Test individual functions in saemix 3.0

Emmanuelle

05/10/2020

Setup

- set up work directories
- two versions toggled by testMode
  - if testMode is FALSE, load the functions in R
  - if testMode is TRUE, use testthat functions

Classes

Data: SaemixData object

- code below is the interactive version of testthat for data classes
  - **TODO** fix problems with testthat (probably call with helper functions)
- **TODO**
  - silence the warnings "NA introduits"
  - problem reading binary data "Column name(s) do(es) not exist in the dataset, please check"

```
# SaemixData class
## From data on disk
namtest<-"Creating SaemixData object from file on disk\n"
cat(namtest)
```

```
## Creating SaemixData object from file on disk
```

```
x<-try(saemixData(name.data=file.path(datDir,"theo.saemix.tab"),header=TRUE,sep=" ",na=NA, name.group=c
```

```
## Reading data from file /home/eco/work/saemix/saemixextension/data/theo.saemix.tab
## These are the first lines of the dataset as read into R. Please check the format of the data is appr
##   Id    Dose Time Concentration Weight Sex
## 1  1 319.992 0.25          2.84   79.6   1
## 2  1 319.992 0.57          6.57   79.6   1
## 3  1 319.992 1.12         10.50   79.6   1
## 4  1 319.992 2.02          9.66   79.6   1
## 5  1 319.992 3.82          8.58   79.6   1
## 6  1 319.992 5.10          8.36   79.6   1
## [1] "Weight" "Sex"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
```

```
##     longitudinal data for use with the SAEM algorithm
## Dataset /home/eco/work/saemix/saemixextension/data/theo.saemix.tab
##     Structured data: Concentration ~ Dose + Time | Id
##     X variable for graphs: Time (hr)
##     covariates: Weight (kg), Sex (-)
##       reference class for covariate Sex :  0
```

```
if(is(x, "try-error")) cat("Problem in",namtest)

## From data as a dataframe in the environment
namtest<-"Creating SaemixData object from dataframe\n"
cat(namtest)
```

```
## Creating SaemixData object from dataframe
```

```
theo.saemix<-read.table(file.path(datDir,"theo.saemix.tab"),header=T,na=".")
x<-try(saemixData(name.data=theo.saemix,header=TRUE,sep=" ",na=NA, name.group=c("Id"),name.predictors=c
```

```
## [1] "Weight" "Sex"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset theo.saemix
##     Structured data: Concentration ~ Dose + Time | Id
##     X variable for graphs: Time (hr)
##     covariates: Weight (kg), Sex (-)
##       reference class for covariate Sex :  0
```

```
if(is(x, "try-error")) cat("Problem in",namtest)

# SaemixRepData class
namtest<-"Creating SaemixRepData object\n"
cat(namtest)
```

```
## Creating SaemixRepData object
```

```
xrep<-new(Class="SaemixRepData",data=x)
print(xrep)
```

```
## Object of class saemixRepData
##     replicated data used in the SAEM algorithm
##     number of subjects in initial dataset 12
##     number of replications 1
##     number of subjects in replicated dataset 12
```

```
if(is(x, "try-error")) cat("Problem in",namtest)

# SaemixSimData class
namtest<-"Creating SaemixSimData object\n"
cat(namtest)
```

```
## Creating SaemixSimData object
```

```
xrep<-new(Class="SaemixSimData",data=x)
print(xrep)
```

```
## Object of class SaemixSimData
##     data simulated according to a non-linear mixed effect model
## Characteristics of original data
##     number of subjects: 12
##     summary of response:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.850   3.513   5.665   5.447   7.325  11.400
## Characteristics of simulated data
##     no simulations performed yet
```

```
if(is(x, "try-error")) cat("Problem in",namtest)
```

Model: SaemixModel object

Testing simple models

- Changes made
  - define modelType at the beginning of initialize to allow empty objects to be printed out
  - added a test for empty models in print to print out an appropriate message
- **TODO**
  - print function for empty models returns NULL, would rather it returned nothing
  - check if function to validate covariance model works
    - change name for consistency (no underscore)

```
# Empty model
namtest<-"Creating empty SaemixModel object\n"
cat(namtest)
```

```
## Creating empty SaemixModel object
```

```
xmod<-new(Class="SaemixModel")
print(xmod)
```

```
## Nonlinear mixed-effects model
## No model function set yet
```

```
## NULL
```

```
if(is(xmod, "try-error")) cat("Problem in",namtest)

# Minimal model
namtest<-"Creating minimal SaemixModel object\n"
cat(namtest)
```

```
## Creating minimal SaemixModel object
```

```
  model1cpt<-function(psi,id,xidep) {
    dose<-xidep[,1]
    tim<-xidep[,2]
    ka<-psi[id,1]
    V<-psi[id,2]
    CL<-psi[id,3]
    k<-CL/V
    ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
    return(ypred)
  }
xmod<-saemixModel(model=model1cpt, psi0=matrix(c(1.,20,0.5), ncol=3,byrow=TRUE, dimnames=list(NULL, c("
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function  Model type:  structural
## function(psi,id,xidep) {
##     dose<-xidep[,1]
##     tim<-xidep[,2]
##     ka<-psi[id,1]
##     V<-psi[id,2]
##     CL<-psi[id,3]
##     k<-CL/V
##     ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
##     return(ypred)
##   }
##   Nb of parameters: 3
##       parameter names:  ka V CL
##       distribution:
##      Parameter Distribution Estimated
## [1,] ka        normal       Estimated
## [2,] V         normal       Estimated
## [3,] CL        normal       Estimated
##   Variance-covariance matrix:
##    ka V CL
## ka  1 0  0
## V   0 1  0
## CL  0 0  1
##   Error model: constant , initial values: a.1=1
##     No covariate in the model.
##     Initial values
##              ka  V  CL
## Pop.CondInit  1 20 0.5
```

```
if(is(xmod, "try-error")) cat("Problem in",namtest)

# Model with all elements
namtest<-"Creating full SaemixModel object\n"
cat(namtest)
```

```
## Creating full SaemixModel object
```

```
xmod<-saemixModel(model=model1cpt,description="One-compartment model with first-order absorption", psi0=
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  One-compartment model with first-order absorption  Model type:  structural
## function(psi,id,xidep) {
##     dose<-xidep[,1]
##     tim<-xidep[,2]
##     ka<-psi[id,1]
##     V<-psi[id,2]
```

```
##     CL<-psi[id,3]
##     k<-CL/V
##     ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
##     return(ypred)
##   }
##   Nb of parameters: 3
##       parameter names:  ka V CL
##       distribution:
##      Parameter Distribution Estimated
## [1,] ka        log-normal   Estimated
## [2,] V         log-normal   Estimated
## [3,] CL        log-normal   Estimated
##   Variance-covariance matrix:
##    ka V CL
## ka  1 0  0
## V   0 1  1
## CL  0 1  1
##   Error model: combined , initial values: a.1=1 b.1=0.5
##   Covariate model:
##      ka V CL
## [1,]  0 0  1
## [2,]  0 0  0
##     Initial values
##                ka  V    CL
## Pop.CondInit 1.0 20  0.50
## Cov.CondInit 0.1  0 -0.01
```

```
if(is(xmod, "try-error")) cat("Problem in",namtest)
```

Results: SaemixRes object

- created testthat (short)
- added a test to vcov to handle empty objects
  - print, fitted, etc work as expected
  - added some messages for empty objects or not available types
- **TODO**
  - resid() or fitted() don't work, I need to use resid.SaemixRes, but I should be able to dispatch based on argument type like vcov

```
xres<-new(Class="SaemixRes")
print(xres)
```

```
## No fit performed yet.
```

```
## NULL
```

```
resid.SaemixRes(xres)
```

```
## No residuals of type ires available
```

```
fitted.SaemixRes(xres)
```

```
## No fitted values of type ipred available
```

```
vcov(xres)
```

```
## NULL
```

Fitted object: SaemixObject object

```
# Control options
xopt<-saemixControl()
cat("K1=",xopt$nbiter.saemix," nb.SA=",xopt$nbiter.sa,"\n")
```

```
## K1= 300 100  nb.SA= 150
```

```
# Empty object
smx.data<-saemixData(name.data=file.path(datDir,"theo.saemix.tab"),header=T,na=".", name.group=c("Id"),n
model1cpt<-function(psi,id,xidep) {
  dose<-xidep[,1]
  tim<-xidep[,2]
  ka<-psi[id,1]
  V<-psi[id,2]
  CL<-psi[id,3]
  k<-CL/V
  ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
  return(ypred)
}
smx.model<-saemixModel(model=model1cpt,description="One-compartment model with first-order absorption",
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  One-compartment model with first-order absorption  Model type:  structural
## function(psi,id,xidep) {
##   dose<-xidep[,1]
##   tim<-xidep[,2]
##   ka<-psi[id,1]
##   V<-psi[id,2]
##   CL<-psi[id,3]
##   k<-CL/V
##   ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
##   return(ypred)
## }
##   Nb of parameters: 3
##       parameter names:  ka V CL
##       distribution:
##      Parameter Distribution Estimated
## [1,] ka        log-normal   Estimated
## [2,] V         log-normal   Estimated
## [3,] CL        log-normal   Estimated
##   Variance-covariance matrix:
##    ka V CL
## ka  1 0  0
## V   0 1  1
## CL  0 1  1
##   Error model: combined , initial values: a.1=1 b.1=0.5
##   Covariate model:
##      ka V CL
## [1,]  0 0  1
```

```
## [2,]  0 0  0
##     Initial values
##               ka  V    CL
## Pop.CondInit 1.0 20  0.50
## Cov.CondInit 0.1  0 -0.01
```

```
smx.opt<-saemixControl(nb.chains=5,nbiter.saemix = c(500,300), ipar.lmcmc = 100)
x<-createSaemixObject.empty(smx.model,smx.data,smx.opt)
print(x)
```

```
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data             ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset /home/eco/work/saemix/saemixextension/data/theo.saemix.tab
##     Structured data: Concentration ~ Dose + Time | Id
##     X variable for graphs: Time (hr)
##     covariates: Weight (-), Sex (-)
##       reference class for covariate Sex :  0
## Dataset characteristics:
##     number of subjects:     12
##     number of observations: 120
##     average/min/max nb obs: 10.00  /  10  /  10
## First 10 lines of data:
##    Id    Dose  Time Concentration Weight Sex mdv cens occ ytype
## 1   1 319.992  0.25          2.84   79.6   1   0    0   1     1
## 2   1 319.992  0.57          6.57   79.6   1   0    0   1     1
## 3   1 319.992  1.12         10.50   79.6   1   0    0   1     1
## 4   1 319.992  2.02          9.66   79.6   1   0    0   1     1
## 5   1 319.992  3.82          8.58   79.6   1   0    0   1     1
## 6   1 319.992  5.10          8.36   79.6   1   0    0   1     1
## 7   1 319.992  7.03          7.47   79.6   1   0    0   1     1
## 8   1 319.992  9.05          6.89   79.6   1   0    0   1     1
## 9   1 319.992 12.12          5.94   79.6   1   0    0   1     1
## 10  1 319.992 24.37          3.28   79.6   1   0    0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  One-compartment model with first-order absorption  Model type:  structural
## function(psi,id,xidep) {
##   dose<-xidep[,1]
##   tim<-xidep[,2]
##   ka<-psi[id,1]
##   V<-psi[id,2]
##   CL<-psi[id,3]
##   k<-CL/V
##   ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
##   return(ypred)
## }
##   Nb of parameters: 3
##       parameter names:  ka V CL
##       distribution:
```

```
##      Parameter Distribution Estimated
## [1,] ka        log-normal   Estimated
## [2,] V         log-normal   Estimated
## [3,] CL        log-normal   Estimated
##   Variance-covariance matrix:
##    ka V CL
## ka  1 0  0
## V   0 1  1
## CL  0 1  1
##   Error model: combined , initial values: a.1=1 b.1=0.5
##   Covariate model:
##        ka V CL
## Weight  0 0  1
## Sex     0 0  0
##     Initial values
##               ka  V    CL
## Pop.CondInit 1.0 20  0.50
## Cov.CondInit 0.1  0 -0.01
## ----------------------------------
## ----    Key algorithm options  ----
## ----------------------------------
##     Estimation of individual parameters (MAP)
##     Estimation of standard errors and linearised log-likelihood
##     Estimation of log-likelihood by importance sampling
##     Number of iterations:  K1=500, K2=300
##     Number of chains:  5
##     Seed:  23456
##     Number of MCMC iterations for IS:  5000
##     Simulations:
##         nb of simulated datasets used for npde:  1000
##         nb of simulated datasets used for VPC:  100
##     Input/output
##         save the results to a file:  TRUE
##         save the graphs to files:  TRUE
##         directory where results should be saved:  newdir
## ------------------------------------------------
## ----                  Results                   ----
## No fit performed yet.

## NULL
```

```
show(x)
```

```
## Nonlinear mixed-effects model fit by the SAEM algorithm
## ----------------------------------------
## ----         Data and Model          ----
## ----------------------------------------
## Data
##     Dataset /home/eco/work/saemix/saemixextension/data/theo.saemix.tab
##     Longitudinal data: Concentration ~ Dose + Time | Id
##
## Model:
##     One-compartment model with first-order absorption
##      3 parameters: ka V CL
##      error model: combined
```

```
##      covariate model:
##        ka V CL
## Weight  0 0  1
## Sex     0 0  0
##
## Key options
##     Estimation of individual parameters (MAP)
##     Estimation of standard errors and linearised log-likelihood
##     Estimation of log-likelihood by importance sampling
##     Number of iterations:  K1=500, K2=300
##     Number of chains:  5
##     Seed:  23456
##     Number of MCMC iterations for IS:  5000
##     Input/output
##         save the results to a file:  TRUE
##         save the graphs to files:  TRUE
##         directory where results are saved:  newdir
```

