Testing examples in saemix 3.0

Emmanuelle

20/10/2020

Setup

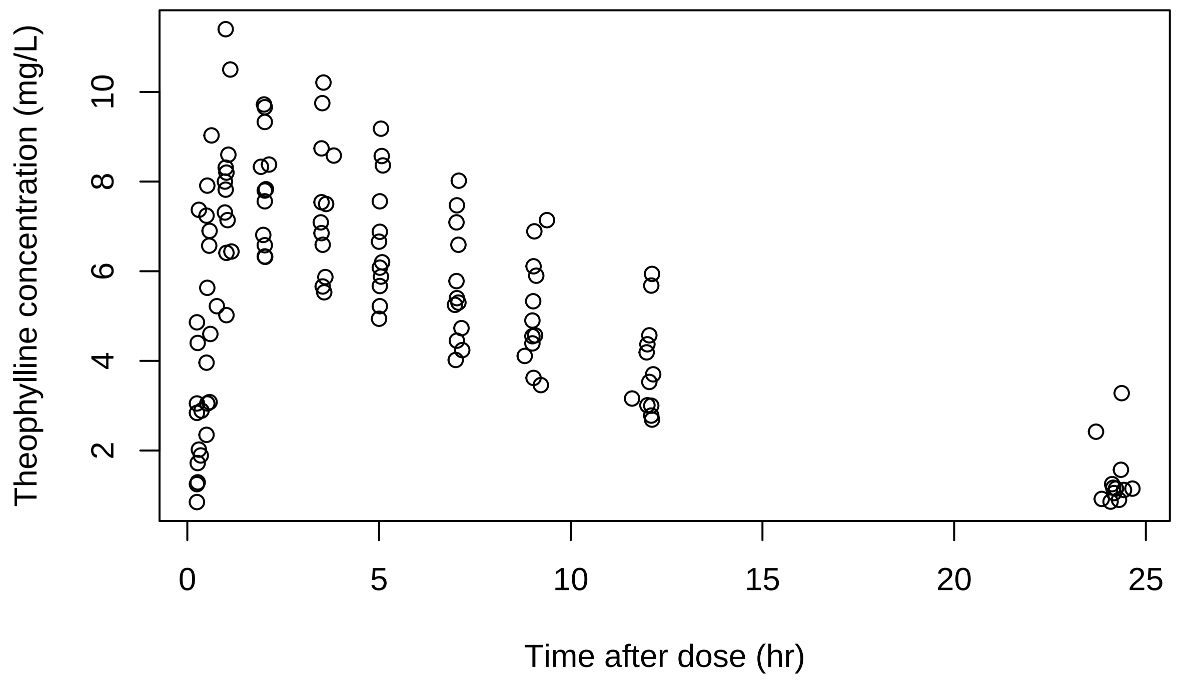
- set up work directories
- two versions toggled by testMode
  - if testMode is FALSE, load the functions in R
  - if testMode is TRUE, load the library in a dev_mode environment
- aim: check the examples used in the online documentation
  - all examples must run without error

Testing library

Continuous response model

Theophylline

```r
if(testMode)
  data(theo.saemix) else
    theo.saemix<-read.table(file.path(datDir, "theo.saemix.tab"), header=TRUE)

#Plotting the theophylline data
plot(Concentration~Time,data=theo.saemix,xlab="Time after dose (hr)",
ylab="Theophylline concentration (mg/L)")
```



```r
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")
```

```
## [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
```

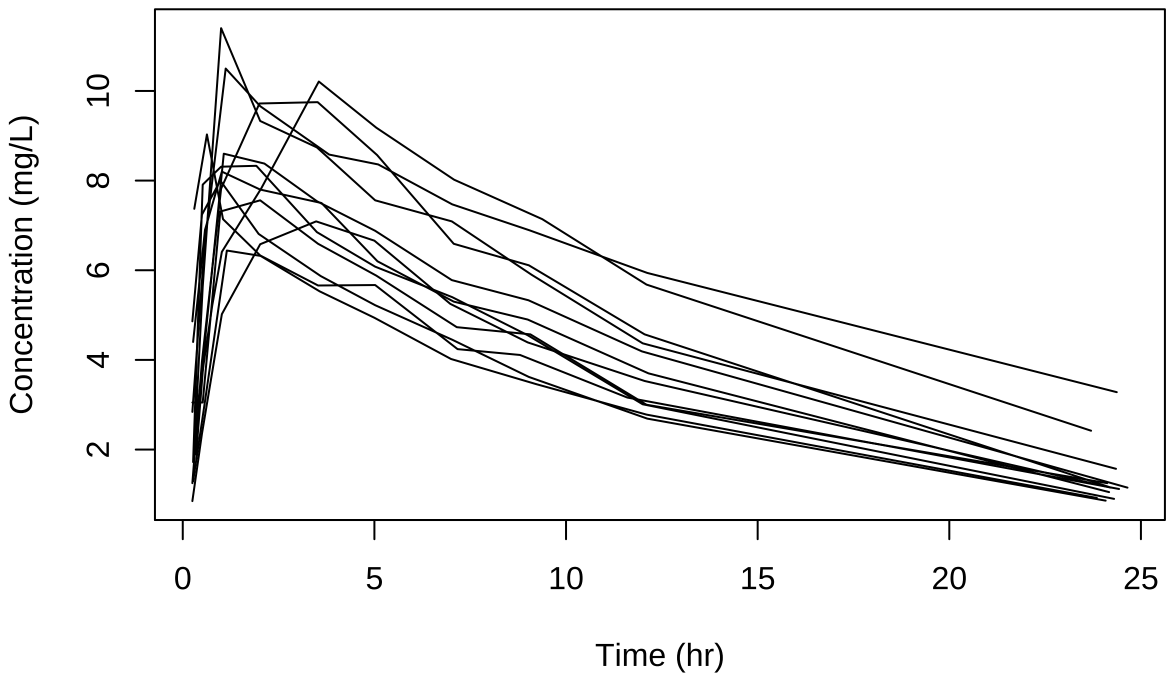
```r
  model1cpt<-function(psi,id,xidep) {
    dose<-xidep[,1]
    tim<-xidep[,2]
    ka<-psi[id,1]
    V<-psi[id,2]
    CL<-psi[id,3]
    k<-CL/V
    ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
    return(ypred)
    }
# Default model, no covariate
saemix.model<-saemixModel(model=model1cpt,
       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"))),transform.par=c(1,1,1))
```

```
##
##
## 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  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 1.0 20  0.50
## Cov.CondInit 0.1  0 -0.01
```

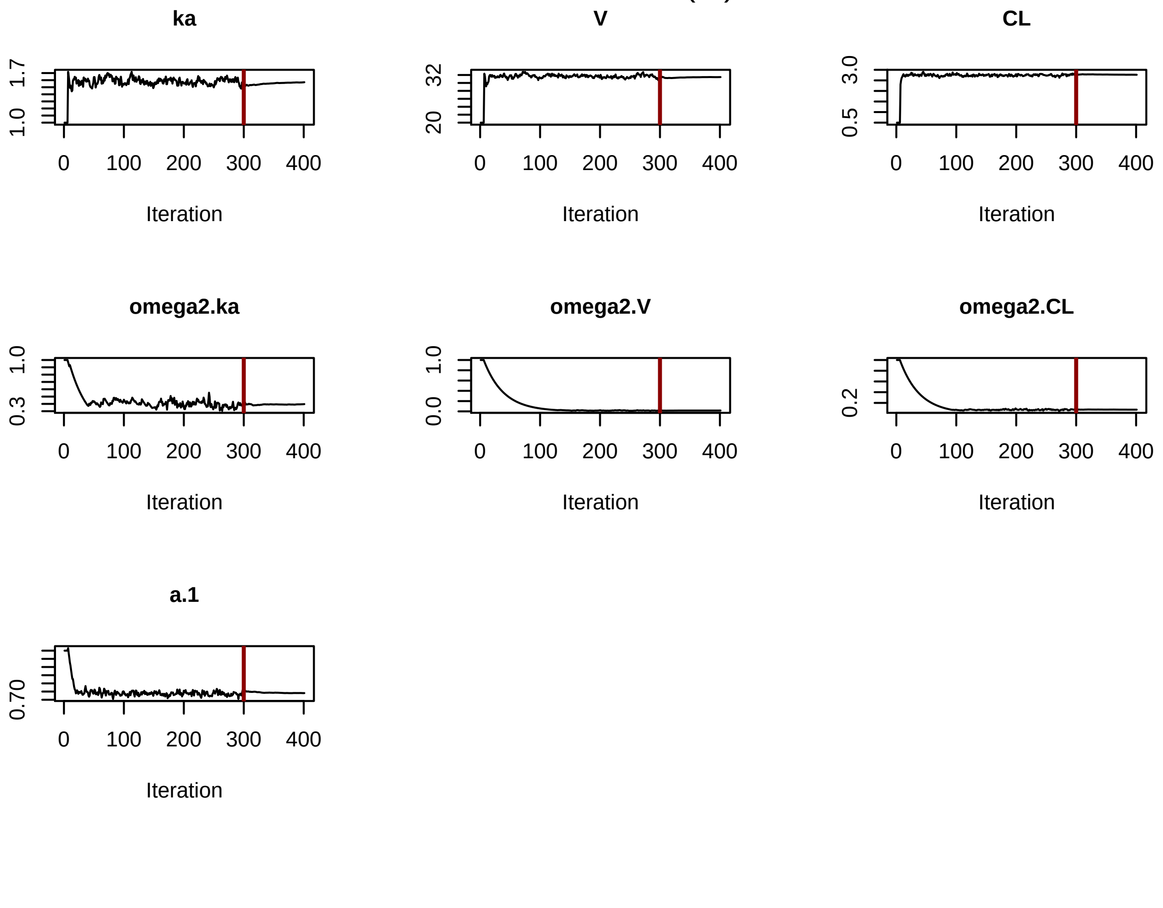
```
 # Note: remove the options save=FALSE and save.graphs=FALSE
 # to save the results and graphs
saemix.options<-list(seed=632545,save=FALSE,save.graphs=FALSE, displayProgress=FALSE)

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: 0x56239a9fb7a8>
##   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
##     No covariate in the model.
##     Initial values
##              ka  V  CL
## Pop.CondInit  1 20 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:  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(%)
## [1,] ka         1.57    0.304 19.3
## [2,] V         31.47    1.423  4.5
## [3,] CL         2.77    0.239  8.7
## [4,] a.1        0.74    0.057  7.7
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##    Parameter Estimate SE     CV(%)
## ka omega2.ka 0.397    0.1790 45
## V  omega2.V  0.017    0.0096 58
## CL omega2.CL 0.074    0.0360 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= 344.1136
##       AIC = 358.1136
##       BIC = 361.5079
##
## Likelihood computed by importance sampling
##       -2LL= 345.4329
##       AIC = 359.4329
##       BIC = 362.8273
## ----------------------------------------------------
```

