 2, cmt = 1, Tlag = Tk0, ka = ka2,
           p = 1-Fr)
elimination(cmt = 1, k)

OUTPUT:
output = Cc
```



The two formulations should be distinguished in the dataset with the column ADMINISTRATION ID.

PK-urine: joint model with plasma concentration and amount in urine

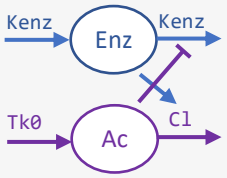
```
[LONGITUDINAL]
input = {ka, Cl, V, p_urine}

PK:
depot(adm = 1, target = Ac, ka)
empty(adm = 2, target = Aurine)

EQUATION:

k_urine = p_urine*Cl/V
k_non_urine = (1-p_urine)*Cl/V

t_0      = 0
Ac_0     = 0
Aurine_0 = 0
ddt_Ac     = - k_non_urine*Ac - k_urine*Ac
ddt_Aurine = k_urine*Ac
Cc = Ac/V
thalf = log(2)*V/Cl

OUTPUT:
output = {Cc, Aurine}
table  = {thalf}
```



Times of emptying of urine compartment are encoded in the dataset as pseudo-doses with ADMINISTRATION ID = 2.
A table of values for the elimination half-life `thalf` is outputted in the result folder at each observation time.

Auto-induction model for time-varying clearance

```
[LONGITUDINAL]
input={Tk0, V, Cl, Kenz, IC50}

PK:
depot(target=Ac, Tk0)

EQUATION:
t_0 = 0
Ac_0 = 0
Enz_0 = 1

ddt_Ac = - Cl/V*Ac*Enz
ddt_Enz = Kenz - Kenz * (1 -
          Cc/(Cc+IC50)) * Enz
Cc = Ac/V

OUTPUT:
output = {Cc}
```



The drug stimulates its own metabolism via induction of the metabolic enzyme expression.

PK – PD – TTE: joint model for plasma concentration, tumor volume and death

```
[LONGITUDINAL]
input = {V, Cl, Q, V2, Vm, Km, Mini, kp, kd,
         R, lambda0, betaM}

;plasma concentration model
PK:
compartment(cmt = 1, amount = Ac, volume = V,
            concentration = Cc)
peripheral(k12 = Q/V, k21 = Q/V2)
elimination(cmt = 1, Cl)
elimination(cmt = 1, Vm, Km)
iv(adm = 1, cmt = 1)

;tumor growth model
EQUATION:
odeType = stiff
t_0 = 0
M_0 = Mini
ddt_M = (kp - kd*Cc*exp(-R*t))*M
Msat = min(1000, M)

;time-to-event (death) model
lambda = lambda0*exp(betaM*Msat)
DEFINITION:
death = {type = event, eventType = exact,
         maxEventNumber = 1, hazard = lambda}

OUTPUT:
output = {Cc, M, death}
```

The statement **odeType = stiff** permits to use a stiff (implicit) ODE solver.

Dose-dependent bioavailability

Using dose-related keywords: amtDose

```
[LONGITUDINAL]
input = {ka, k, V, D50}

PK:
F = amtDose / (amtDose + D50)
Cc = pkmodel(ka, V, k, p = F)

OUTPUT:
output = {Cc}
```

Using a regressor containing the dose information

```
[LONGITUDINAL]
input = {ka, k, V, DoseReg, D50}
DoseReg = {use = regressor}

PK:
F = DoseReg / (DoseReg + D50)
Cc = pkmodel(ka, V, k, p = F)

OUTPUT:
output = {Cc}
```

Count model: zero - inflated Poisson model

```
[LONGITUDINAL]
input = {lambda0, nu, f}

EQUATION:
lambda = lambda0*exp(-t/nu)

DEFINITION:
CountNumber = {type = count,
  if k==0
    Pk = exp(-lambda)*(1-f) + f
  else
    Pk = exp(k*log(lambda) - lambda
         - factln(k))*(1-f)
  end
  P(CountNumber = k) = Pk}

OUTPUT:
output = CountNumber
```

The output is the number of events (in {0, ..., infinity}) with an inflation of zero counts. Lambda decreases exponentially over time.