NLMEM fits

Continuous response model

Main fit

- Theophylline data

```
# Theophylline data, base model
theo.saemix<-read.table(file.path(datDir,"theo.saemix.tab"),header=T)
saemix.data<-saemixData(name.data=theo.saemix,header=TRUE,sep=" ",na=NA,
                        name.group=c("Id"),name.predictors=c("Dose","Time"),
                        name.response=c("Concentration"),name.covariates=c("Weight","Sex"),
                        units=list(x="hr",y="mg/L",covariates=c("kg","-")), name.X="Time", verbose = FA

model1cpt<-function(psi,id,xidep) {
  dose<-xidep[,1]
  tim<-xidep[,2]
  ka<-psi[id,1]
  V<-psi[id,2]
  CL<-psi[id,3]
  k<-CL/V
  ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
  return(ypred)
}

# Model with covariate Weight
saemix.model<-saemixModel(model=model1cpt,modeltype="structural",
                          description="One-compartment model with first-order absorption",
                          psi0=matrix(c(1.,20,0.5,0.1,0,-0.01),ncol=3,byrow=TRUE, dimnames=list(NULL, c
                          transform.par=c(1,1,1),covariate.model=matrix(c(0,0,1,0,0,0),ncol=3,byrow=TRU

saemix.options<-list(seed=632545,save=FALSE,save.graphs=FALSE, displayProgress=FALSE)
saemix.fit<-saemix(saemix.model,saemix.data,saemix.options)
```

```
## Nonlinear mixed-effects model fit by the SAEM algorithm
## ----------------------------------
## ----          Data              ----
## ----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset theo.saemix
##     Structured data: Concentration ~ Dose + Time | Id
##     X variable for graphs: Time (hr)
##     covariates: Weight (kg), Sex (-)
##       reference class for covariate Sex :  0
## Dataset characteristics:
##     number of subjects:     12
##     number of observations: 120
##     average/min/max nb obs: 10.00  /  10  /  10
## First 10 lines of data:
##    Id    Dose  Time Concentration Weight Sex mdv cens occ ytype
## 1   1 319.992  0.25          2.84   79.6   1   0    0   1     1
## 2   1 319.992  0.57          6.57   79.6   1   0    0   1     1
## 3   1 319.992  1.12         10.50   79.6   1   0    0   1     1
## 4   1 319.992  2.02          9.66   79.6   1   0    0   1     1
## 5   1 319.992  3.82          8.58   79.6   1   0    0   1     1
## 6   1 319.992  5.10          8.36   79.6   1   0    0   1     1
## 7   1 319.992  7.03          7.47   79.6   1   0    0   1     1
## 8   1 319.992  9.05          6.89   79.6   1   0    0   1     1
## 9   1 319.992 12.12          5.94   79.6   1   0    0   1     1
## 10  1 319.992 24.37          3.28   79.6   1   0    0   1     1
## ----------------------------------
## ----          Model             ----
## ----------------------------------
## Nonlinear mixed-effects model
##   Model function:  One-compartment model with first-order absorption  Model type:  structural
## function(psi,id,xidep) {
##   dose<-xidep[,1]
##   tim<-xidep[,2]
##   ka<-psi[id,1]
##   V<-psi[id,2]
##   CL<-psi[id,3]
##   k<-CL/V
##   ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
##   return(ypred)
## }
## <bytecode: 0x56264ddc1a90>
##   Nb of parameters: 3
##       parameter names:  ka V CL
##       distribution:
##      Parameter Distribution Estimated
## [1,] ka        log-normal   Estimated
## [2,] V         log-normal   Estimated
## [3,] CL        log-normal   Estimated
##   Variance-covariance matrix:
##    ka V CL
## ka  1 0  0
## V   0 1  0
```

```
## CL  0 0  1
##   Error model: constant , initial values: a.1=1
##   Covariate model:
##        [,1] [,2] [,3]
## Weight    0    0    1
##     Initial values
##               ka  V    CL
## Pop.CondInit 1.0 20  0.50
## Cov.CondInit 0.1  0 -0.01
## ----------------------------------
## ----    Key algorithm options  ----
## ----------------------------------
##     Estimation of individual parameters (MAP)
##     Estimation of standard errors and linearised log-likelihood
##     Estimation of log-likelihood by importance sampling
##     Number of iterations:  K1=300, K2=100
##     Number of chains:  5
##     Seed:  632545
##     Number of MCMC iterations for IS:  5000
##     Simulations:
##         nb of simulated datasets used for npde:  1000
##         nb of simulated datasets used for VPC:  100
##     Input/output
##         save the results to a file:  FALSE
##         save the graphs to files:  FALSE
## ----------------------------------------------------
## ----                  Results                    ----
## ----------------------------------------------------
## -----------------  Fixed effects  ------------------
## ----------------------------------------------------
##      Parameter       Estimate SE    CV(%) p-value
## [1,] ka               1.573   0.300  19.1 -
## [2,] V               31.524   1.410   4.5 -
## [3,] CL               1.587   1.005  63.3 -
## [4,] beta_Weight(CL)  0.008   0.009 113.3 0.19
## [5,] a.1              0.742   0.057   7.7 -
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##    Parameter Estimate SE     CV(%)
## ka omega2.ka 0.385    0.1738 45
## V  omega2.V  0.016    0.0094 58
## CL omega2.CL 0.068    0.0333 49
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##           omega2.ka omega2.V omega2.CL
## omega2.ka 1         0        0
## omega2.V  0         1        0
## omega2.CL 0         0        1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
```

```
##       -2LL= 343.4026
##       AIC = 359.4026
##       BIC = 363.2818
##
## Likelihood computed by importance sampling
##       -2LL= 344.6988
##       AIC = 360.6988
##       BIC = 364.5781
## ----------------------------------------------------
```

```
# Model with 2 covariates and a covariance model
saemix.model3<-saemixModel(model=model1cpt,modeltype="structural",
          description="One-compartment model with first-order absorption",
          psi0=matrix(c(1.,20,0.5,0.1,0,-0.01),ncol=3,byrow=TRUE, dimnames=list(NULL, c("ka","V","CL"))),
          covariance.model=matrix(c(1,0,0,0,1,1,0,1,1),ncol=3,byrow=TRUE),
          covariate.model=matrix(c(0,0,1,0,1,0),ncol=3,byrow=TRUE),
          transform.par=c(1,1,1),error.model="proportional")
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  One-compartment model with first-order absorption  Model type:  structural
## function(psi,id,xidep) {
##   dose<-xidep[,1]
##   tim<-xidep[,2]
##   ka<-psi[id,1]
##   V<-psi[id,2]
##   CL<-psi[id,3]
##   k<-CL/V
##   ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
##   return(ypred)
## }
## <bytecode: 0x56264ddc1a90>
##   Nb of parameters: 3
##       parameter names:  ka V CL
##       distribution:
##      Parameter Distribution Estimated
## [1,] ka        log-normal   Estimated
## [2,] V         log-normal   Estimated
## [3,] CL        log-normal   Estimated
##   Variance-covariance matrix:
##    ka V CL
## ka  1 0  0
## V   0 1  1
## CL  0 1  1
##   Error model: proportional , initial values: b.1=1
##   Covariate model:
##      ka V CL
## [1,]  0 0  1
## [2,]  0 1  0
##     Initial values
##               ka  V    CL
## Pop.CondInit 1.0 20  0.50
```

```
## Cov.CondInit 0.1  0 -0.01
```

```
saemix.options<-list(seed=12345,save=FALSE,save.graphs=FALSE, displayProgress=FALSE)
saemix.fit3<-saemix(saemix.model3,saemix.data,saemix.options)
```

```
## Error in optim(par = phi1, fn = conditional.distribution_c, phii = phii,  :
##   la fonction ne peut être évaluée aux paramètres initiaux
## Nonlinear mixed-effects model fit by the SAEM algorithm
## ----------------------------------
## ----          Data             ----
## ----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset theo.saemix
##     Structured data: Concentration ~ Dose + Time | Id
##     X variable for graphs: Time (hr)
##     covariates: Weight (kg), Sex (-)
##       reference class for covariate Sex :  0
## Dataset characteristics:
##     number of subjects:     12
##     number of observations: 120
##     average/min/max nb obs: 10.00  /  10  /  10
## First 10 lines of data:
##    Id    Dose  Time Concentration Weight Sex mdv cens occ ytype
## 1   1 319.992  0.25          2.84   79.6   1   0    0   1     1
## 2   1 319.992  0.57          6.57   79.6   1   0    0   1     1
## 3   1 319.992  1.12         10.50   79.6   1   0    0   1     1
## 4   1 319.992  2.02          9.66   79.6   1   0    0   1     1
## 5   1 319.992  3.82          8.58   79.6   1   0    0   1     1
## 6   1 319.992  5.10          8.36   79.6   1   0    0   1     1
## 7   1 319.992  7.03          7.47   79.6   1   0    0   1     1
## 8   1 319.992  9.05          6.89   79.6   1   0    0   1     1
## 9   1 319.992 12.12          5.94   79.6   1   0    0   1     1
## 10  1 319.992 24.37          3.28   79.6   1   0    0   1     1
## ----------------------------------
## ----          Model            ----
## ----------------------------------
## Nonlinear mixed-effects model
##   Model function:  One-compartment model with first-order absorption  Model type:  structural
## function(psi,id,xidep) {
##   dose<-xidep[,1]
##   tim<-xidep[,2]
##   ka<-psi[id,1]
##   V<-psi[id,2]
##   CL<-psi[id,3]
##   k<-CL/V
##   ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
##   return(ypred)
## }
## <bytecode: 0x56264ddc1a90>
##   Nb of parameters: 3
##       parameter names:  ka V CL
##       distribution:
##      Parameter Distribution Estimated
## [1,] ka        log-normal   Estimated
```

```
## [2,] V         log-normal   Estimated
## [3,] CL        log-normal   Estimated
##   Variance-covariance matrix:
##    ka V CL
## ka  1 0  0
## V   0 1  1
## CL  0 1  1
##   Error model: proportional , initial values: b.1=1
##   Covariate model:
##        [,1] [,2] [,3]
## Weight    0    0    1
## Sex       0    1    0
##     Initial values
##               ka  V    CL
## Pop.CondInit 1.0 20  0.50
## Cov.CondInit 0.1  0 -0.01
## psi1         0.1  0 -0.01
## ----------------------------------
## ----    Key algorithm options  ----
## ----------------------------------
##     Estimation of individual parameters (MAP)
##     Estimation of standard errors and linearised log-likelihood
##     Estimation of log-likelihood by importance sampling
##     Number of iterations:  K1=300, K2=100
##     Number of chains:  5
##     Seed:  12345
##     Number of MCMC iterations for IS:  5000
##     Simulations:
##         nb of simulated datasets used for npde:  1000
##         nb of simulated datasets used for VPC:  100
##     Input/output
##         save the results to a file:  FALSE
##         save the graphs to files:  FALSE
## ----------------------------------------------------
## ----                  Results                   ----
## ----------------------------------------------------
## -----------------  Fixed effects  ------------------
## ----------------------------------------------------
##      Parameter       Estimate SE     CV(%) p-value
## [1,] ka               1.4864  0.3016  20.3 -
## [2,] V               30.5095  1.9290   6.3 -
## [3,] beta_Sex(V)      0.0816  0.0725  88.9 0.13
## [4,] CL               3.9037  1.7707  45.4 -
## [5,] beta_Weight(CL) -0.0049  0.0064 130.7 0.22
## [6,] b.1              0.1597  0.0122   7.6 -
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##       Parameter Estimate SE    CV(%)
## ka    omega2.ka 0.441    0.193 44
## V     omega2.V  0.017    0.010 61
## CL    omega2.CL 0.063    0.028 44
## covar cov.V.CL  0.033    0.015 47
## ----------------------------------------------------
```

```
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##           omega2.ka omega2.V omega2.CL
## omega2.ka 1         0        0
## omega2.V  0         1        1
## omega2.CL 0         1        1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 333.5813
##       AIC = 353.5813
##       BIC = 358.4304
##
## Likelihood computed by importance sampling
##       -2LL= 349.1338
##       AIC = 369.1338
##       BIC = 373.9829
## ----------------------------------------------------
```

```
theo.fit1<-saemix.fit
theo.fit3<-saemix.fit3
```

Individual parameters and predictions

- predict function
  - **TODO** check why we don't have predictions at the end of the fit ? We should have them by default (from MAP at least for individual predictions and for ypred for population predictions)

```
saemixObject<-theo.fit1

# Conditional distribution
myfit <- conddist.saemix(saemixObject, nsamp = 100)
dim(myfit@results@psi.samp)
```

```
## [1]  12   3 100
```

```
# Predictions for the observations in the original data
## Extract predictions - empty for the moment (?)
vec<-predict(saemixObject)
```

```
## No fitted values of type ipred available
```

```
vec<-predict(saemixObject, type="ypred")
```

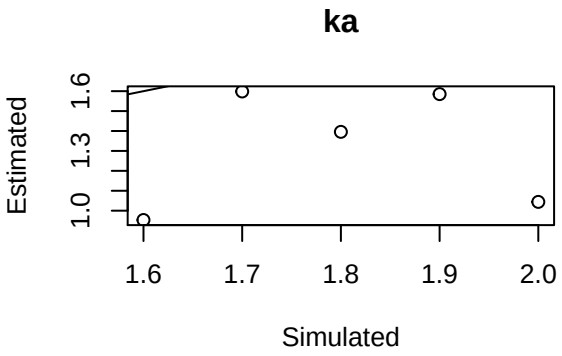
```
## No fitted values of type ypred available
```

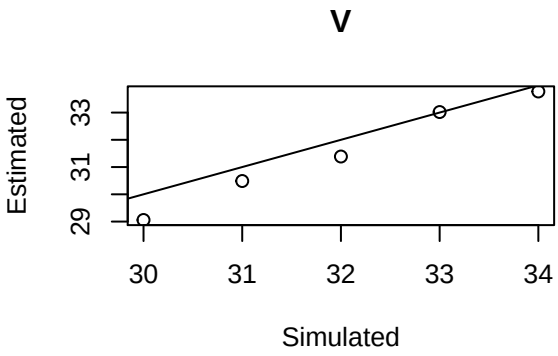
```
# Fit then extract predictions
fit.pred<-saemix.predict(saemixObject)
predict(fit.pred)
```