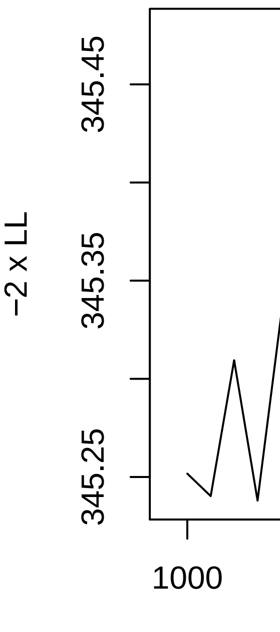
```r
plot(saemix.fit)
```



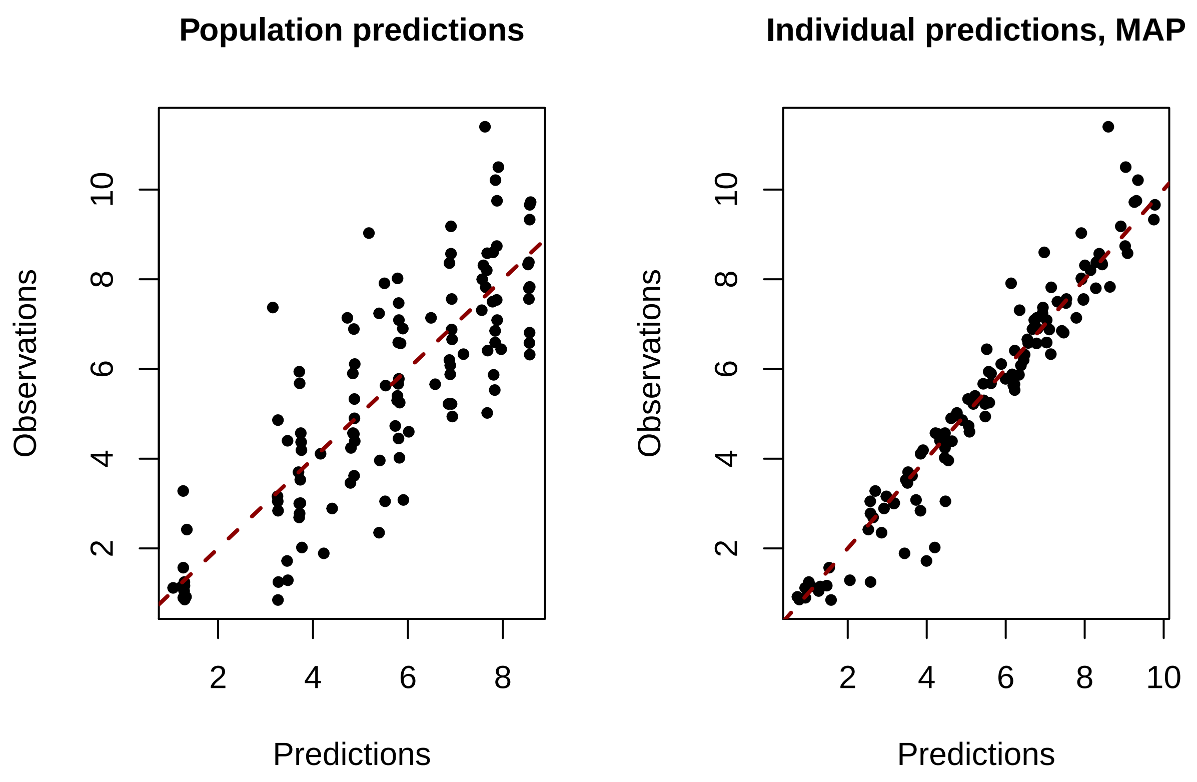
Concentration (mg/L)
Time (hr)



ka
V
CL
omega2.ka
omega2.V
omega2.CL
a.1
Iteration



−2 x LL
345.45
345.35
345.25
1000



```
# Model with covariates
saemix.model<-saemixModel(model=model1cpt,
                          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"))),transform.par=c(1,1,1),
                          covariate.model=matrix(c(0,0,1,0,0,0),ncol=3,byrow=TRUE),fixed.estim=c(1,1,1)
                          covariance.model=matrix(c(1,0,0,0,1,1,0,1,1),ncol=3,byrow=TRUE),
                          omega.init=matrix(c(1,0,0,0,1,0,0,0,1),ncol=3,byrow=TRUE),error.model="combin
```

```
##
##
## 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: 0x56239a9fb7a8>
##   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=1
##   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
```

```
saemix.options<-list(seed=39546,save=FALSE,save.graphs=FALSE, displayProgress=FALSE)
saemix.fit<-saemix(saemix.model,saemix.data,saemix.options)
```

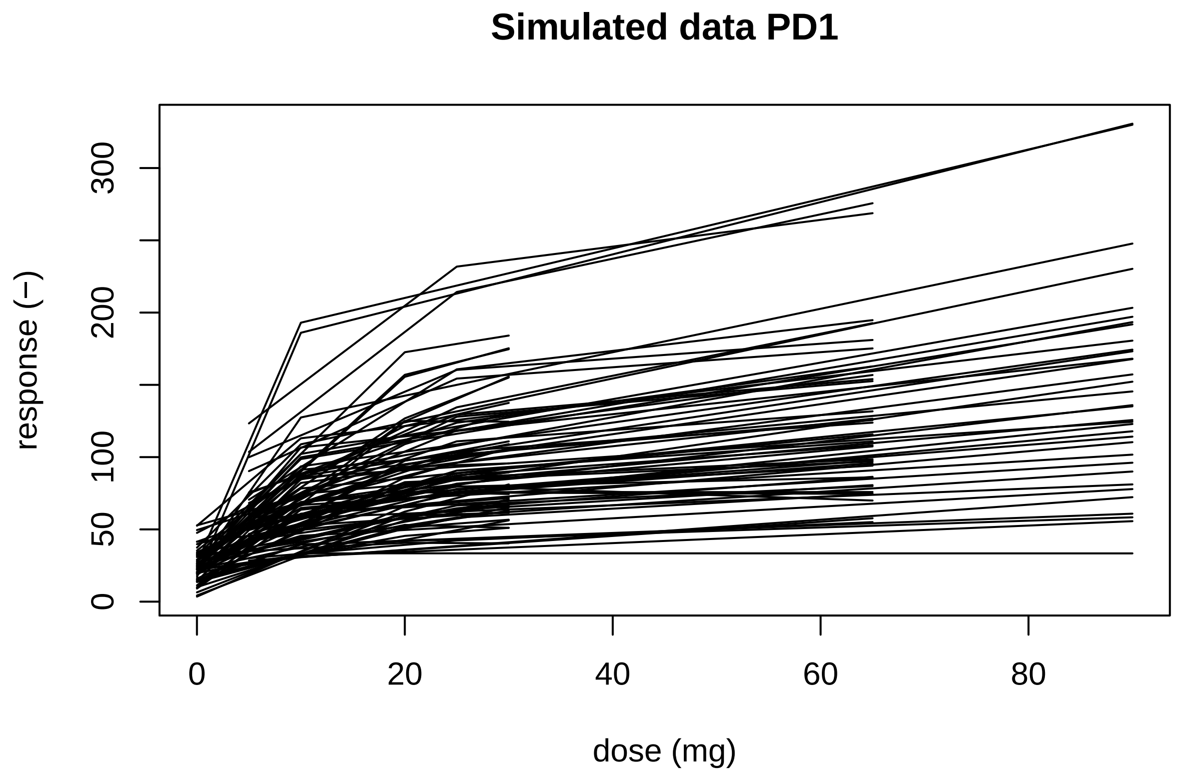
```
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data              ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset theo.saemix
##     Structured 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: 0x56239a9fb7a8>
##   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=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:  39546
##     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               1.5565  0.3050 19.6  -
## [2,] V               31.6621  1.4946  4.7  -
## [3,] CL               4.4308  1.9206 43.3  -
## [4,] beta_Weight(CL) -0.0067  0.0061 91.3  0.14
## [5,] a.1              0.5734  0.1211 21.1  -
## [6,] b.1              0.0748  0.0223 29.8  -
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##       Parameter Estimate SE    CV(%)
## ka    omega2.ka 0.412    0.179 44
## V     omega2.V  0.019    0.011 56
## CL    omega2.CL 0.064    0.031 48
## covar cov.V.CL  0.035    0.016 45
## ----------------------------------------------------
## ------  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= 330.7213
##       AIC = 350.7213
##       BIC = 355.5704
##
## Likelihood computed by importance sampling
##       -2LL= 333.9945
##       AIC = 353.9945
##       BIC = 358.8436
## ----------------------------------------------------
```