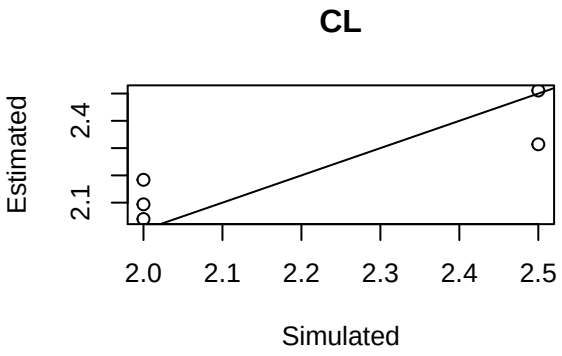
```
##   [1] 3.8361716 6.7770612 9.0453887 9.7915773 9.0824875 8.4180687 7.4909441
##   [8] 6.6284085 5.5037789 2.6209359 3.9937931 6.1363488 8.0051945 8.4432717
##  [15] 7.4193818 6.3821842 5.2194385 4.2852271 3.1735165 0.9263284 4.3341624
##  [22] 6.8074403 8.1469815 8.2836718 7.3107810 6.4548006 5.4445827 4.6160310
##  [29] 3.5257537 1.2610461 3.4391588 5.0831921 6.9762892 8.2944330 7.9620006
##  [36] 7.0972344 5.9864215 5.0341750 3.8936438 1.2962915 4.2090935 6.1973169
```

```
##  [43] 8.5909108 9.7337694 9.0191273 7.9841382 6.7572127 5.6773281 4.4533590
##  [50] 1.5833842 2.0527692 3.7274756 5.5220302 6.4853334 6.2283441 5.4738437
##  [57] 4.4330134 3.4803094 2.5387570 0.7004470 1.5838733 2.8648349 4.7740623
##  [64] 6.5715834 7.0381999 6.5539482 5.6011228 4.6667618 3.5077998 1.1113723
##  [71] 2.5721438 4.4758982 6.3538580 7.5387984 7.0392285 6.1659766 5.0660080
##  [78] 4.2263351 3.1744892 1.0199796 6.9342108 7.9300490 7.8042648 7.1398069
##  [85] 6.1951953 5.3856307 4.4002610 3.7752484 2.9017224 0.8688890 2.9279027
##  [92] 5.1888895 6.2382307 8.6370674 9.3364372 8.9040112 7.9179543 6.8036525
##  [99] 5.6580852 2.5663961 4.8963427 6.9273209 7.9185371 7.4719685 6.3520819
## [106] 5.5006674 4.4868312 3.6635676 2.6784857 0.7969607 2.5840958 4.5539021
## [113] 7.1561695 9.2495965 9.2972432 8.3652775 7.0479362 5.9120712 4.4997449
## [120] 1.5053295
```

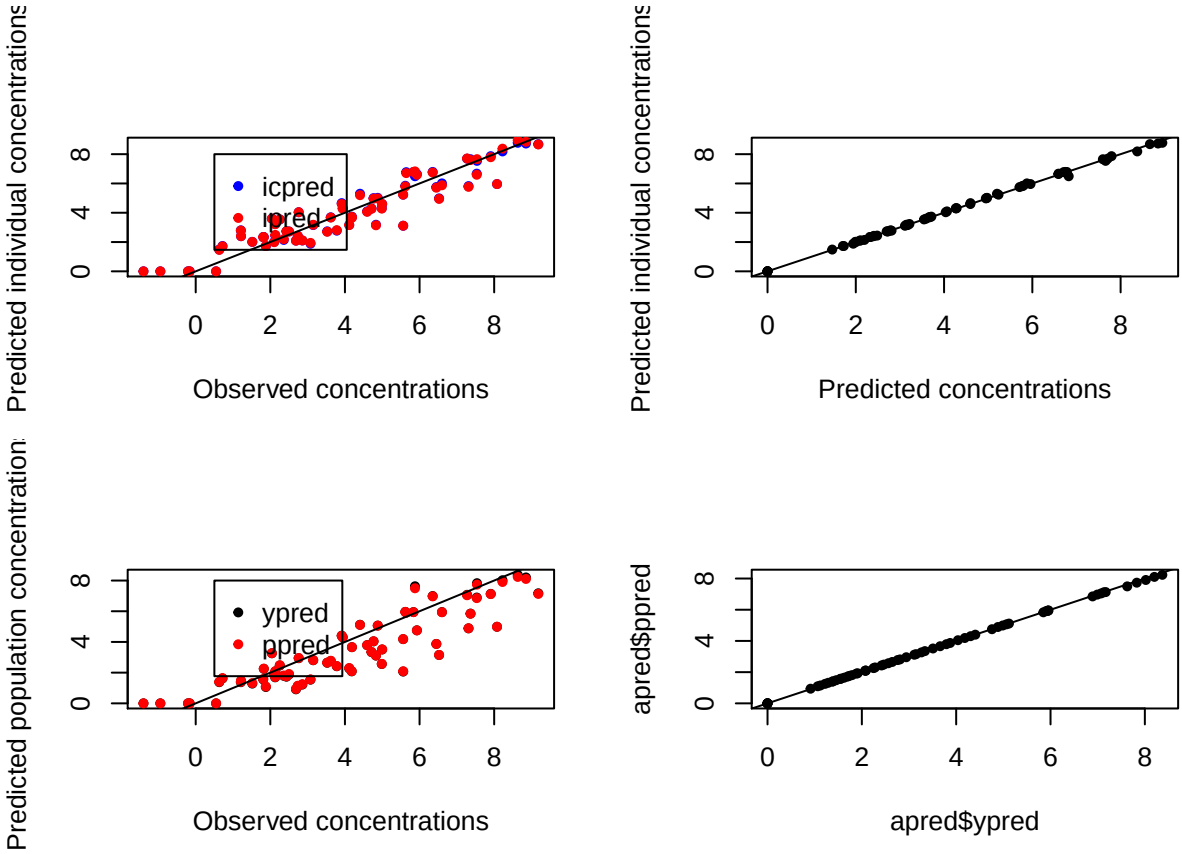
```r
# Predictions for the observations in a new dataset
## Create a new dataset
xtim<-seq(0,24,2)
nsuj<-5
xwei<-seq(50,90,length.out = nsuj)
xsex<-rep(c("F","M"),length.out=nsuj)
xdose<-seq(280,320,length.out=nsuj)
theo.newdata<-data.frame(Id=rep(1:nsuj,each=length(xtim)),Time=rep(xtim,nsuj),Dose=rep(xdose,each=lengt

psiM<-data.frame(ka=seq(1.6,2,0.1),V=seq(34,30),CL=c(2,2.5,2,2.5,2))
fpred<-saemixObject["model"]["model"](psiM, theo.newdata$Id, theo.newdata[,c("Dose","Time")])
theo.newdata$Concentration<-fpred+rnorm(length(fpred),sd=0.74)
theo.psiM<-psiM
test.newdata<-theo.newdata

## Use predict function
mylist<-predict.newdata(saemixObject, theo.newdata, type=c("ipred", "ypred", "ppred", "icpred"))
param<-mylist$param$map.psi
par(mfrow=c(2,2))
for(i in 1:3) {
  plot(theo.psiM[,i],param[,i],main=colnames(psiM)[i],xlab="Simulated",ylab="Estimated")
  abline(0,1)
}

apred<-mylist$predictions
par(mfrow=c(2,2))
```








```
plot(theo.newdata$Concentration,apred$ipred,pch=20,col="Blue", xlab="Observed concentrations", ylab="Pr
points(theo.newdata$Concentration,apred$icpred,pch=20,col="Red", xlab="Observed concentrations", ylab="
abline(0,1)
legend(0.5,8,pch=20,col=c("Blue","Red"),c("icpred","ipred"))

plot(apred$icpred,apred$ipred,pch=20,col="Black", xlab="Predicted concentrations", ylab="Predicted indi
abline(0,1)

plot(theo.newdata$Concentration,apred$ypred,pch=20,col="Black", xlab="Observed concentrations", ylab="P
points(theo.newdata$Concentration,apred$ppred,pch=20,col="Red", xlab="Observed concentrations", ylab="P
abline(0,1)
legend(0.5,8,pch=20,col=c("Black","Red"),c("ypred","ppred"))

plot(apred$ypred,apred$ppred,pch=20,col="Black")
abline(0,1)
```

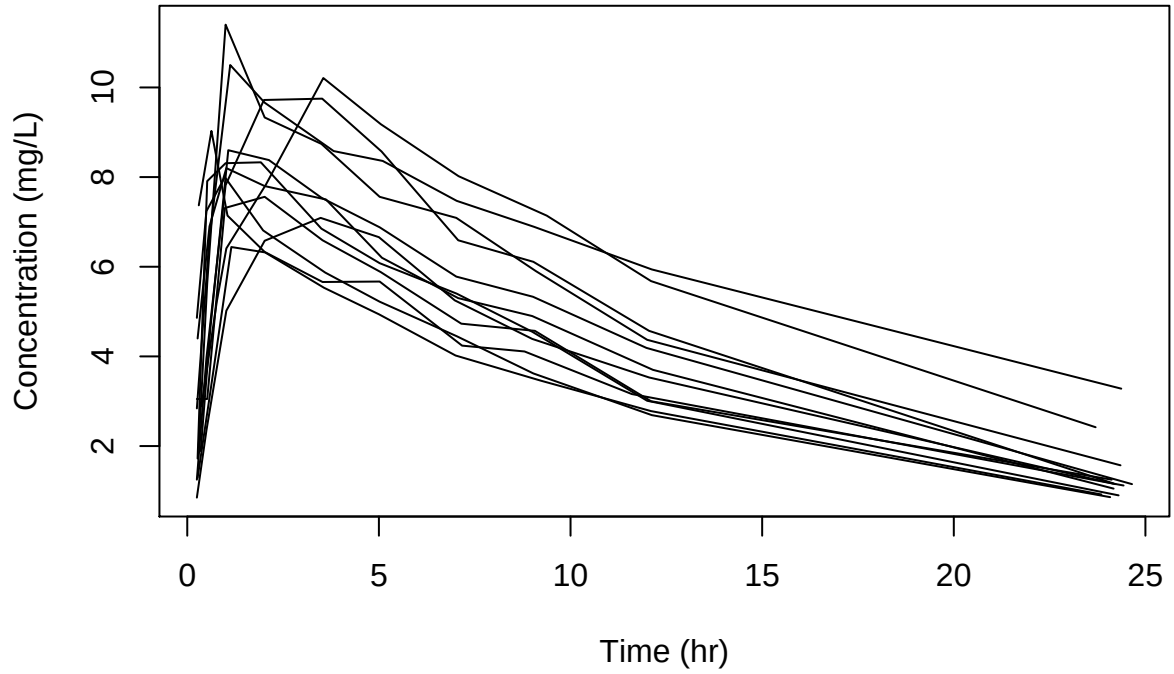


Plots

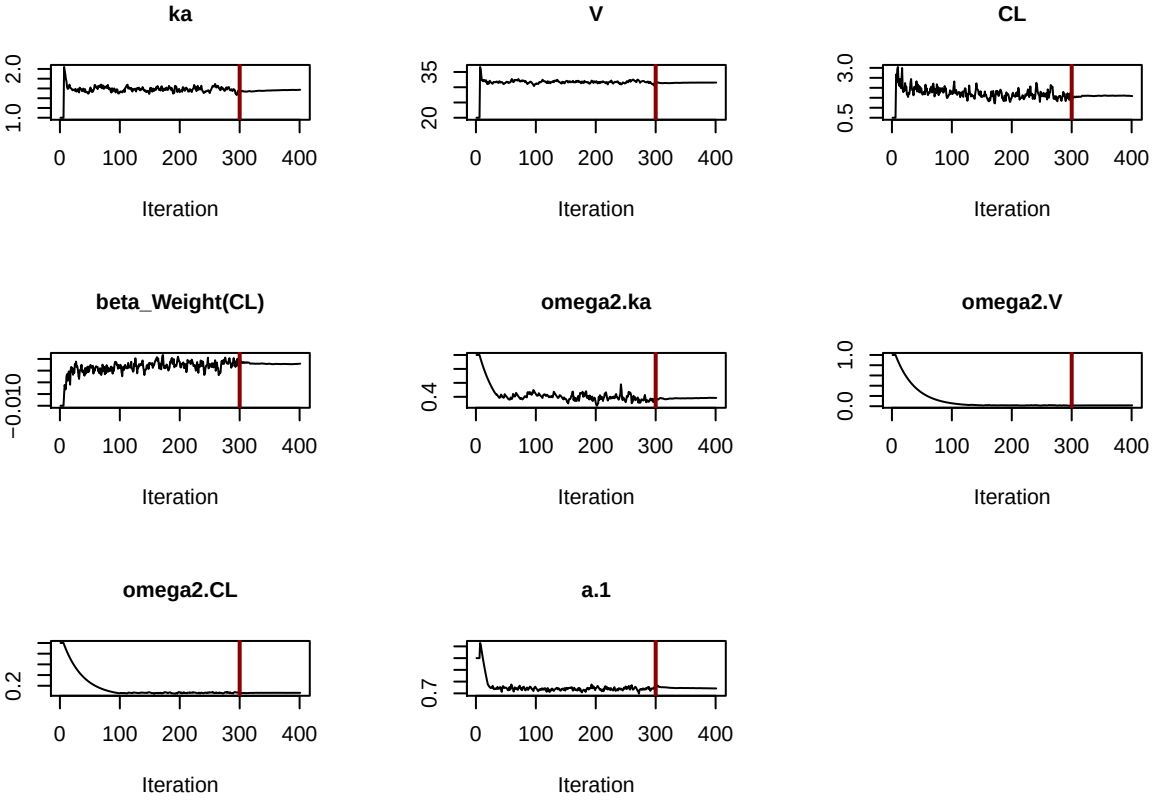
- **bug**
  - individual fit doesn't work (optim(par = phi1, fn = conditional.distribution_c, phii = phii, la fonction ne peut être évaluée aux paramètres initiaux) for theo.fit3 (check why)
  - covariate plots not working ("The following plot types were not found or are ambiguous: rand-eff.versus.covariates, parameters.versus.covariates")
- **TODO**
  - include new npde plots
  - include mirror plot
  - include diagnostic plots with samples from the conditional distribution (next version 3.1 ?)

```
myfit<-theo.fit1
# Generic plots
plot(myfit)
```

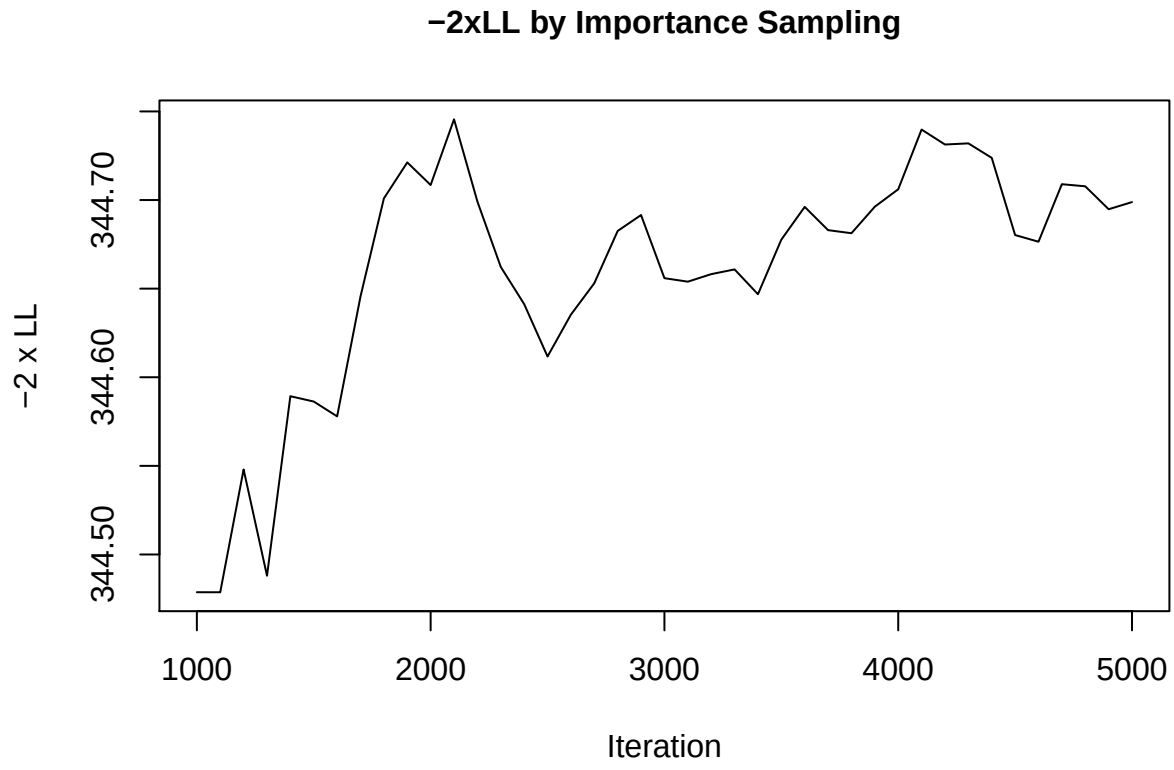
```
## Plotting the data
```



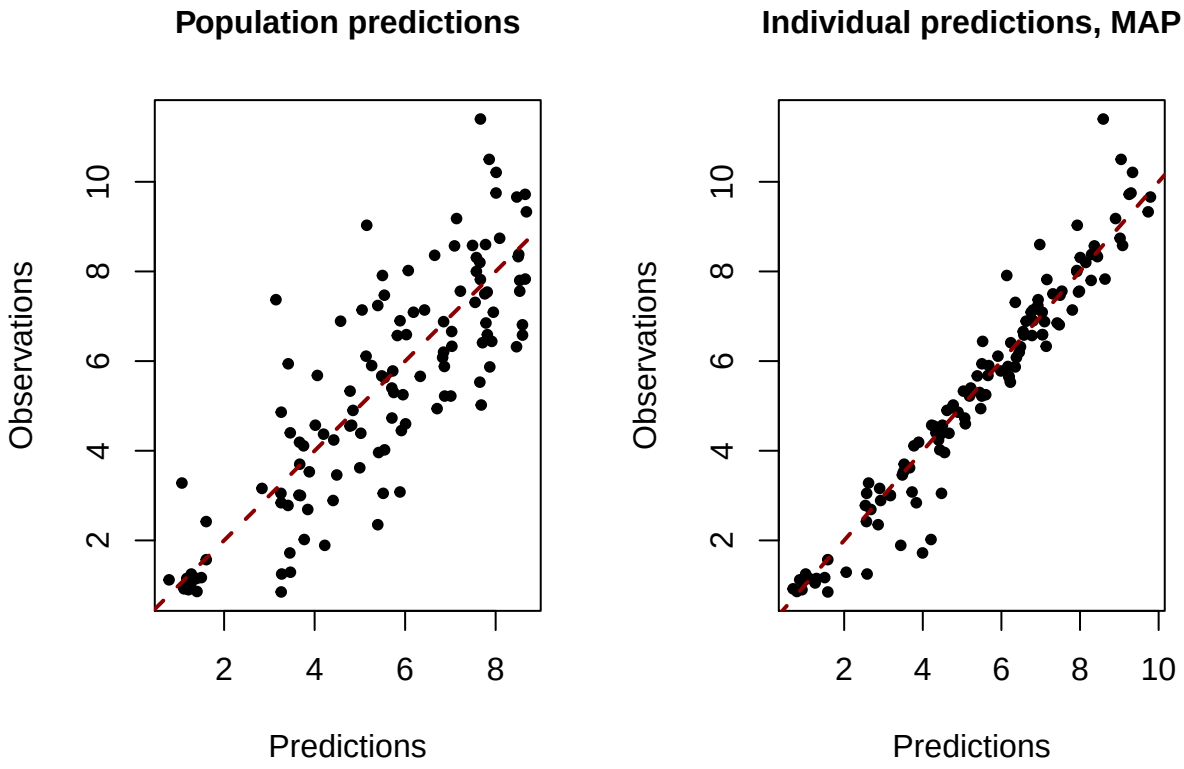

```
## Plotting convergence plots
```




```
## Plotting the likelihood
```

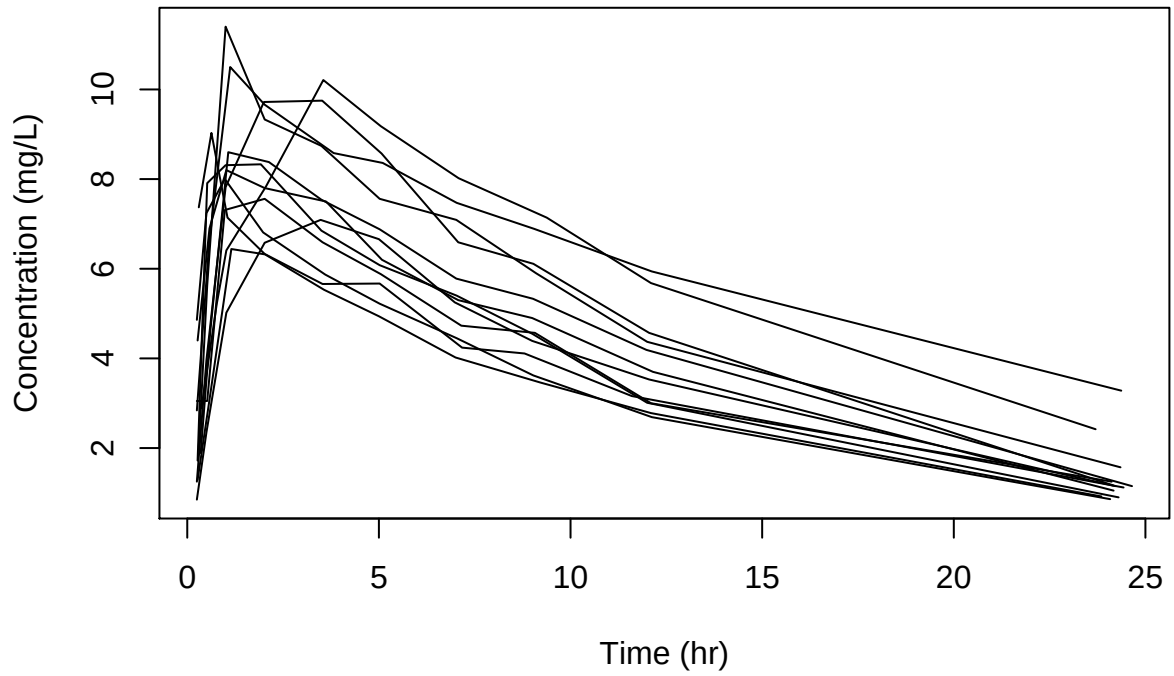



```
## Plotting observations versus predictions
```



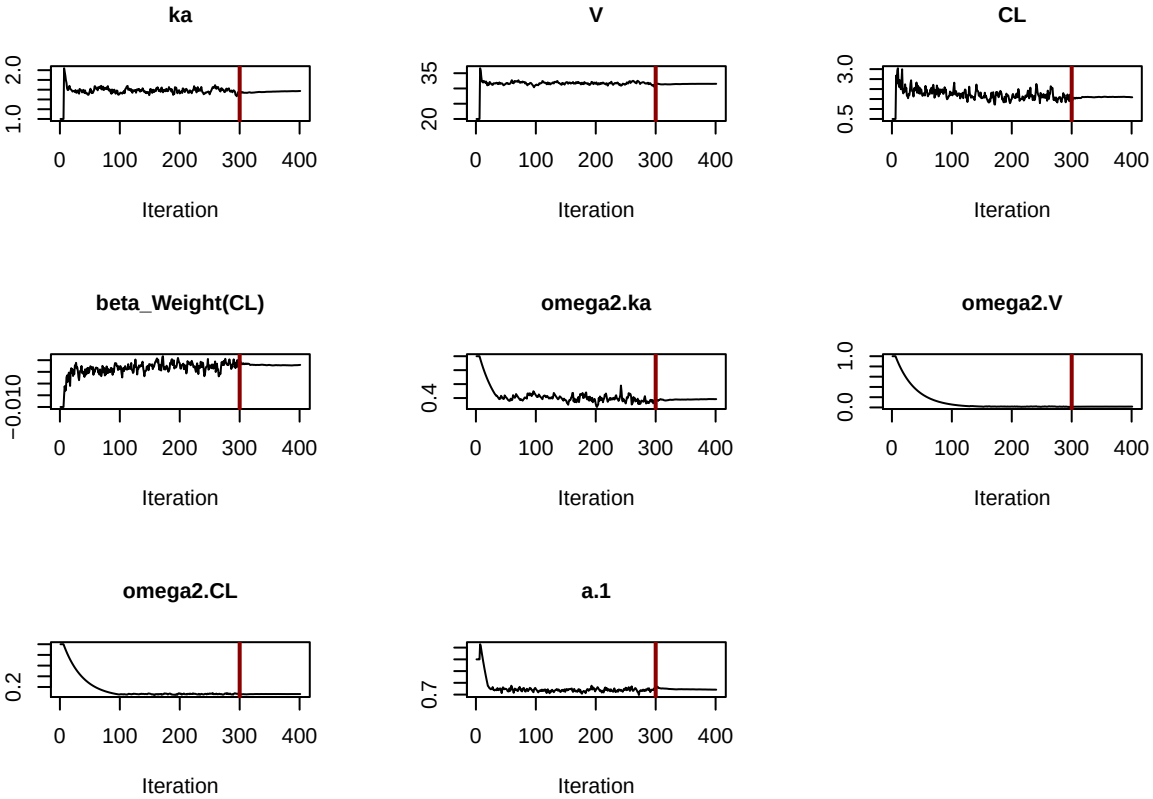

```
# Individual plots
plot(myfit, plot.type="data")
```

```
## Plotting the data
```



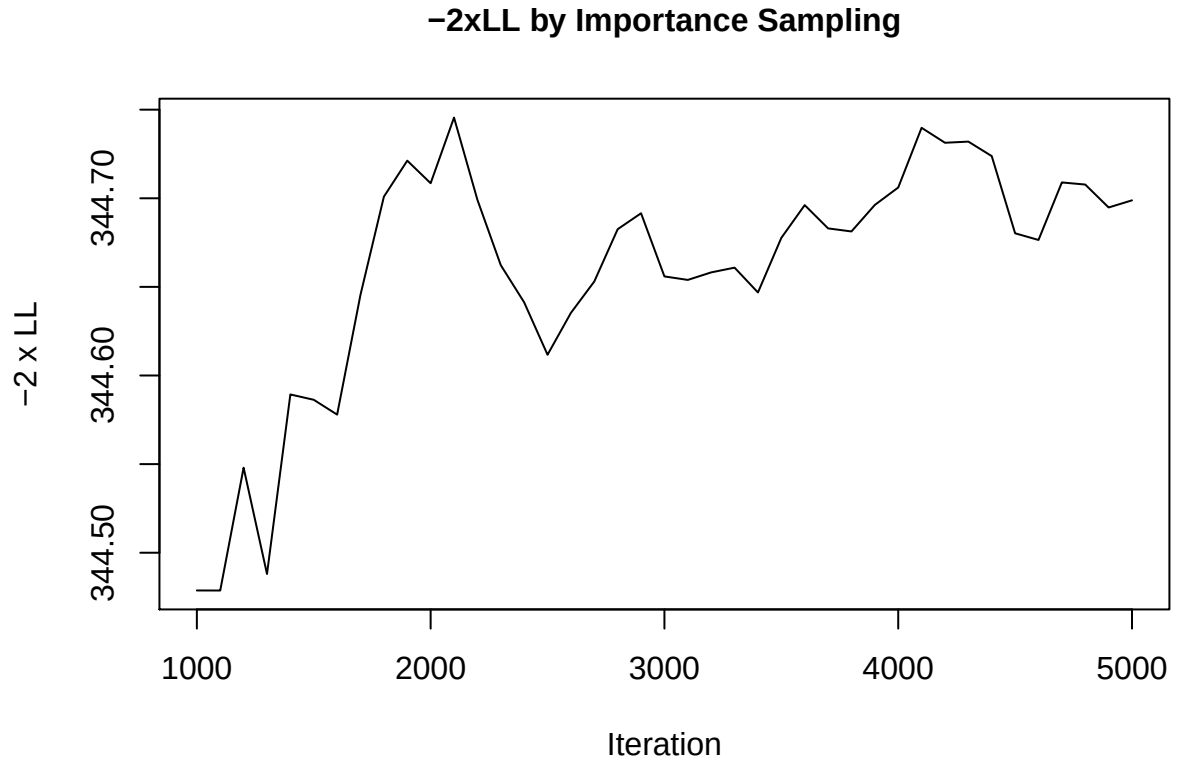

```
plot(myfit, plot.type="convergence")
```

```
## Plotting convergence plots
```