Simulated PD

```
if(testMode) {
  data(PD1.saemix)
  data(PD2.saemix)
  } else {
    PD1.saemix<-read.table(file.path(datDir, "PD1.saemix.tab"), header=TRUE)
    PD2.saemix<-read.table(file.path(datDir, "PD1.saemix.tab"), header=TRUE)
  }

saemix.data<-saemixData(name.data=PD1.saemix,header=TRUE,name.group=c("subject"),
      name.predictors=c("dose"),name.response=c("response"),
      name.covariates=c("gender"), units=list(x="mg",y="-",covariates=c("-")))
```

```
## [1] "gender"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset PD1.saemix
##     Structured data: response ~ dose | subject
##     Predictor: dose (mg)
##     covariates: gender (-)
##       reference class for covariate gender :  0
```

```
modelemax<-function(psi,id,xidep) {
# input:
#   psi : matrix of parameters (3 columns, E0, Emax, EC50)
#   id : vector of indices
#   xidep : dependent variables (same nb of rows as length of id)
# returns:
#   a vector of predictions of length equal to length of id
  dose<-xidep[,1]
  e0<-psi[id,1]
  emax<-psi[id,2]
  e50<-psi[id,3]
  f<-e0+emax*dose/(e50+dose)
  return(f)
}

# Plotting the data
plot(saemix.data,main="Simulated data PD1")
```



```
# Compare models with and without covariates with LL by Importance Sampling
model1<-saemixModel(model=modelemax,description="Emax growth model",
       psi0=matrix(c(20,300,20,0,0,0),ncol=3,byrow=TRUE,dimnames=list(NULL,
       c("E0","Emax","EC50"))), transform.par=c(1,1,1),
       covariate.model=matrix(c(0,0,0), ncol=3,byrow=TRUE),fixed.estim=c(1,1,1))
```

```
##
```

```
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Emax growth model  Model type:  structural
## function(psi,id,xidep) {
## # input:
## #   psi : matrix of parameters (3 columns, E0, Emax, EC50)
## #   id : vector of indices
## #   xidep : dependent variables (same nb of rows as length of id)
## # returns:
## #   a vector of predictions of length equal to length of id
##   dose<-xidep[,1]
##   e0<-psi[id,1]
##   emax<-psi[id,2]
##   e50<-psi[id,3]
##   f<-e0+emax*dose/(e50+dose)
##   return(f)
## }
##   Nb of parameters: 3
##       parameter names:  E0 Emax EC50
##       distribution:
##      Parameter Distribution Estimated
## [1,] E0        log-normal   Estimated
## [2,] Emax      log-normal   Estimated
## [3,] EC50      log-normal   Estimated
##   Variance-covariance matrix:
##      E0 Emax EC50
## E0    1    0    0
## Emax  0    1    0
## EC50  0    0    1
##   Error model: constant , initial values: a.1=1
##     No covariate in the model.
##     Initial values
##              E0 Emax EC50
## Pop.CondInit 20  300   20
## Cov.CondInit  0    0    0
```

```
model2<-saemixModel(model=modelemax,description="Emax growth model",
       psi0=matrix(c(20,300,20,0,0,0),ncol=3,byrow=TRUE,dimnames=list(NULL,
       c("E0","Emax","EC50"))), transform.par=c(1,1,1),
       covariate.model=matrix(c(0,0,1), ncol=3,byrow=TRUE),fixed.estim=c(1,1,1))
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Emax growth model  Model type:  structural
## function(psi,id,xidep) {
## # input:
## #   psi : matrix of parameters (3 columns, E0, Emax, EC50)
## #   id : vector of indices
## #   xidep : dependent variables (same nb of rows as length of id)
## # returns:
```

```
## #   a vector of predictions of length equal to length of id
##   dose<-xidep[,1]
##   e0<-psi[id,1]
##   emax<-psi[id,2]
##   e50<-psi[id,3]
##   f<-e0+emax*dose/(e50+dose)
##   return(f)
## }
##   Nb of parameters: 3
##       parameter names:  E0 Emax EC50
##       distribution:
##      Parameter Distribution Estimated
## [1,] E0        log-normal   Estimated
## [2,] Emax      log-normal   Estimated
## [3,] EC50      log-normal   Estimated
##   Variance-covariance matrix:
##      E0 Emax EC50
## E0    1    0    0
## Emax  0    1    0
## EC50  0    0    1
##   Error model: constant , initial values: a.1=1
##   Covariate model:
##      E0 Emax EC50
## [1,]  0    0    1
##     Initial values
##              E0 Emax EC50
## Pop.CondInit 20  300   20
## Cov.CondInit  0    0    0
```

```
# SE not computed as not needed for the test
saemix.options<-list(algorithms=c(0,1,1),nb.chains=3,seed=765754,
       nbiter.saemix=c(500,300),save=FALSE,save.graphs=FALSE, displayProgress=FALSE)

fit1<-saemix(model1,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 PD1.saemix
##     Structured data: response ~ dose | subject
##     Predictor: dose (mg)
##     covariates: gender (-)
##       reference class for covariate gender :  0
## Dataset characteristics:
##     number of subjects:     100
##     number of observations: 300
##     average/min/max nb obs: 3.00  /  3  /  3
## First 10 lines of data:
##    subject dose response gender mdv cens occ ytype
## 1        1    0  11.2870      1   0    0   1     1
## 2        1   10  63.6114      1   0    0   1     1
## 3        1   90 122.9170      1   0    0   1     1
## 4        2    0  15.0514      1   0    0   1     1
## 5        2   10  39.5296      1   0    0   1     1
## 6        2   90  60.8522      1   0    0   1     1
## 7        3    0  25.5390      1   0    0   1     1
## 8        3   10  58.0035      1   0    0   1     1
## 9        3   90  81.1173      1   0    0   1     1
## 10       4    0  22.1446      1   0    0   1     1
## ----------------------------------
## ----          Model            ----
## ----------------------------------
## Nonlinear mixed-effects model
##   Model function:  Emax growth model  Model type:  structural
## function(psi,id,xidep) {
## # input:
## #   psi : matrix of parameters (3 columns, E0, Emax, EC50)
## #   id : vector of indices
## #   xidep : dependent variables (same nb of rows as length of id)
## # returns:
## #   a vector of predictions of length equal to length of id
##   dose<-xidep[,1]
##   e0<-psi[id,1]
##   emax<-psi[id,2]
##   e50<-psi[id,3]
##   f<-e0+emax*dose/(e50+dose)
##   return(f)
## }
## <bytecode: 0x56239eac46f0>
##   Nb of parameters: 3
##       parameter names:  E0 Emax EC50
##       distribution:
##      Parameter Distribution Estimated
## [1,] E0        log-normal   Estimated
## [2,] Emax      log-normal   Estimated
## [3,] EC50      log-normal   Estimated
##   Variance-covariance matrix:
##      E0 Emax EC50
## E0    1    0    0
## Emax  0    1    0
## EC50  0    0    1
##   Error model: constant , initial values: a.1=1
##     No covariate in the model.
##     Initial values
##              E0 Emax EC50
## Pop.CondInit 20  300   20
## ----------------------------------
## ----    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:  3
##     Seed:  765754
##     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,] E0         23.4    1.08 4.6
## [2,] Emax      107.2    6.09 5.7
## [3,] EC50       15.2    0.77 5.0
## [4,] a.1         4.8    0.42 8.8
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##      Parameter   Estimate SE    CV(%)
## E0   omega2.E0   0.128    0.028 22
## Emax omega2.Emax 0.302    0.045 15
## EC50 omega2.EC50 0.071    0.027 38
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##             omega2.E0 omega2.Emax omega2.EC50
## omega2.E0   1         0           0
## omega2.Emax 0         1           0
## omega2.EC50 0         0           1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 2463.063
##       AIC = 2477.063
##       BIC = 2495.299
##
## Likelihood computed by importance sampling
##       -2LL= 2466.154
##       AIC = 2480.154
##       BIC = 2498.39
## ----------------------------------------------------
```

```
fit2<-saemix(model2,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 PD1.saemix
##     Structured data: response ~ dose | subject
##     Predictor: dose (mg)
##     covariates: gender (-)
##       reference class for covariate gender :  0
## Dataset characteristics:
##     number of subjects:     100
##     number of observations: 300
##     average/min/max nb obs: 3.00  /  3  /  3
## First 10 lines of data:
##    subject dose response gender mdv cens occ ytype
## 1        1    0  11.2870      1   0    0   1     1
## 2        1   10  63.6114      1   0    0   1     1
## 3        1   90 122.9170      1   0    0   1     1
## 4        2    0  15.0514      1   0    0   1     1
## 5        2   10  39.5296      1   0    0   1     1
## 6        2   90  60.8522      1   0    0   1     1
## 7        3    0  25.5390      1   0    0   1     1
## 8        3   10  58.0035      1   0    0   1     1
## 9        3   90  81.1173      1   0    0   1     1
## 10       4    0  22.1446      1   0    0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  Emax growth model  Model type:  structural
## function(psi,id,xidep) {
## # input:
## #   psi : matrix of parameters (3 columns, E0, Emax, EC50)
## #   id : vector of indices
## #   xidep : dependent variables (same nb of rows as length of id)
## # returns:
## #   a vector of predictions of length equal to length of id
##   dose<-xidep[,1]
##   e0<-psi[id,1]
##   emax<-psi[id,2]
##   e50<-psi[id,3]
##   f<-e0+emax*dose/(e50+dose)
##   return(f)
## }
## <bytecode: 0x56239eac46f0>
##   Nb of parameters: 3
##       parameter names:  E0 Emax EC50
##       distribution:
##      Parameter Distribution Estimated
## [1,] E0        log-normal   Estimated
## [2,] Emax      log-normal   Estimated
## [3,] EC50      log-normal   Estimated
##   Variance-covariance matrix:
##      E0 Emax EC50
## E0    1    0    0
## Emax  0    1    0
## EC50  0    0    1
##   Error model: constant , initial values: a.1=1
##   Covariate model:
##        [,1] [,2] [,3]
## gender    0    0    1
##     Initial values
##              E0 Emax EC50
## Pop.CondInit 20  300   20
## Cov.CondInit  0    0    0
## ----------------------------------
## ----    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:  3
##     Seed:  765754
##     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,] E0                 23.24   1.072  4.6  -
## [2,] Emax              107.20   6.120  5.7  -
## [3,] EC50               11.45   0.980  8.6  -
## [4,] beta_gender(EC50)   0.39   0.099 25.6  4.7e-05
## [5,] a.1                 4.72   0.407  8.6  -
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##      Parameter   Estimate SE    CV(%)
## E0   omega2.E0   0.129    0.028 22
## Emax omega2.Emax 0.307    0.045 15
## EC50 omega2.EC50 0.052    0.022 43
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##             omega2.E0 omega2.Emax omega2.EC50
## omega2.E0   1         0           0
## omega2.Emax 0         1           0
## omega2.EC50 0         0           1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 2448.635
##       AIC = 2464.635
##       BIC = 2485.477
##
## Likelihood computed by importance sampling
##       -2LL= 2452.279
##       AIC = 2468.279
##       BIC = 2489.121
## ----------------------------------------------------
```

```
wstat<-(-2)*(fit1["results"]["ll.is"]-fit2["results"]["ll.is"])

cat("LRT test for covariate effect on EC50: p-value=",1-pchisq(wstat,1),"\n")
```