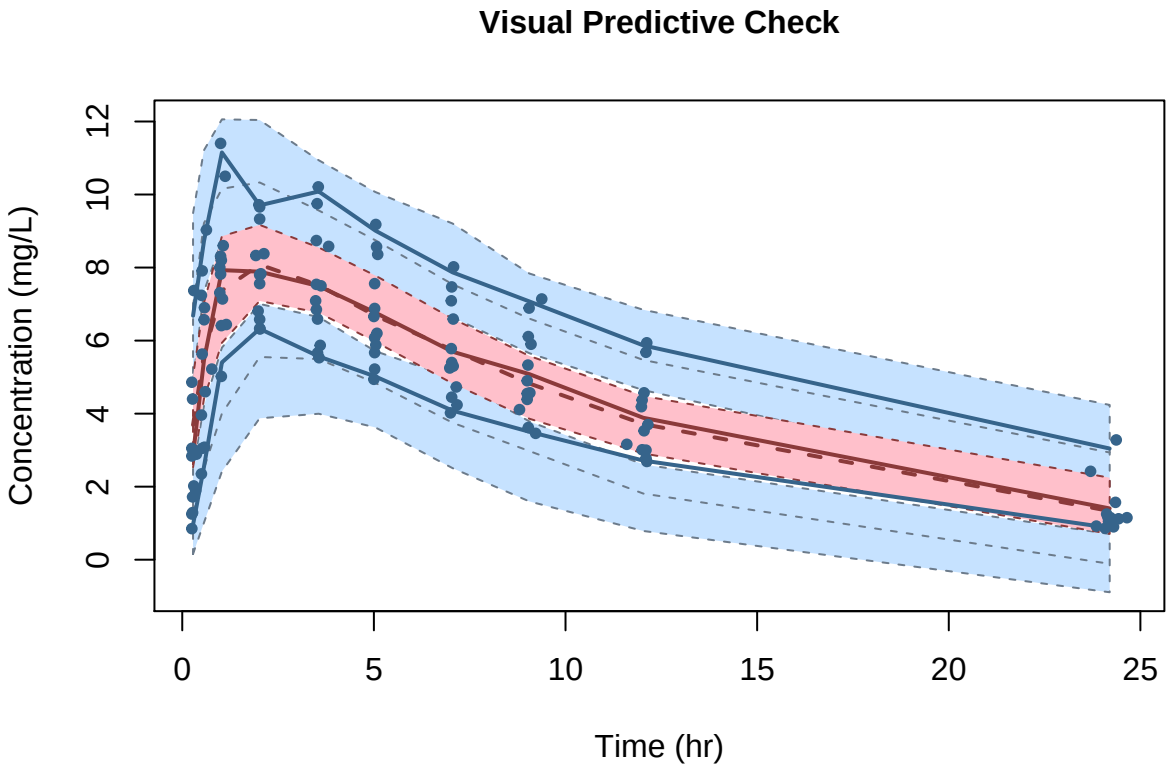
```
plot(myfit, plot.type="likelihood")
```

```
## Plotting the likelihood
```



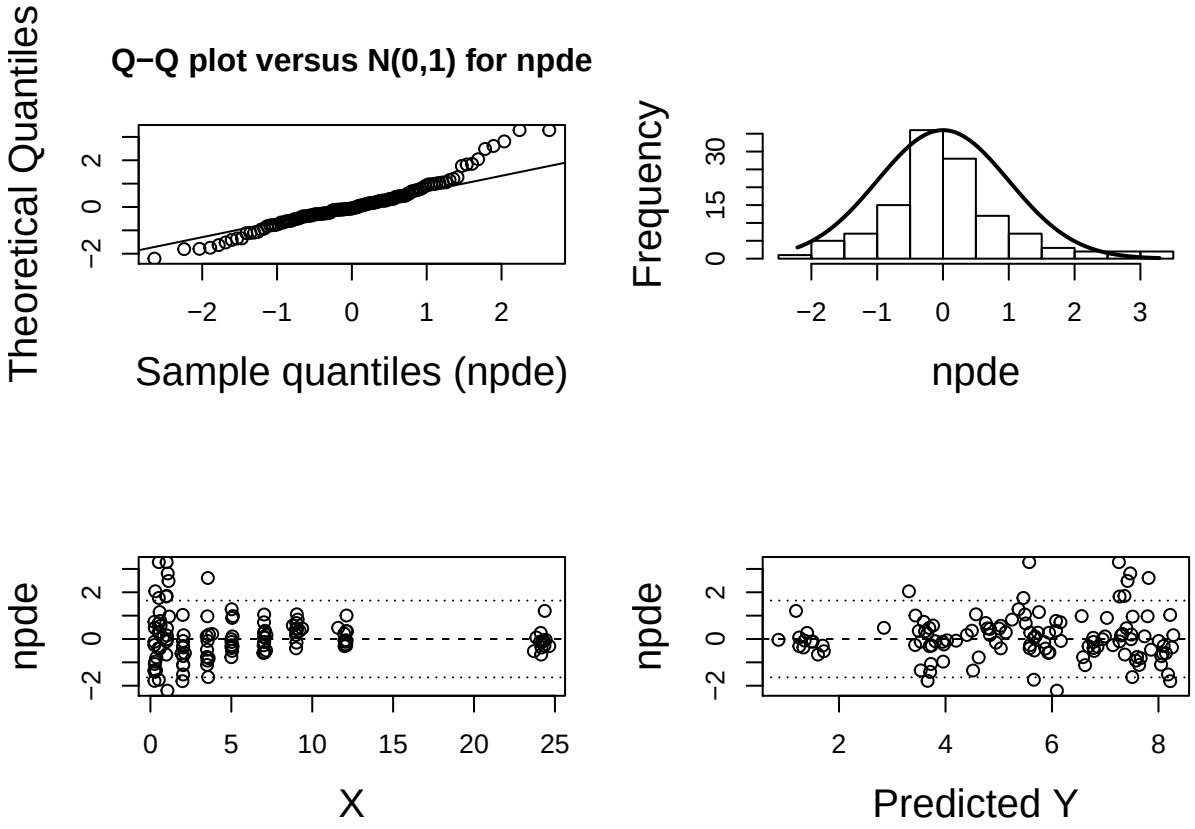

```
plot(myfit, plot.type="vpc")
```

```
## Performing simulations under the model.
## Plotting VPC
```



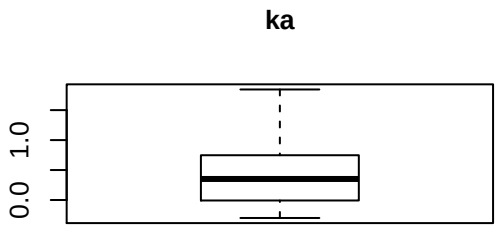

```
plot(myfit, plot.type="npde")
```

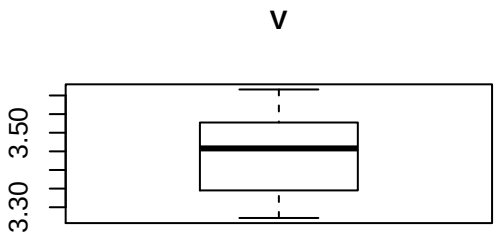
```
## Computing WRES and npde ..
## Plotting npde
```

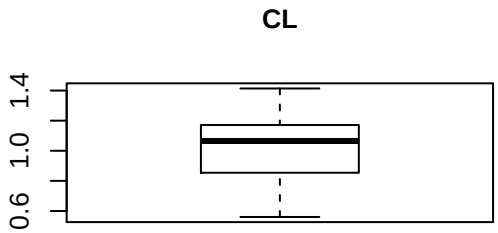



```
## ------------------------------------------------
## Distribution of npde:
##             mean= 0.07246   (SE= 0.09 )
##         variance= 0.9619    (SE= 0.12 )
##         skewness= 0.8011
##         kurtosis= 1.633
## ------------------------------------------------
##
## Statistical tests
##   Wilcoxon signed rank test  : 0.922
##   Fisher variance test       : 0.799
##   SW test of normality       : 0.000159 ***
## Global adjusted p-value      : 0.000476 ***
## ---
## Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
## ------------------------------------------------
```

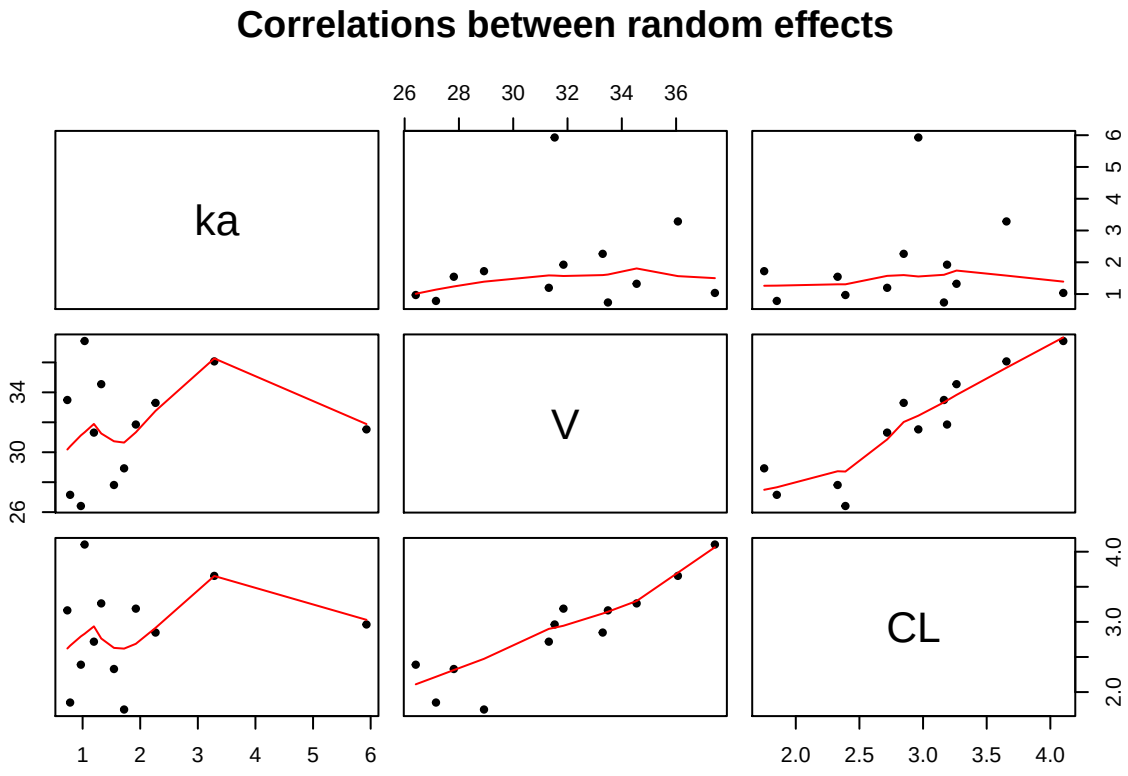
```
plot(myfit, plot.type="random.effects")
```



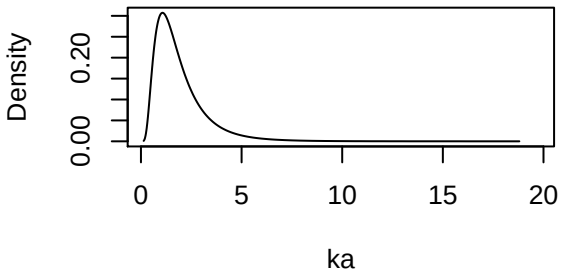


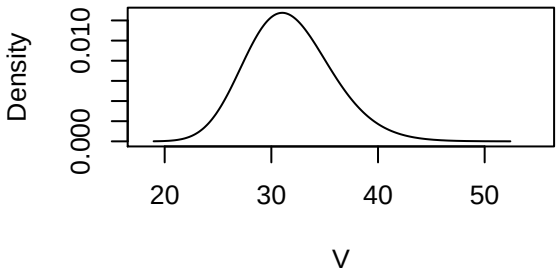


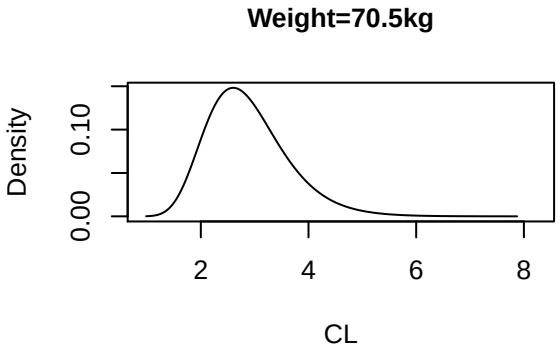

```
plot(myfit, plot.type="correlations")
```




```
plot(myfit, plot.type="marginal.distribution")
```

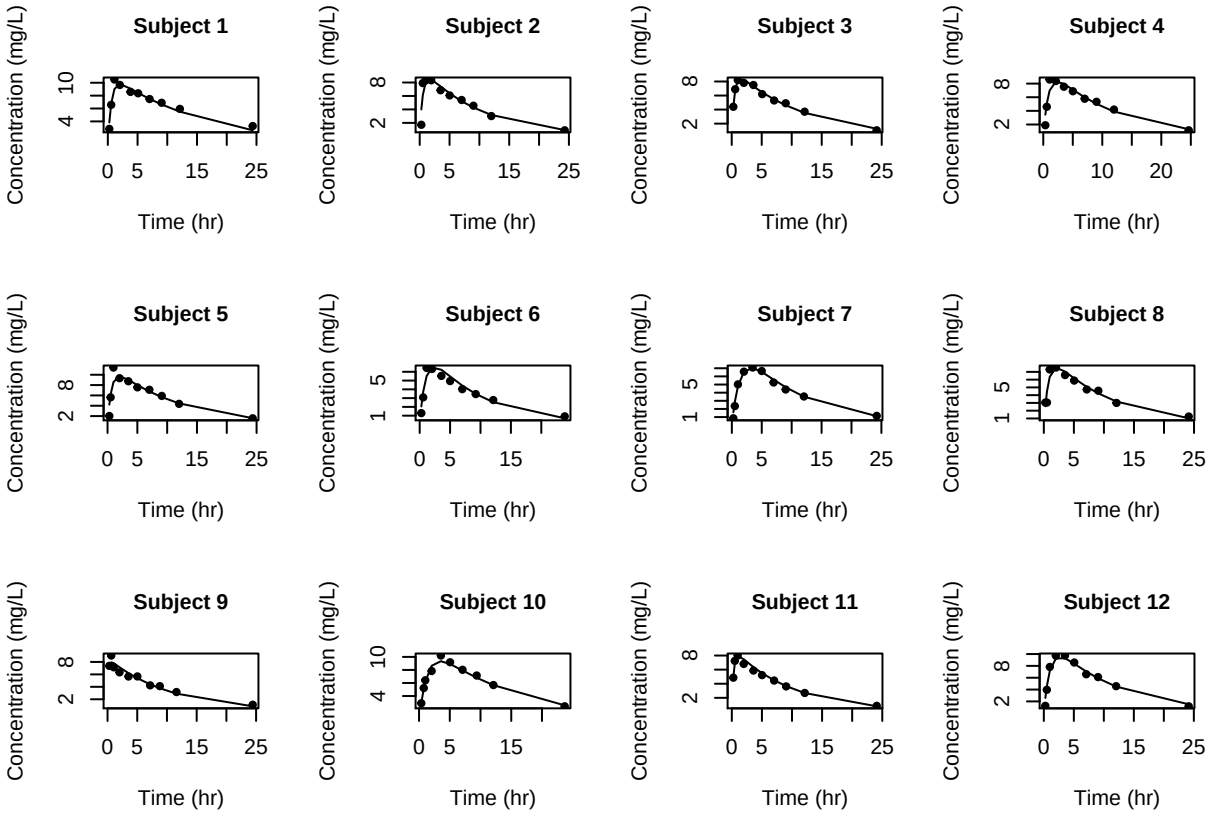






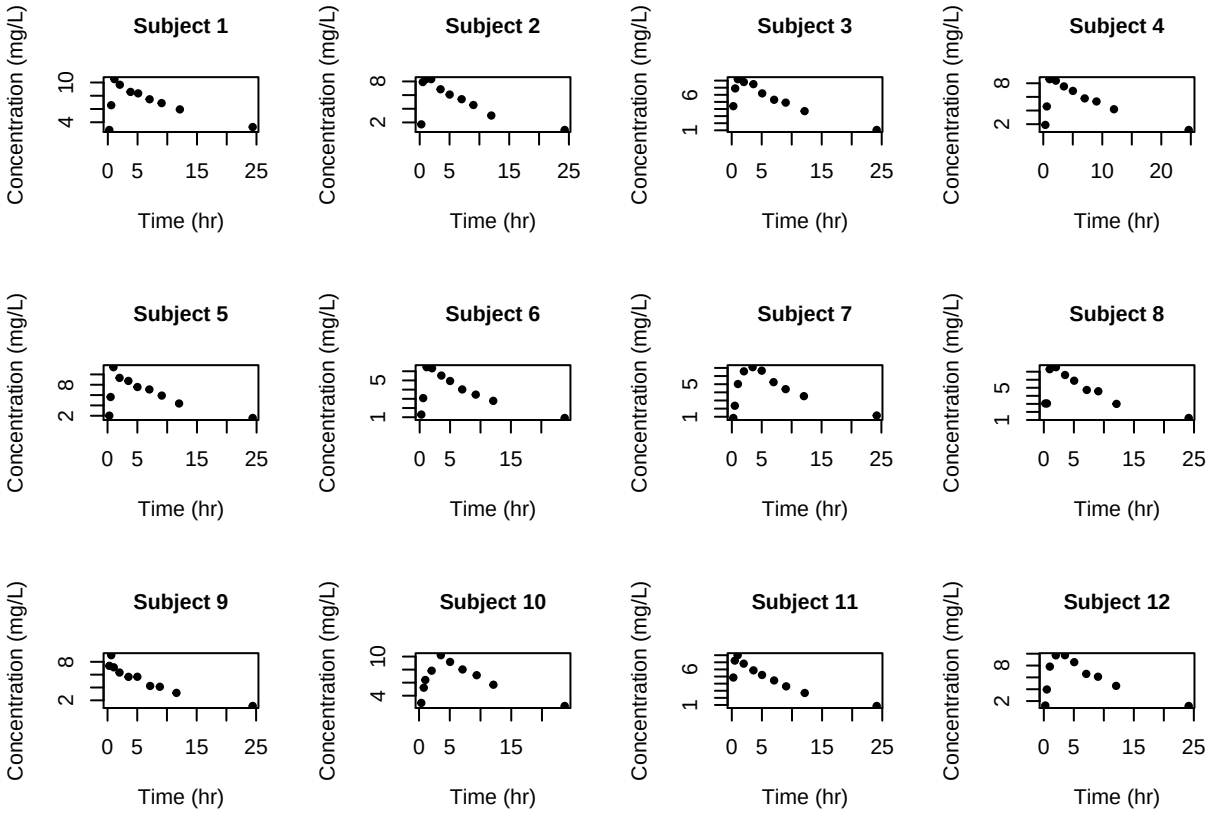

```
try(plot(myfit, plot.type="individual.fit"))
```

```
## Plotting individual fits
```



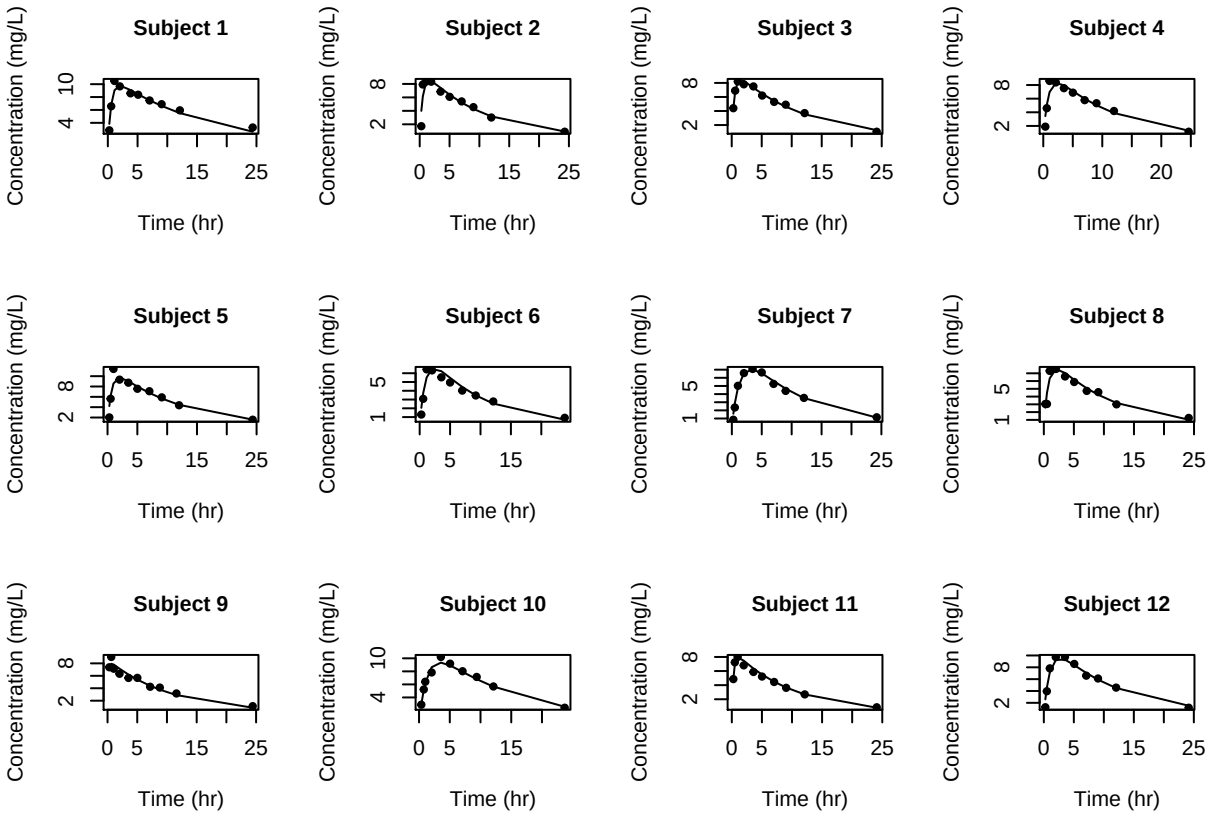
```
plot(myfit, plot.type="population.fit")
```

```
## Plotting fits obtained with population predictions
```



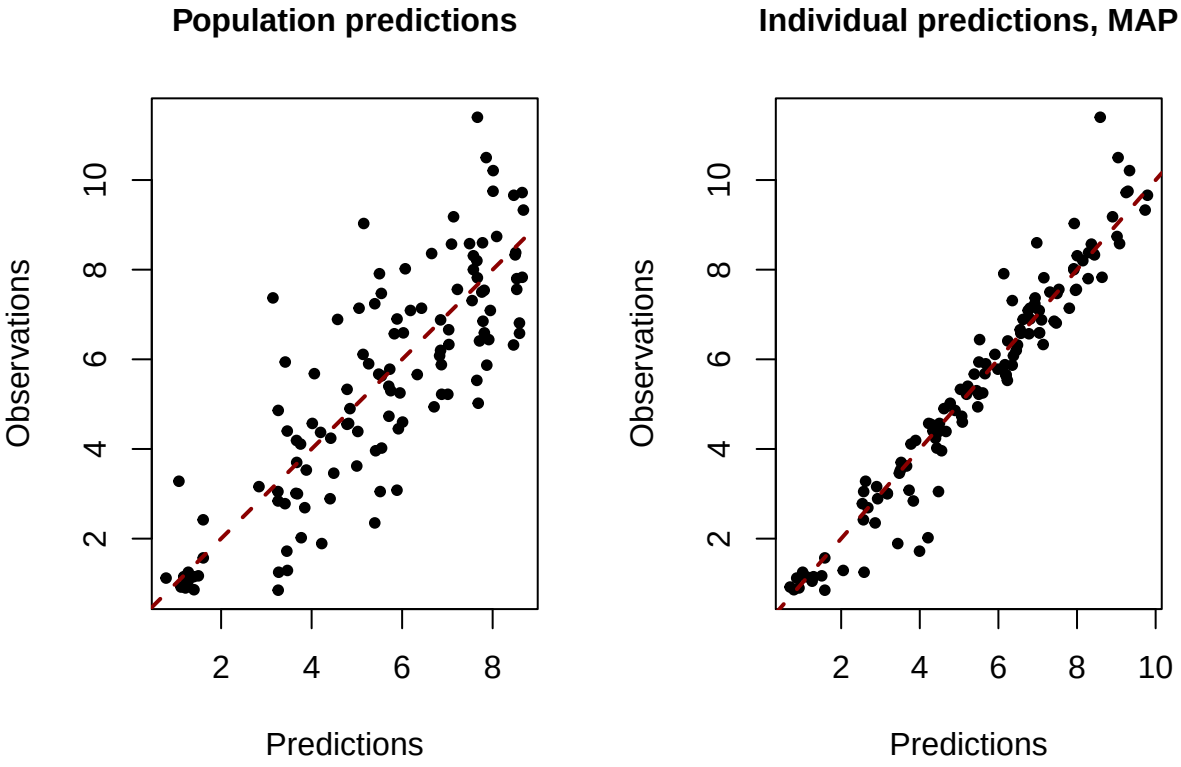

```
try(plot(myfit, plot.type="both.fit"))
```

```
## Plotting the fits overlaying individual and population predictions
```




```
try(plot(myfit, plot.type="observations.vs.predictions"))
```

```
## Plotting observations versus predictions
```