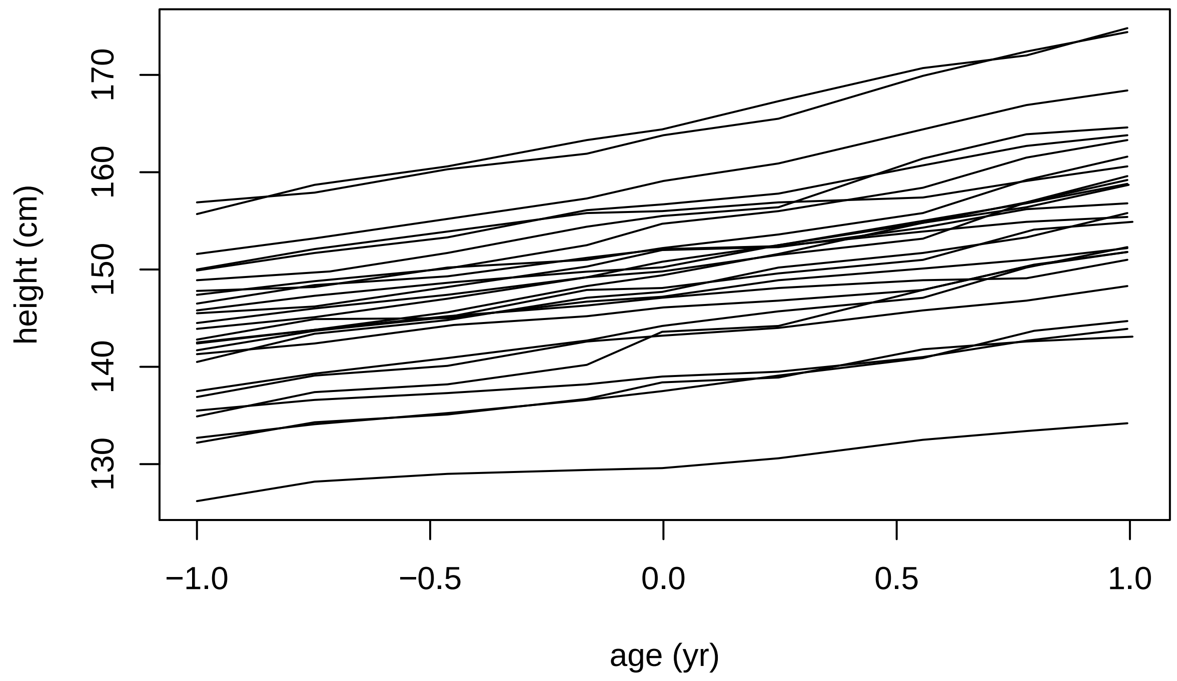
```
## LRT test for covariate effect on EC50: p-value= 0.0001954234
```

Oxford boys

```
if(testMode)
  data(oxboys.saemix) else
    oxboys.saemix<-read.table(file.path(datDir, "oxboys.saemix.tab"), header=TRUE)

saemix.data<-saemixData(name.data=oxboys.saemix,header=TRUE,
      name.group=c("Subject"),name.predictors=c("age"),name.response=c("height"),
      units=list(x="yr",y="cm"))
```

```
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset oxboys.saemix
##     Structured data: height ~ age | Subject
##     Predictor: age (yr)
```

```
# plot the data
plot(saemix.data)
```



```
growth.linear<-function(psi,id,xidep) {
  x<-xidep[,1]
  base<-psi[id,1]
  slope<-psi[id,2]
  f<-base+slope*x
  return(f)
}
saemix.model<-saemixModel(model=growth.linear,description="Linear model",
      psi0=matrix(c(140,1),ncol=2,byrow=TRUE,dimnames=list(NULL,c("base","slope"))),
      transform.par=c(1,0),covariance.model=matrix(c(1,1,1,1),ncol=2,byrow=TRUE),
      error.model="constant")
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Linear model  Model type:  structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   base<-psi[id,1]
##   slope<-psi[id,2]
##   f<-base+slope*x
##   return(f)
## }
##   Nb of parameters: 2
##       parameter names:  base slope
##       distribution:
##      Parameter Distribution Estimated
## [1,] base      log-normal   Estimated
## [2,] slope     normal       Estimated
##   Variance-covariance matrix:
##       base slope
## base     1     1
## slope    1     1
##   Error model: constant , initial values: a.1=1
##     No covariate in the model.
##     Initial values
##              base slope
## Pop.CondInit  140     1
```

```
saemix.options<-list(algorithms=c(1,1,1),nb.chains=1,seed=201004,
      save=FALSE,save.graphs=FALSE, displayProgress=FALSE)
saemix.fit<-saemix(saemix.model,saemix.data,saemix.options)
```

```
## The number of subjects is small, increasing the number of chains to 2 to improve convergence
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data             ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset oxboys.saemix
##     Structured data: height ~ age | Subject
##     Predictor: age (yr)
## Dataset characteristics:
##     number of subjects:     26
##     number of observations: 234
##     average/min/max nb obs: 9.00  /  9  /  9
## First 10 lines of data:
##    Subject     age height mdv cens occ ytype
## 1        1 -1.0000  140.5   0    0   1     1
## 2        1 -0.7479  143.4   0    0   1     1
## 3        1 -0.4630  144.8   0    0   1     1
## 4        1 -0.1643  147.1   0    0   1     1
## 5        1 -0.0027  147.7   0    0   1     1
## 6        1  0.2466  150.2   0    0   1     1
## 7        1  0.5562  151.7   0    0   1     1
## 8        1  0.7781  153.3   0    0   1     1
## 9        1  0.9945  155.8   0    0   1     1
## 10       2 -1.0000  136.9   0    0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  Linear model  Model type:  structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   base<-psi[id,1]
##   slope<-psi[id,2]
##   f<-base+slope*x
##   return(f)
## }
## <bytecode: 0x562398b194d0>
##   Nb of parameters: 2
##       parameter names:  base slope
##      distribution:
##     Parameter Distribution Estimated
## [1,] base      log-normal   Estimated
## [2,] slope     normal       Estimated
##   Variance-covariance matrix:
##      base slope
## base     1     1
## slope    1     1
##   Error model: constant , initial values: a.1=1
##     No covariate in the model.
##     Initial values
##             base slope
## Pop.CondInit  140     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:  2
##     Seed:  201004
##     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,] base      149.16   1.563 1.0
## [2,] slope       6.51   0.331 5.1
## [3,] a.1         0.66   0.035 5.2
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##      Parameter      Estimate SE      CV(%)
## base  omega2.base    0.0029   0.00079 28
## slope omega2.slope   2.7361   0.79109 29
## covar cov.base.slope 0.0564   0.02087 37
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##             omega2.base omega2.slope
## omega2.base  1.00        0.64
## omega2.slope 0.64        1.00
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 726.5422
##       AIC = 738.5422
##       BIC = 746.0908
##
## Likelihood computed by importance sampling
##       -2LL= 726.5619
##       AIC = 738.5619
##       BIC = 746.1105
## ----------------------------------------------------
```

Cow

```
if(testMode)
  data(cow.saemix) else
    cow.saemix<-read.table(file.path(datDir, "cow.saemix.tab"), header=TRUE)

saemix.data<-saemixData(name.data=cow.saemix,header=TRUE,name.group=c("cow"),
      name.predictors=c("time"),name.response=c("weight"),
      name.covariates=c("birthyear","twin","birthrank"),
      units=list(x="days",y="kg",covariates=c("yr","-","-")))
```

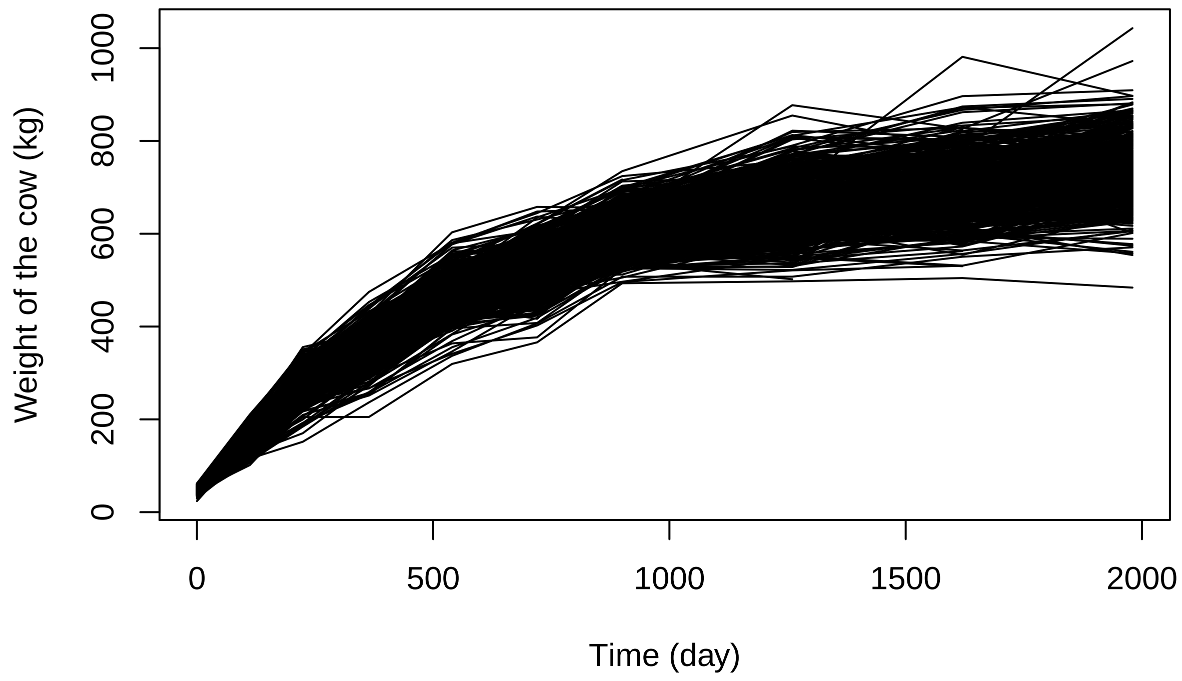
```
## [1] "birthyear" "twin"      "birthrank"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset cow.saemix
##     Structured data: weight ~ time | cow
##     Predictor: time (days)
##     covariates: birthyear (yr), twin (-), birthrank (-)
##       reference class for covariate twin :  1
```

```
growthcow<-function(psi,id,xidep) {
  x<-xidep[,1]
  a<-psi[id,1]
  b<-psi[id,2]
  k<-psi[id,3]
  f<-a*(1-b*exp(-k*x))
  return(f)
}
saemix.model<-saemixModel(model=growthcow,
      description="Exponential growth model",
      psi0=matrix(c(700,0.9,0.02,0,0,0),ncol=3,byrow=TRUE,
        dimnames=list(NULL,c("A","B","k"))),transform.par=c(1,1,1),fixed.estim=c(1,1,1),
      covariate.model=matrix(c(0,0,0),ncol=3,byrow=TRUE),
      covariance.model=matrix(c(1,0,0,0,1,0,0,0,1),ncol=3,byrow=TRUE),
      omega.init=matrix(c(1,0,0,0,1,0,0,0,1),ncol=3,byrow=TRUE),error.model="constant")
```

```
##
##
## The following SaemixModel object was successfully created:
```

```
##
## Nonlinear mixed-effects model
##   Model function:  Exponential growth model  Model type:  structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   a<-psi[id,1]
##   b<-psi[id,2]
##   k<-psi[id,3]
##   f<-a*(1-b*exp(-k*x))
##   return(f)
## }
##   Nb of parameters: 3
##       parameter names:  A B k
##       distribution:
##      Parameter Distribution Estimated
## [1,] A         log-normal   Estimated
## [2,] B         log-normal   Estimated
## [3,] k         log-normal   Estimated
##   Variance-covariance matrix:
##   A B k
## A 1 0 0
## B 0 1 0
## k 0 0 1
##   Error model: constant , initial values: a.1=1
##     No covariate in the model.
##     Initial values
##                  A   B    k
## Pop.CondInit 700 0.9 0.02
## Cov.CondInit   0 0.0 0.00
```

```
saemix.options<-list(algorithms=c(1,1,1),nb.chains=1,nbiter.saemix=c(200,100),
             seed=4526,save=FALSE,save.graphs=FALSE, displayProgress=FALSE)

# Plotting the data
plot(saemix.data,xlab="Time (day)",ylab="Weight of the cow (kg)")
```



```
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 cow.saemix
##     Structured data: weight ~ time | cow
##     Predictor: time (days)
##     covariates: birthyear (yr), twin (-), birthrank (-)
##       reference class for covariate twin :  1
## Dataset characteristics:
##     number of subjects:     560
##     number of observations: 5455
##     average/min/max nb obs: 9.74  /  7  /  10
## First 10 lines of data:
##        cow time weight birthyear twin birthrank mdv cens occ ytype
## 1  1988005    0   44.0      1988    1         3   0    0   1     1
## 2  1988005  112  173.4      1988    1         3   0    0   1     1
## 3  1988005  224  292.8      1988    1         3   0    0   1     1
## 4  1988005  364  364.6      1988    1         3   0    0   1     1
## 5  1988005  540  490.4      1988    1         3   0    0   1     1
## 6  1988005  720  522.0      1988    1         3   0    0   1     1
## 7  1988005  900  601.1      1988    1         3   0    0   1     1
## 8  1988005 1260  698.1      1988    1         3   0    0   1     1
## 9  1988005 1620  657.7      1988    1         3   0    0   1     1
## 10 1988005 1980  776.7      1988    1         3   0    0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  Exponential growth model  Model type:  structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   a<-psi[id,1]
##   b<-psi[id,2]
##   k<-psi[id,3]
##   f<-a*(1-b*exp(-k*x))
##   return(f)
## }
## <bytecode: 0x56239f618fb8>
##   Nb of parameters: 3
##       parameter names:  A B k
##       distribution:
##      Parameter Distribution Estimated
## [1,] A         log-normal   Estimated
## [2,] B         log-normal   Estimated
## [3,] k         log-normal   Estimated
##   Variance-covariance matrix:
##   A B k
## A 1 0 0
## B 0 1 0
## k 0 0 1
##   Error model: constant , initial values: a.1=1
##     No covariate in the model.
##     Initial values
##                A   B    k
## Pop.CondInit 700 0.9 0.02
## -----------------------------------
## ----    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=200, K2=100
##     Number of chains:  1
##     Seed:  4526
##     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,] A         7.5e+02  2.9e+00 0.38
## [2,] B         9.4e-01  1.2e-03 0.13
## [3,] k         1.6e-03  1.2e-05 0.72
## [4,] a.1       2.7e+01  3.0e-01 1.12
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##   Parameter Estimate SE      CV(%)
## A omega2.A  6.4e-03  4.4e-04   7.0
## B omega2.B  4.7e-05  5.2e-05 110.7
## k omega2.k  1.4e-02  1.4e-03   9.8
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##          omega2.A omega2.B omega2.k
## omega2.A 1        0        0
## omega2.B 0        1        0
## omega2.k 0        0        1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 53732
##       AIC = 53746
##       BIC = 53776.29
##
## Likelihood computed by importance sampling
##       -2LL= 53731.51
##       AIC = 53745.51
##       BIC = 53775.8
## ----------------------------------------------------
```