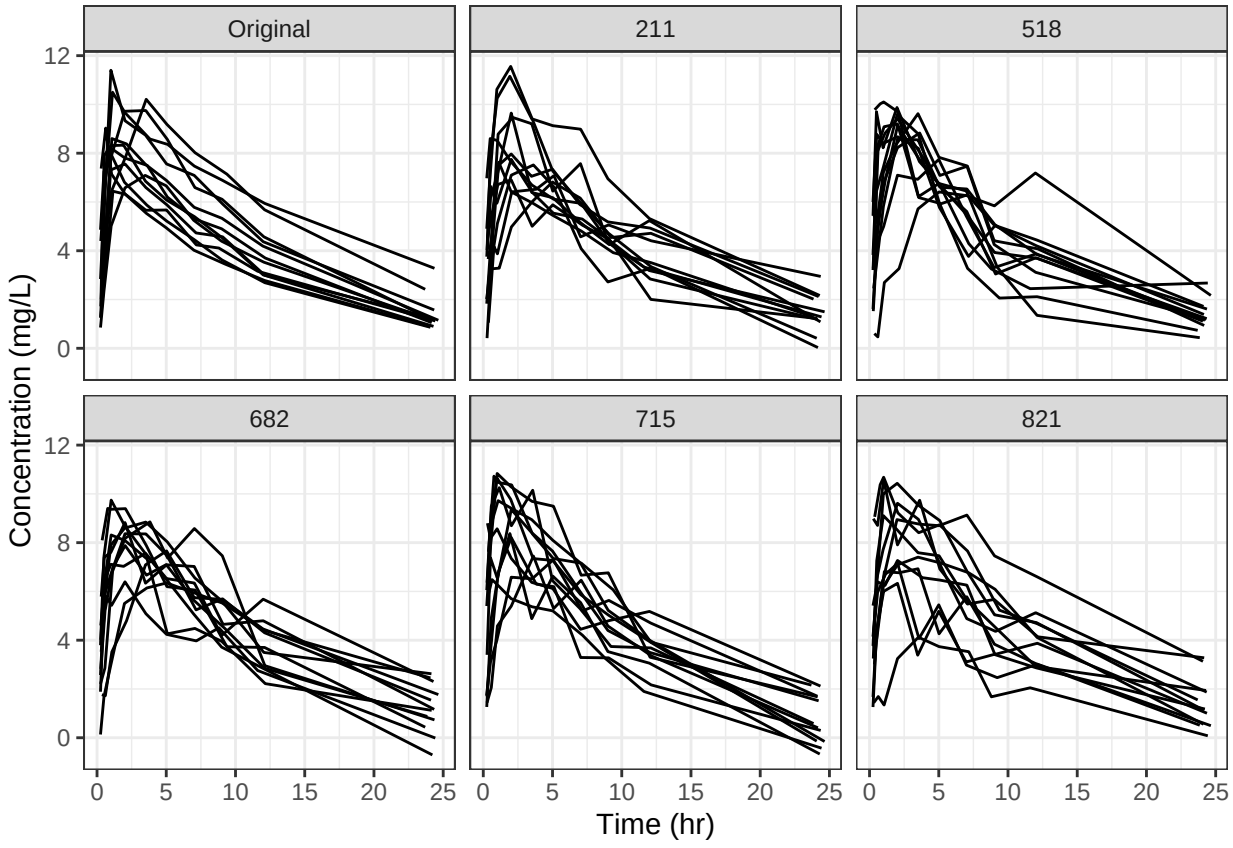
```
plot(myfit, plot.type="parameters.versus.covariates")
```

```
## The following plot types were not found or are ambiguous: parameters.versus.covariates
## NULL
```

```
plot(myfit, plot.type="randeff.versus.covariates")
```

```
## The following plot types were not found or are ambiguous: randeff.versus.covariates
## NULL
```

```
# Simulations
mysim<-saemix.simul(myfit)

## Mirror plot
nmir<-5
isamp<-sample(1:mysim@sim.data@nsim, nmir, replace=FALSE)
datsim<-mysim@sim.data@datasim[mysim@sim.data@datasim$irep %in% isamp, ]
gdat<-cbind(mysim@data@data[,c(mysim@data@name.group, mysim@data@name.X, mysim@data@name.response)], da
gdat1<-data.frame(id=datsim[,c("idsim")], x=rep(gdat[,2],nmir),y=datsim[,"ysim"], data=as.character(dat
colnames(gdat)<-colnames(gdat1)
gdat<-rbind(gdat, gdat1)

ggplot(gdat, aes(x=x, y=y, group=id)) + geom_line() + facet_wrap(.~data, nrow=2, ncol=3) + theme_bw() +
```



Binary response

- **TODO**
  - error message “le nombre d’objets à remplacer n’est pas multiple de la taille du remplacement”

```
nsuj<-1000
xtim<-c(0:3)
parnam<-c("Intercept","beta.time")
param<-c(0,-0.37)
omega<-c(.21,.1)

partab<-as.data.frame(matrix(data=0,nrow=nsuj,ncol=2,dimnames=list(NULL,parnam)))
for(i in 1:2) partab[,i]<-rnorm(nsuj,mean=param[i],sd=omega[i])

psim<-data.frame()
for(itim in xtim) {
  logit.sim<-partab[,1]+partab[,2]*itim
  xtab<-exp(logit.sim)/(1+exp(logit.sim))
  psim<-rbind(psim,xtab)
}
datsim<-data.frame(id=rep(1:nsuj,each=length(xtim)),time=rep(xtim,nsuj),psim=unlist(psim))
rownames(datsim)<-NULL
ysim<-rbinom(nsuj*length(xtim),size=1,prob=datsim$psim)
summary(datsim)
```

```
##        id              time           psim
##  Min.   :   1.0   Min.   :0.00   Min.   :0.09165
##  1st Qu.: 250.8   1st Qu.:0.75   1st Qu.:0.28748
##  Median : 500.5   Median :1.50   Median :0.37424
##  Mean   : 500.5   Mean   :1.50   Mean   :0.37226
##  3rd Qu.: 750.2   3rd Qu.:2.25   3rd Qu.:0.45753
##  Max.   :1000.0   Max.   :3.00   Max.   :0.68242
```

```
datsim$y<-ysim
datsim$risk<-ifelse(datsim$id>500,1,0)

# Running saemix
saemix.data<-saemixData(name.data=datsim,
      name.group=c("id"),name.predictors=c("time","y"), name.covariates=c("risk"),name.X=c("time"))
```

```
## Column name(s)  do(es) not exist in the dataset, please check
## Remove columns 1 (  )
## No valid name given, attempting automatic recognition
## Automatic recognition of columns y successful
## [1] "risk"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset datsim
##     Structured data: y ~ time + y | id
##     X variable for graphs: time ()
##     covariates: risk (-)
```

```
##       reference class for covariate risk :  0
```

```
binary.model<-function(psi,id,xidep) {
  tim<-xidep[,1]
  y<-xidep[,2]
  inter<-psi[id,1]
  slope<-psi[id,2]
  logit<-inter+slope*tim
  pevent<-exp(logit)/(1+exp(logit))
  logpdf<-rep(0,length(tim))
  P.obs = (y==0)*(1-pevent)+(y==1)*pevent
  logpdf <- log(P.obs)
  return(logpdf)
}

saemix.model<-saemixModel(model=binary.model,description="Binary model",
        modeltype="likelihood",
        psi0=matrix(c(0,-.5,0.5,0),ncol=2,byrow=TRUE,dimnames=list(NULL,parnam[1:2])),
        transform.par=c(0,0),covariance.model=matrix(c(1,0,0,1),ncol=2))
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Binary model  Model type:  likelihood
## function(psi,id,xidep) {
##   tim<-xidep[,1]
##   y<-xidep[,2]
##   inter<-psi[id,1]
##   slope<-psi[id,2]
##   logit<-inter+slope*tim
##   pevent<-exp(logit)/(1+exp(logit))
##   logpdf<-rep(0,length(tim))
##   P.obs = (y==0)*(1-pevent)+(y==1)*pevent
##   logpdf <- log(P.obs)
##   return(logpdf)
## }
##   Nb of parameters: 2
##       parameter names:  Intercept beta.time
##       distribution:
##      Parameter Distribution Estimated
## [1,] Intercept normal       Estimated
## [2,] beta.time normal       Estimated
##   Variance-covariance matrix:
##           Intercept beta.time
## Intercept         1         0
## beta.time         0         1
##     No covariate in the model.
##     Initial values
##              Intercept beta.time
## Pop.CondInit       0.0      -0.5
## Cov.CondInit       0.5       0.0
```

```r
saemix.options<-list(seed=632545,save=FALSE,save.graphs=FALSE, displayProgress=FALSE)
# saemix.fit<-saemix(saemix.model,saemix.data,saemix.options)

binary.fit<-saemix(saemix.model,saemix.data,saemix.options)
```

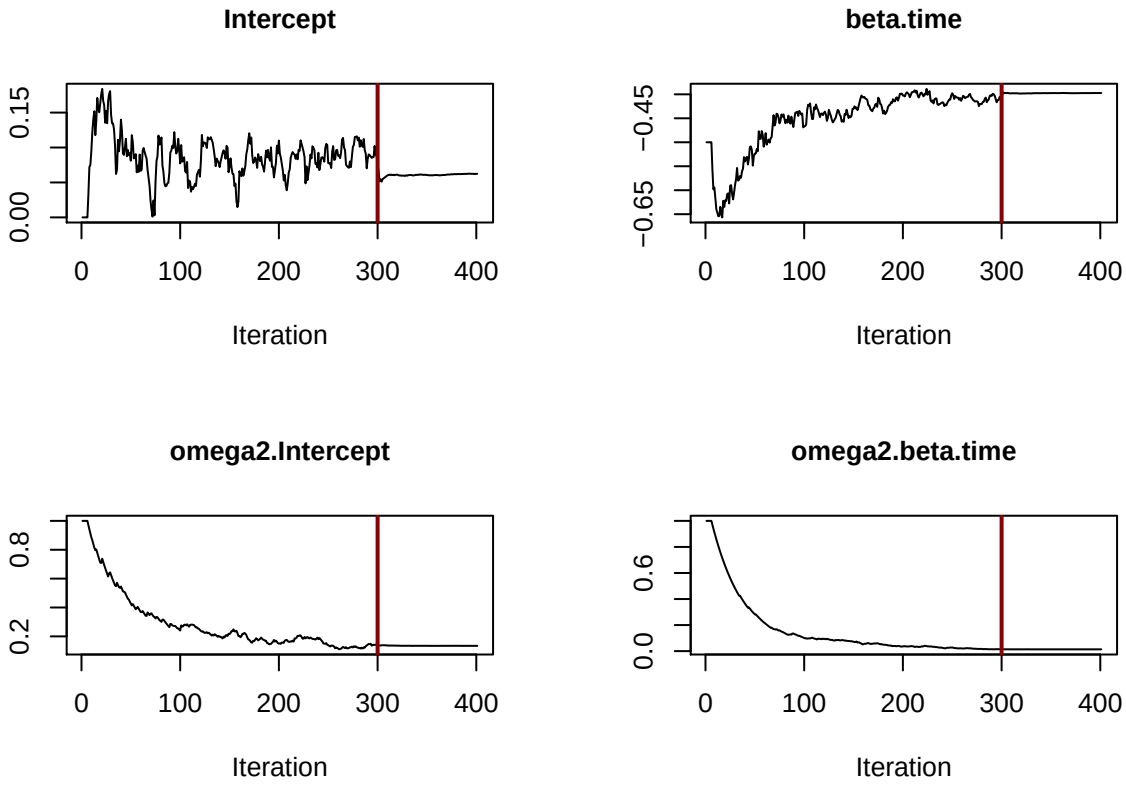
```
## Error in solve.default(FO) :
##   routine Lapack dgesv : le système est exactement singulier : U[2,2] = 0
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data             ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset datsim
##     Structured data: y ~ time + y | id
##     X variable for graphs: time ()
##     covariates: risk (-)
##       reference class for covariate risk :  0
## Dataset characteristics:
##     number of subjects:     1000
##     number of observations: 4000
##     average/min/max nb obs: 4.00  /  4  /  4
## First 10 lines of data:
##    id time y y.1 risk mdv cens occ ytype
## 1   1    0 1   1    0   0    0   1     1
## 2   1    1 0   0    0   0    0   1     1
## 3   1    2 0   0    0   0    0   1     1
## 4   1    3 0   0    0   0    0   1     1
## 5   2    0 1   1    0   0    0   1     1
## 6   2    1 0   0    0   0    0   1     1
## 7   2    2 1   1    0   0    0   1     1
## 8   2    3 0   0    0   0    0   1     1
## 9   3    0 0   0    0   0    0   1     1
## 10  3    1 1   1    0   0    0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  Binary model  Model type:  likelihood
## function(psi,id,xidep) {
##   tim<-xidep[,1]
##   y<-xidep[,2]
##   inter<-psi[id,1]
##   slope<-psi[id,2]
##   logit<-inter+slope*tim
##   pevent<-exp(logit)/(1+exp(logit))
##   logpdf<-rep(0,length(tim))
##   P.obs = (y==0)*(1-pevent)+(y==1)*pevent
##   logpdf <- log(P.obs)
##   return(logpdf)
## }
## <bytecode: 0x56264b1864f0>
##   Nb of parameters: 2
##       parameter names:  Intercept beta.time
```

```
##       distribution:
##      Parameter Distribution Estimated
## [1,] Intercept normal       Estimated
## [2,] beta.time normal       Estimated
##   Variance-covariance matrix:
##           Intercept beta.time
## Intercept         1         0
## beta.time         0         1
##     No covariate in the model.
##     Initial values
##              Intercept beta.time
## Pop.CondInit         0      -0.5
## ----------------------------------
## ----    Key algorithm options  ----
## ----------------------------------
##     Estimation of individual parameters (MAP)
##     Estimation of standard errors and linearised log-likelihood
##     Estimation of log-likelihood by importance sampling
##     Number of iterations:  K1=300, K2=100
##     Number of chains:  1
##     Seed:  632545
##     Number of MCMC iterations for IS:  5000
##     Simulations:
##         nb of simulated datasets used for npde:  1000
##         nb of simulated datasets used for VPC:  100
##     Input/output
##         save the results to a file:  FALSE
##         save the graphs to files:  FALSE
## ----------------------------------------------------
## ----                  Results                   ----
## ----------------------------------------------------
## -----------------  Fixed effects  ------------------
## ----------------------------------------------------
##      Parameter Estimate SE    CV(%)
## [1,] Intercept  0.062   0.064 103
## [2,] beta.time -0.397   0.048  12
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##           Parameter        Estimate SE CV(%)
## Intercept omega2.Intercept 0.134    NA NA
## beta.time omega2.beta.time 0.014    NA NA
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##                  omega2.Intercept omega2.beta.time
## omega2.Intercept 1                0
## omega2.beta.time 0                1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 12563.29
##       AIC = 12573.29
```

```
##       BIC = 12597.83
##
## Likelihood computed by importance sampling
##       -2LL= 5175.245
##       AIC = 5185.245
##       BIC = 5209.783
## ----------------------------------------------------
```

```
plot(binary.fit, plot.type="convergence")
```

```
## Plotting convergence plots
```



Ordinal data

- **TODO**
  - check results compared to previous version
  - check LL by GQ to compare to LL by IS
  - test the optimisation and change the algorithm for one-dimension

```
smx.ord <- read.table(file.path(datDir,"categorical1_data.txt"),header=T)
saemix.data<-saemixData(name.data=smx.ord, header=TRUE, sep=" ", na=NA,
                        name.group=c("ID"), name.predictors=c("Y"), name.X=c("TIME"))
```

```
## Column name(s)  do(es) not exist in the dataset, please check
## Remove columns 1 (  )
## No valid name given, attempting automatic recognition
## Automatic recognition of columns Y successful
## Attribute name.X TIME does not correspond to a valid column in the dataset, setting the X axis for g
##
##
```