Wheat yield

```
if(testMode)
  data(yield.saemix) else
    yield.saemix<-read.table(file.path(datDir, "yield.saemix.tab"), header=TRUE)
saemix.data<-saemixData(name.data=yield.saemix,header=TRUE,name.group=c("site"),
      name.predictors=c("dose"),name.response=c("yield"),
      name.covariates=c("soil.nitrogen"),units=list(x="kg/ha",y="t/ha",covariates=c("kg/ha")))
```

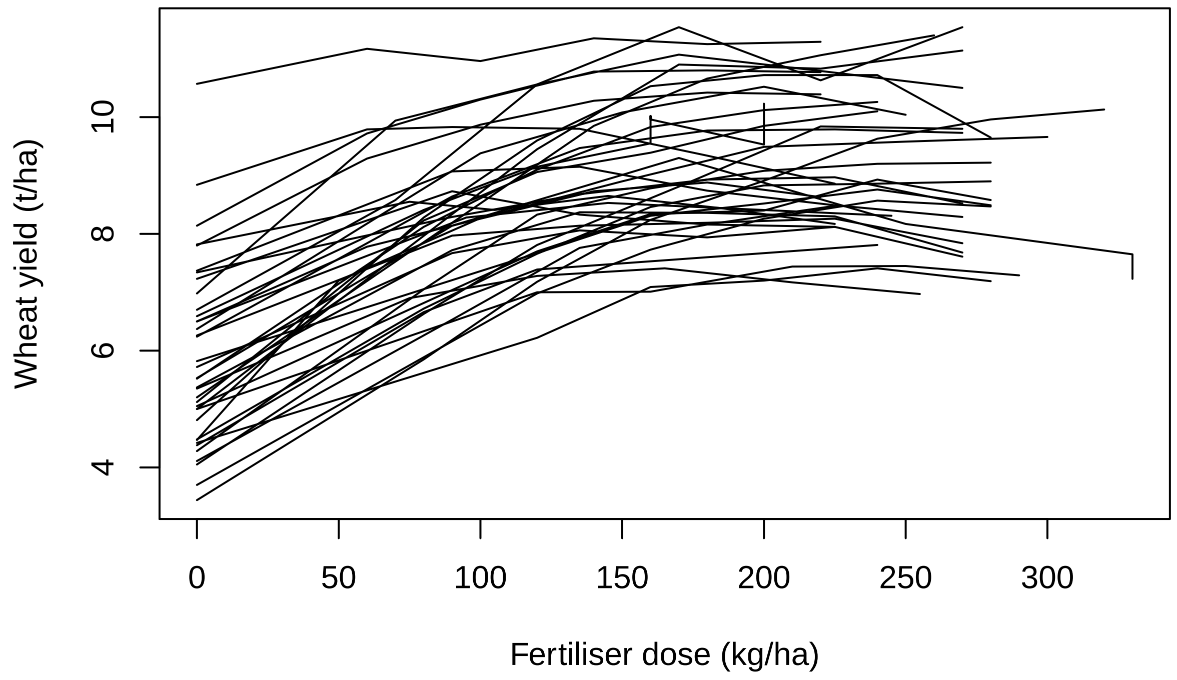
```
## [1] "soil.nitrogen"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset yield.saemix
##     Structured data: yield ~ dose | site
##     Predictor: dose (kg/ha)
##     covariates: soil.nitrogen (kg/ha)
```

```
#  Model: linear + plateau
yield.LP<-function(psi,id,xidep) {
  x<-xidep[,1]
  ymax<-psi[id,1]
  xmax<-psi[id,2]
  slope<-psi[id,3]
  f<-ymax+slope*(x-xmax)
  #'  cat(length(f),"  ",length(ymax),"\n")
  f[x>xmax]<-ymax[x>xmax]
  return(f)
}
saemix.model<-saemixModel(model=yield.LP,description="Linear plus plateau model",
        psi0=matrix(c(8,100,0.2,0,0,0),ncol=3,byrow=TRUE,dimnames=list(NULL,
            c("Ymax","Xmax","slope"))),covariate.model=matrix(c(0,0,0),ncol=3,byrow=TRUE),
        transform.par=c(0,0,0),covariance.model=matrix(c(1,0,0,0,1,0,0,0,1),ncol=3,
            byrow=TRUE),error.model="constant")
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Linear plus plateau model  Model type:  structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   ymax<-psi[id,1]
##   xmax<-psi[id,2]
##   slope<-psi[id,3]
##   f<-ymax+slope*(x-xmax)
##   #'  cat(length(f),"  ",length(ymax),"\n")
##   f[x>xmax]<-ymax[x>xmax]
##   return(f)
## }
##   Nb of parameters: 3
##       parameter names:  Ymax Xmax slope
##       distribution:
##      Parameter Distribution Estimated
## [1,] Ymax      normal       Estimated
## [2,] Xmax      normal       Estimated
## [3,] slope     normal       Estimated
##   Variance-covariance matrix:
##       Ymax Xmax slope
## Ymax     1    0     0
## Xmax     0    1     0
## slope    0    0     1
##   Error model: constant , initial values: a.1=1
##     No covariate in the model.
##     Initial values
##              Ymax Xmax slope
## Pop.CondInit    8  100   0.2
## Cov.CondInit    0    0   0.0
```

```
saemix.options<-list(algorithms=c(1,1,1),nb.chains=1,seed=666,
       save=FALSE,save.graphs=FALSE, displayProgress=FALSE)

# Plotting the data
```

```r
plot(saemix.data,xlab="Fertiliser dose (kg/ha)", ylab="Wheat yield (t/ha)")
```



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

```
## The number of subjects is small, increasing the number of chains to 2 to improve convergence
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data              ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset yield.saemix
##     Structured data: yield ~ dose | site
##     Predictor: dose (kg/ha)
##     covariates: soil.nitrogen (kg/ha)
## Dataset characteristics:
##     number of subjects:     37
##     number of observations: 224
##     average/min/max nb obs: 6.05  /  5  /  8
## First 10 lines of data:
##    site dose yield soil.nitrogen mdv cens occ ytype
## 1  1901    0  6.70            70   0    0   1     1
## 2  1901   70  8.58            70   0    0   1     1
## 3  1901  120 10.56            70   0    0   1     1
## 4  1901  170 11.54            70   0    0   1     1
## 5  1901  220 10.63            70   0    0   1     1
## 6  1901  270 11.54            70   0    0   1     1
## 7  1902    0  6.98            80   0    0   1     1
## 8  1902   70  9.94            80   0    0   1     1
## 9  1902  120 10.56            80   0    0   1     1
## 10 1902  170 11.07           80   0    0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  Linear plus plateau model  Model type:  structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   ymax<-psi[id,1]
##   xmax<-psi[id,2]
##   slope<-psi[id,3]
##   f<-ymax+slope*(x-xmax)
##   #'  cat(length(f),"  ",length(ymax),"\n")
##   f[x>xmax]<-ymax[x>xmax]
##   return(f)
## }
## <bytecode: 0x562398a17c00>
##   Nb of parameters: 3
##       parameter names:  Ymax Xmax slope
##       distribution:
##      Parameter Distribution Estimated
## [1,] Ymax      normal       Estimated
## [2,] Xmax      normal       Estimated
## [3,] slope     normal       Estimated
##   Variance-covariance matrix:
##       Ymax Xmax slope
## Ymax     1    0     0
## Xmax     0    1     0
## slope    0    0     1
##   Error model: constant , initial values: a.1=1
##     No covariate in the model.
##     Initial values
##              Ymax Xmax slope
## Pop.CondInit    8  100   0.2
## -----------------------------------
## ----    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:  2
##     Seed:  666
##     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,] Ymax       8.89    0.176  2.0
## [2,] Xmax      19.75    5.089 25.8
## [3,] slope      0.15    0.037 24.7
## [4,] a.1        0.71    0.041  5.8
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##       Parameter    Estimate SE      CV(%)
## Ymax  omega2.Ymax  1.0e+00   0.2659    25
## Xmax  omega2.Xmax  5.3e+01  38.0311    72
## slope omega2.slope 9.2e-06   0.0018 19486
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##              omega2.Ymax omega2.Xmax omega2.slope
## omega2.Ymax  1           0           0
## omega2.Xmax  0           1           0
## omega2.slope 0           0           1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 616.5701
##       AIC = 630.5701
##       BIC = 641.8466
##
## Likelihood computed by importance sampling
##       -2LL= 616.5048
##       AIC = 630.5048
##       BIC = 641.7812
## ----------------------------------------------------
```

```
# Comparing the likelihoods obtained by linearisation and importance sampling
# to the likelihood obtained by Gaussian Quadrature
saemix.fit<-llgq.saemix(saemix.fit)
{
   cat("LL by Importance sampling, LL_IS=",saemix.fit["results"]["ll.is"],"\n")
   cat("LL by linearisation, LL_lin=",saemix.fit["results"]["ll.lin"],"\n")
   cat("LL by Gaussian Quadrature, LL_GQ=",saemix.fit["results"]["ll.gq"],"\n")
}
```

```
## LL by Importance sampling, LL_IS= -308.2524
## LL by linearisation, LL_lin= -308.2851
## LL by Gaussian Quadrature, LL_GQ= -308.2772
```

```
# Testing for an effect of covariate soil.nitrogen on Xmax
saemix.model2<-saemixModel(model=yield.LP,description="Linear plus plateau model",
        psi0=matrix(c(8,100,0.2,0,0,0),ncol=3,byrow=TRUE,dimnames=list(NULL,
           c("Ymax","Xmax","slope"))),covariate.model=matrix(c(0,1,0),ncol=3,byrow=TRUE),
        transform.par=c(0,0,0),covariance.model=matrix(c(1,0,0,0,1,0,0,0,1),ncol=3,
            byrow=TRUE),error.model="constant")
```

```
##
##
```

```
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Linear plus plateau model  Model type:  structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   ymax<-psi[id,1]
##   xmax<-psi[id,2]
##   slope<-psi[id,3]
##   f<-ymax+slope*(x-xmax)
##   #'  cat(length(f),"  ",length(ymax),"\n")
##   f[x>xmax]<-ymax[x>xmax]
##   return(f)
## }
## <bytecode: 0x562398a17c00>
##   Nb of parameters: 3
##       parameter names:  Ymax Xmax slope
##       distribution:
##      Parameter Distribution Estimated
## [1,] Ymax      normal       Estimated
## [2,] Xmax      normal       Estimated
## [3,] slope     normal       Estimated
##   Variance-covariance matrix:
##       Ymax Xmax slope
## Ymax     1    0     0
## Xmax     0    1     0
## slope    0    0     1
##   Error model: constant , initial values: a.1=1
##   Covariate model:
##      Ymax Xmax slope
## [1,]    0    1     0
##     Initial values
##              Ymax Xmax slope
## Pop.CondInit    8  100   0.2
## Cov.CondInit    0    0   0.0
```

```
saemix.fit2<-saemix(saemix.model2,saemix.data,saemix.options)
```

```
## The number of subjects is small, increasing the number of chains to 2 to improve convergence
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data             ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset yield.saemix
##     Structured data: yield ~ dose | site
##     Predictor: dose (kg/ha)
##     covariates: soil.nitrogen (kg/ha)
## Dataset characteristics:
##     number of subjects:     37
##     number of observations: 224
##     average/min/max nb obs: 6.05  /  5  /  8
## First 10 lines of data:
##    site dose yield soil.nitrogen mdv cens occ ytype
## 1  1901    0  6.70           70   0    0   1     1
## 2  1901   70  8.58           70   0    0   1     1
## 3  1901  120 10.56           70   0    0   1     1
## 4  1901  170 11.54           70   0    0   1     1
## 5  1901  220 10.63           70   0    0   1     1
## 6  1901  270 11.54           70   0    0   1     1
## 7  1902    0  6.98           80   0    0   1     1
## 8  1902   70  9.94           80   0    0   1     1
## 9  1902  120 10.56           80   0    0   1     1
## 10 1902  170 11.07           80   0    0   1     1
## ----------------------------------
## ----          Model           ----
## ----------------------------------
## Nonlinear mixed-effects model
##   Model function:  Linear plus plateau model  Model type:  structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   ymax<-psi[id,1]
##   xmax<-psi[id,2]
##   slope<-psi[id,3]
##   f<-ymax+slope*(x-xmax)
##   #'  cat(length(f),"  ",length(ymax),"\n")
##   f[x>xmax]<-ymax[x>xmax]
##   return(f)
## }
## <bytecode: 0x562398a17c00>
##   Nb of parameters: 3
##       parameter names:  Ymax Xmax slope
##       distribution:
##      Parameter Distribution Estimated
## [1,] Ymax      normal       Estimated
## [2,] Xmax      normal       Estimated
## [3,] slope     normal       Estimated
##   Variance-covariance matrix:
##       Ymax Xmax slope
## Ymax     1    0     0
## Xmax     0    1     0
## slope    0    0     1
##   Error model: constant , initial values: a.1=1
##   Covariate model:
##               [,1] [,2] [,3]
## soil.nitrogen    0    1    0
##     Initial values
##              Ymax Xmax slope
## Pop.CondInit    8  100   0.2
## Cov.CondInit    0    0   0.0
## ----------------------------------
## ----    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:  2
##     Seed:  666
##     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,] Ymax                       9.184   0.1919  2.1  -
## [2,] Xmax                     218.403  15.7188  7.2  -
## [3,] beta_soil.nitrogen(Xmax)  -1.106   0.1715 15.5  5.8e-11
## [4,] slope                      0.026   0.0012  4.7  -
## [5,] a.1                        0.302   0.0192  6.4  -
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##       Parameter    Estimate SE      CV(%)
## Ymax  omega2.Ymax  1.3e+00  3.2e-01 24
## Xmax  omega2.Xmax  1.0e+03  2.9e+02 28
## slope omega2.slope 2.9e-05  1.1e-05 38
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##              omega2.Ymax omega2.Xmax omega2.slope
## omega2.Ymax  1           0           0
## omega2.Xmax  0           1           0
## omega2.slope 0           0           1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 389.099
##       AIC = 405.099
##       BIC = 417.9863
##
## Likelihood computed by importance sampling
##       -2LL= 380.8696
##       AIC = 396.8696
##       BIC = 409.7569
## ----------------------------------------------------
```

```r
# BIC for the two models
{
  cat("Model without covariate, BIC=",saemix.fit["results"]["bic.is"],"\n")
  cat("Model with covariate, BIC=",saemix.fit2["results"]["bic.is"],"\n")
  pval<-1-pchisq(-2*saemix.fit["results"]["ll.is"]+2*saemix.fit2["results"]["ll.is"],1)
  cat("        LRT: p=",pval,"\n")
}
```

```
## Model without covariate, BIC= 641.7812
## Model with covariate, BIC= 409.7569
##         LRT: p= 0
```

Discrete data model

Binary response model

Toenail data

- **TODO**
  - add diagnostics (npd-categorical ?)

```
if(testMode)
  data(toenail.saemix) else
    toenail.saemix<-read.table(file.path(datDir, "toenail.saemix.tab"), header=TRUE)

saemix.data<-saemixData(name.data=toenail.saemix,name.group=c("id"),name.predictors=c("time","y"), name
                        name.covariates=c("treatment"),name.X=c("time"))
```

```
## [1] "treatment"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset toenail.saemix
##     Structured data: y ~ time + y | id
##     X variable for graphs: time ()
##     covariates: treatment (-)
##       reference class for covariate treatment :  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,-.5,0,0.5),ncol=2,byrow=TRUE,dimnames=list(NULL,c("theta1","th
                          transform.par=c(0,0), covariate.model=c(0,1),covariance.model=matrix(c(1,0,0,
```

```
##
##
## 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:  theta1 theta2
##       distribution:
##      Parameter Distribution Estimated
## [1,] theta1    normal       Estimated
## [2,] theta2    normal       Estimated
##   Variance-covariance matrix:
##        theta1 theta2
## theta1      1      0
## theta2      0      1
##   Covariate model:
##      theta1 theta2
## [1,]      0      1
##     Initial values
##             theta1 theta2
## Pop.CondInit      0   -0.5
## Cov.CondInit      0    0.5
```

```
saemix.options<-list(seed=1234567,save=FALSE,save.graphs=FALSE, displayProgress=FALSE, nb.chains=10, fi

binary.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 toenail.saemix
##     Structured data: y ~ time + y | id
##     X variable for graphs: time ()
##     covariates: treatment (-)
##       reference class for covariate treatment :  0
## Dataset characteristics:
##     number of subjects:     294
##     number of observations: 1908
##     average/min/max nb obs: 6.49  /  1  /  7
## First 10 lines of data:
##    id      time y y.1 treatment mdv cens occ ytype
## 1   1 0.0000000 1   1         1   0    0   1     1
## 2   1 0.8571429 1   1         1   0    0   1     1
## 3   1  3.5357143 1   1         1   0    0   1     1
## 4   1  4.5357143 0   0         1   0    0   1     1
## 5   1  7.5357143 0   0         1   0    0   1     1
## 6   1 10.0357143 0   0         1   0    0   1     1
## 7   1 13.0714286 0   0         1   0    0   1     1
## 8   2  0.0000000 0   0         0   0    0   1     1
## 9   2  0.9642857 0   0         0   0    0   1     1
## 10  2  2.0000000 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: 0x56239f29ff70>
##   Nb of parameters: 2
##       parameter names:  theta1 theta2
##       distribution:
##      Parameter Distribution Estimated
## [1,] theta1    normal       Estimated
## [2,] theta2    normal       Estimated
##   Variance-covariance matrix:
##        theta1 theta2
## theta1      1      0
## theta2      0      1
##   Covariate model:
##           [,1] [,2]
## treatment    0    1
##     Initial values
##              theta1 theta2
## Pop.CondInit      0   -0.5
## Cov.CondInit      0    0.5
## ----------------------------------
## ----    Key algorithm options  ----
## ----------------------------------
##     Estimation of individual parameters (MAP)
##     Estimation of log-likelihood by importance sampling
##     Number of iterations:  K1=300, K2=100
##     Number of chains:  10
##     Seed:  1234567
##     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
## [1,] theta1                 -2.20
## [2,] theta2                 -1.25
## [3,] beta_treatment(theta2) -0.47
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##        Parameter     Estimate
## theta1 omega2.theta1 59.3
## theta2 omega2.theta2  1.1
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##               omega2.theta1 omega2.theta2
## omega2.theta1 1             0
## omega2.theta2 0             1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
##
## Likelihood computed by importance sampling
##       -2LL= 1116.755
##       AIC = 1128.755
##       BIC = 1150.856
## ----------------------------------------------------
```