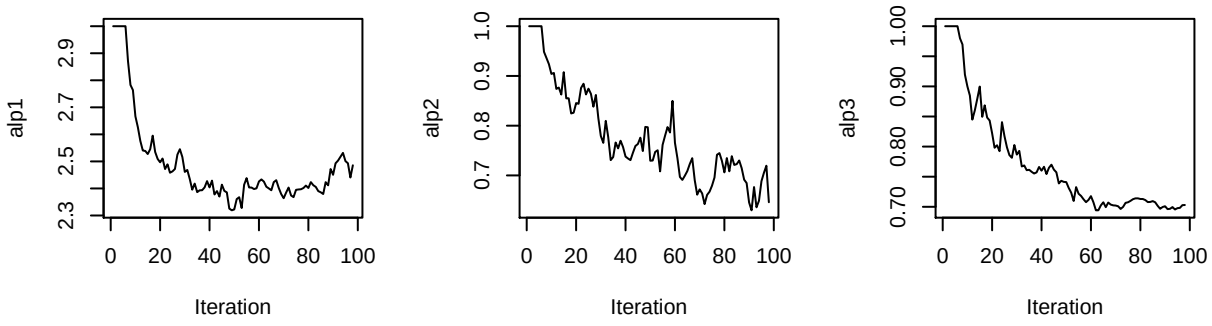
```
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset smx.ord
##     Structured data: Y ~ Y | ID
##     Predictor: Y ()
```

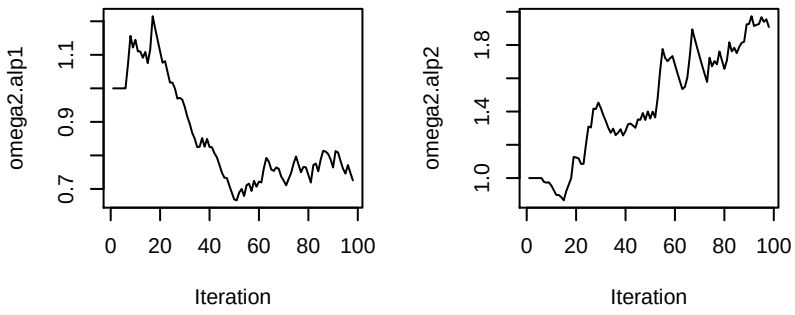
```
ordinal.model<-function(psi,id,xidep) {
  y<-xidep[,1]
  alp1<-psi[id,1]
  alp2<-psi[id,2]
  alp3<-psi[id,3]
  logit1<-alp1
  logit2<-alp1+alp2
  logit3<-alp1+alp2+alp3
  pge1<-exp(logit1)/(1+exp(logit1))
  pge2<-exp(logit2)/(1+exp(logit2))
  pge3<-exp(logit3)/(1+exp(logit3))
  logpdf<-rep(0,length(y))
  P.obs = (y==0)*pge1+(y==1)*(pge2 - pge1)+(y==2)*(pge3 - pge2)+(y==3)*(1 - pge3)
  logpdf <- log(P.obs)

  return(logpdf)
}

saemix.model<-saemixModel(model=ordinal.model,description="Ordinal categorical model",modeltype="likeli
                          psi0=matrix(c(3,1,1),ncol=3,byrow=TRUE,dimnames=list(NULL,c("alp1","alp2","al
                          omega.init=matrix(c(1,0,0,0,1,0,0,0,1),ncol=3,byrow=TRUE),
                          transform.par=c(0,1,1),covariance.model=matrix(c(1,0,0,0,1,0,0,0,0),ncol=3))
```

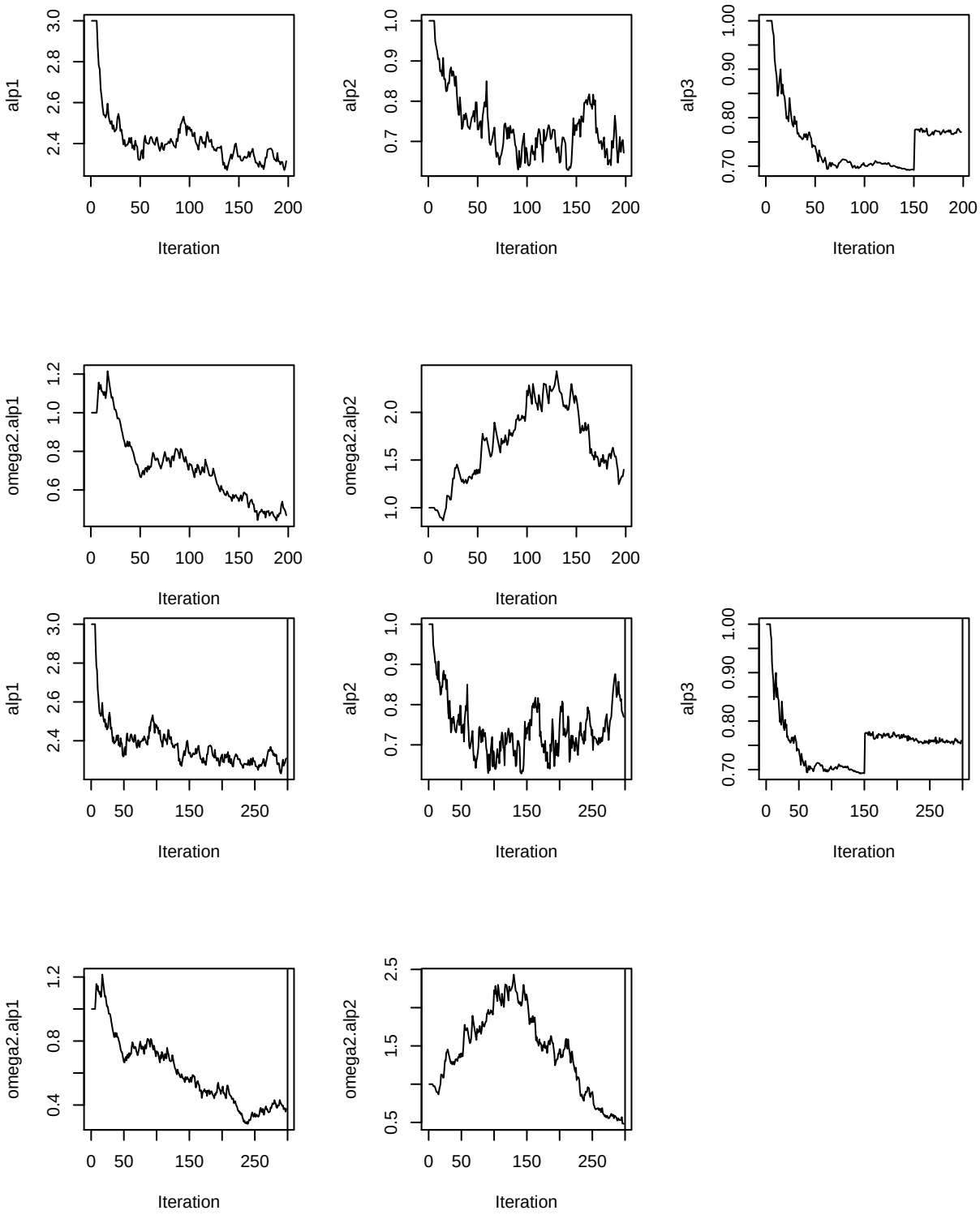
```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Ordinal categorical model  Model type:  likelihood
## function(psi,id,xidep) {
##   y<-xidep[,1]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   logit1<-alp1
##   logit2<-alp1+alp2
##   logit3<-alp1+alp2+alp3
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==0)*pge1+(y==1)*(pge2 - pge1)+(y==2)*(pge3 - pge2)+(y==3)*(1 - pge3)
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
##   Nb of parameters: 3
```

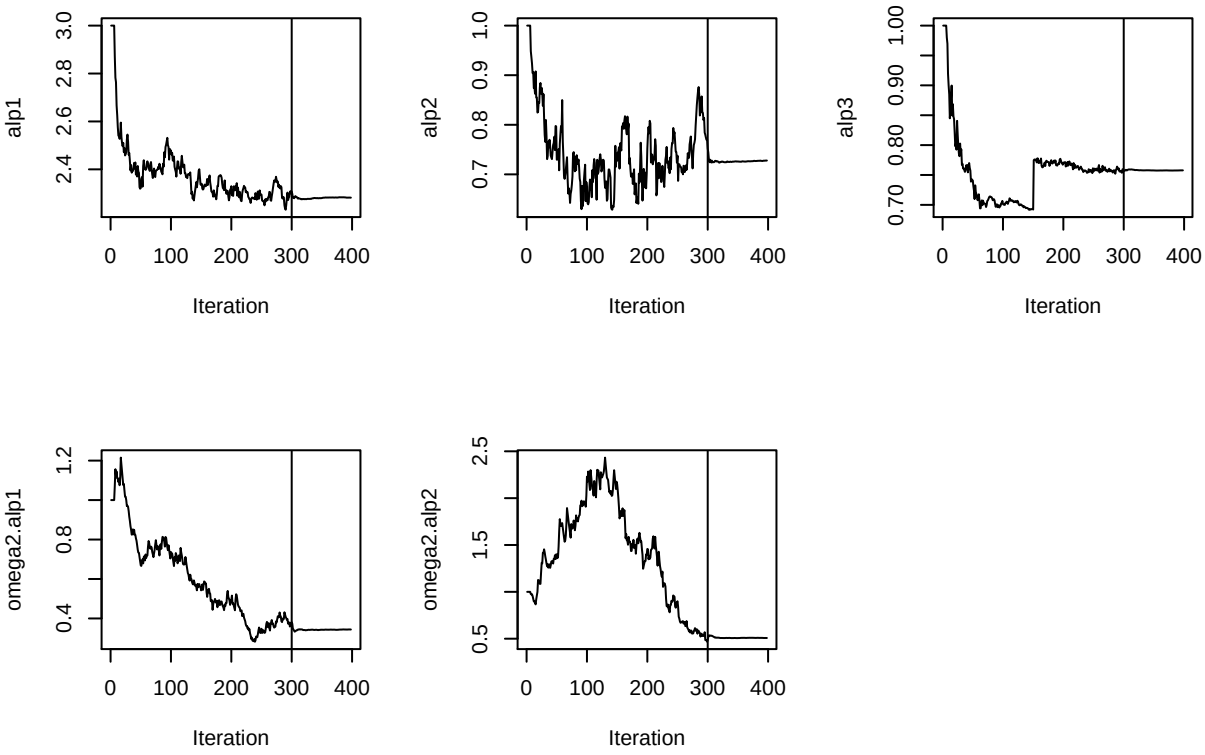
```
##       parameter names:  alp1 alp2 alp3
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3
## alp1    1    0    0
## alp2    0    1    0
## alp3    0    0    0
##     No covariate in the model.
##     Initial values
##              alp1 alp2 alp3
## Pop.CondInit    3    1    1
```

```
saemix.options<-list(seed=632545,save=FALSE,save.graphs=FALSE)
saemix.fit<-saemix(saemix.model,saemix.data,saemix.options)
```






alp1
alp2
alp3
omega2.alp1
omega2.alp2
alp1
alp2
alp3
omega2.alp1
omega2.alp2
Iteration




```
## Error in solve.default(FO) :
##   routine Lapack dgesv : le système est exactement singulier : U[2,2] = 0
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data             ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset smx.ord
##     Structured data: Y ~ Y | ID
##     Predictor: Y ()
## Dataset characteristics:
##     number of subjects:     1000
##     number of observations: 4000
##     average/min/max nb obs: 4.00  /  4  /  4
## First 10 lines of data:
##    ID Y Y.1 mdv cens occ ytype
## 1   1 3   3   0    0   1     1
## 2   1 0   0   0    0   1     1
## 3   1 0   0   0    0   1     1
## 4   1 0   0   0    0   1     1
## 5   2 3   3   0    0   1     1
## 6   2 0   0   0    0   1     1
## 7   2 0   0   0    0   1     1
## 8   2 0   0   0    0   1     1
## 9   3 0   0   0    0   1     1
## 10  3 0   0   0    0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
```

```
## Nonlinear mixed-effects model
##   Model function:  Ordinal categorical model  Model type:  likelihood
## function(psi,id,xidep) {
##   y<-xidep[,1]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   logit1<-alp1
##   logit2<-alp1+alp2
##   logit3<-alp1+alp2+alp3
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==0)*pge1+(y==1)*(pge2 - pge1)+(y==2)*(pge3 - pge2)+(y==3)*(1 - pge3)
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
## <bytecode: 0x56264f3040b8>
##   Nb of parameters: 3
##       parameter names:  alp1 alp2 alp3
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3
## alp1    1    0    0
## alp2    0    1    0
## alp3    0    0    0
##     No covariate in the model.
##     Initial values
##              alp1 alp2 alp3
## Pop.CondInit    3    1    1
## ----------------------------------
## ----    Key algorithm options  ----
## ----------------------------------
##     Estimation of individual parameters (MAP)
##     Estimation of standard errors and linearised log-likelihood
##     Estimation of log-likelihood by importance sampling
##     Number of iterations:  K1=300, K2=100
##     Number of chains:  1
##     Seed:  632545
##     Number of MCMC iterations for IS:  5000
##     Simulations:
##         nb of simulated datasets used for npde:  1000
##         nb of simulated datasets used for VPC:  100
##     Input/output
##         save the results to a file:  FALSE
##         save the graphs to files:  FALSE
## ----------------------------------------------------
## ----                  Results                   ----
```

```
## ------------------------------------------------
## -----------------  Fixed effects  ------------------
## ------------------------------------------------
##      Parameter Estimate SE    CV(%)
## [1,] alp1      2.28     0.083  3.6
## [2,] alp2      0.73     0.094 13.0
## [3,] alp3      0.76     0.122 16.1
## ------------------------------------------------
## -----------  Variance of random effects  -----------
## ------------------------------------------------
##      Parameter   Estimate SE CV(%)
## alp1 omega2.alp1 0.34     NA NA
## alp2 omega2.alp2 0.51     NA NA
## ------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ------------------------------------------------
##             omega2.alp1 omega2.alp2
## omega2.alp1 1           0
## omega2.alp2 0           1
## ------------------------------------------------
## ---------------  Statistical criteria  -------------
## ------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 12212.95
##       AIC = 12224.95
##       BIC = 12254.4
##
## Likelihood computed by importance sampling
##       -2LL= 3568.555
##       AIC = 3580.555
##       BIC = 3610.001
## ------------------------------------------------
```

```
ord.fit<-saemix.fit
```

Count model

TTE model