Categorical response model

- Knee pain after 3, 7 and 10 days of treatment compared to baseline (time=0)
  - longitudinal ordinal model with 5 categories
  - similar results to Monolix in terms of parameter estimates
  - SE don't seem so off, but still higher than Monolix (but computed with linearisation anyway)
- Comparing the 3 covariate models - model with Age on alp1 and treatment on beta best

```
if(testMode)
  data(knee.saemix) else
    knee.saemix<-read.table(file.path(datDir, "knee.saemix.tab"), header=TRUE)

saemix.data<-saemixData(name.data=knee.saemix,name.group=c("id"),
                        name.predictors=c("y", "time"), name.X=c("time"),
                        name.covariates = c("Age","Sex","treatment"))
```

```
## 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] "Age"       "Sex"       "treatment"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset knee.saemix
##     Structured data: y ~ y + time | id
##     X variable for graphs: time ()
##     covariates: Age (-), Sex (-), treatment (-)
##       reference class for covariate Sex :  0
##       reference class for covariate treatment :  0
```

```
ordinal.model<-function(psi,id,xidep) {
  y<-xidep[,1]
  time<-xidep[,2]
  alp1<-psi[id,1]
  alp2<-psi[id,2]
  alp3<-psi[id,3]
  alp4<-psi[id,4]
  beta<-psi[id,5]

  logit1<-alp1 + beta*time
  logit2<-logit1+alp2
  logit3<-logit2+alp3
  logit4<-logit3+alp4
  pge1<-exp(logit1)/(1+exp(logit1))
  pge2<-exp(logit2)/(1+exp(logit2))
  pge3<-exp(logit3)/(1+exp(logit3))
  pge4<-exp(logit4)/(1+exp(logit4))
  logpdf<-rep(0,length(y))
  P.obs = (y==1)*pge1+(y==2)*(pge2 - pge1)+(y==3)*(pge3 - pge2)+(y==4)*(pge4 - pge3)+(y==5)*(1 - pge4)
  logpdf <- log(P.obs)

  return(logpdf)
}
covmodel3<-covmodel2<-covmodel1<-matrix(data=0,ncol=5,nrow=3)
covmodel1[1:2,1]<-1
covmodel1[,5]<-1
covmodel2[1,1]<-covmodel2[3,5]<-1
covmodel2<-covmodel<-matrix(data=0,ncol=5,nrow=3)
covmodel3[1,1]<-1

saemix.model<-saemixModel(model=ordinal.model,description="Ordinal categorical model",modeltype="likelil
                          psi0=matrix(c(0,0.2, 0.6, 3, 0.2),ncol=5,byrow=TRUE,dimnames=list(NULL,c("alp
                          transform.par=c(0,1,1,1,1),omega.init=diag(rep(1,5)), covariance.model = diag
```

```
##
##
## 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]
##   time<-xidep[,2]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   alp4<-psi[id,4]
##   beta<-psi[id,5]
##
##   logit1<-alp1 + beta*time
##   logit2<-logit1+alp2
##   logit3<-logit2+alp3
##   logit4<-logit3+alp4
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   pge4<-exp(logit4)/(1+exp(logit4))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==1)*pge1+(y==2)*(pge2 - pge1)+(y==3)*(pge3 - pge2)+(y==4)*(pge4 - pge3)+(y==5)*(1 - pge
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
##   Nb of parameters: 5
##       parameter names:  alp1 alp2 alp3 alp4 beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
## [4,] alp4      log-normal   Estimated
## [5,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3 alp4 beta
## alp1    1    0    0    0    0
## alp2    0    0    0    0    0
## alp3    0    0    0    0    0
## alp4    0    0    0    0    0
## beta    0    0    0    0    1
##     No covariate in the model.
##     Initial values
##              alp1 alp2 alp3 alp4 beta
## Pop.CondInit    0  0.2  0.6    3  0.2
```

```
saemix.model.cov1<-saemixModel(model=ordinal.model,description="Ordinal categorical model",modeltype="l
                          psi0=matrix(c(0,0.2, 0.6, 3, 0.2),ncol=5,byrow=TRUE,dimnames=list(NULL,c("alp
                          transform.par=c(0,1,1,1,1),omega.init=diag(rep(1,5)), covariance.model = diag
                          covariate.model = covmodel)
```

```
##
##
## 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]
##   time<-xidep[,2]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   alp4<-psi[id,4]
##   beta<-psi[id,5]
##
##   logit1<-alp1 + beta*time
##   logit2<-logit1+alp2
##   logit3<-logit2+alp3
##   logit4<-logit3+alp4
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   pge4<-exp(logit4)/(1+exp(logit4))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==1)*pge1+(y==2)*(pge2 - pge1)+(y==3)*(pge3 - pge2)+(y==4)*(pge4 - pge3)+(y==5)*(1 - pge
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
##   Nb of parameters: 5
##       parameter names:  alp1 alp2 alp3 alp4 beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
## [4,] alp4      log-normal   Estimated
## [5,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3 alp4 beta
## alp1    1    0    0    0    0
## alp2    0    0    0    0    0
## alp3    0    0    0    0    0
## alp4    0    0    0    0    0
## beta    0    0    0    0    1
##     No covariate in the model.
##     Initial values
##              alp1 alp2 alp3 alp4 beta
## Pop.CondInit    0  0.2  0.6    3  0.2
```

```
saemix.model.cov2<-saemixModel(model=ordinal.model,description="Ordinal categorical model",modeltype="l
                              psi0=matrix(c(0,0.2, 0.6, 3, 0.2),ncol=5,byrow=TRUE,dimnames=list(NULL,c(
                              transform.par=c(0,1,1,1,1),omega.init=diag(rep(1,5)), covariance.model = 
                              covariate.model = covmodel2)
```

```
##
##
## 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]
##   time<-xidep[,2]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   alp4<-psi[id,4]
##   beta<-psi[id,5]
##
##   logit1<-alp1 + beta*time
##   logit2<-logit1+alp2
##   logit3<-logit2+alp3
##   logit4<-logit3+alp4
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   pge4<-exp(logit4)/(1+exp(logit4))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==1)*pge1+(y==2)*(pge2 - pge1)+(y==3)*(pge3 - pge2)+(y==4)*(pge4 - pge3)+(y==5)*(1 - pge
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
##   Nb of parameters: 5
##       parameter names:  alp1 alp2 alp3 alp4 beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
## [4,] alp4      log-normal   Estimated
## [5,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3 alp4 beta
## alp1    1    0    0    0    0
## alp2    0    0    0    0    0
## alp3    0    0    0    0    0
## alp4    0    0    0    0    0
## beta    0    0    0    0    1
##     No covariate in the model.
##     Initial values
##              alp1 alp2 alp3 alp4 beta
## Pop.CondInit    0  0.2  0.6    3  0.2
```

```
saemix.model.cov3<-saemixModel(model=ordinal.model,description="Ordinal categorical model",modeltype="l
                               psi0=matrix(c(0,0.2, 0.6, 3, 0.2),ncol=5,byrow=TRUE,dimnames=list(NULL,c
                               transform.par=c(0,1,1,1,1),omega.init=diag(rep(1,5)), covariance.model =
                               covariate.model = covmodel3)
```

```
##
##
## 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]
##   time<-xidep[,2]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   alp4<-psi[id,4]
##   beta<-psi[id,5]
##
##   logit1<-alp1 + beta*time
##   logit2<-logit1+alp2
##   logit3<-logit2+alp3
##   logit4<-logit3+alp4
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   pge4<-exp(logit4)/(1+exp(logit4))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==1)*pge1+(y==2)*(pge2 - pge1)+(y==3)*(pge3 - pge2)+(y==4)*(pge4 - pge3)+(y==5)*(1 - pge
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
##   Nb of parameters: 5
##       parameter names:  alp1 alp2 alp3 alp4 beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
## [4,] alp4      log-normal   Estimated
## [5,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3 alp4 beta
## alp1    1    0    0    0    0
## alp2    0    0    0    0    0
## alp3    0    0    0    0    0
## alp4    0    0    0    0    0
## beta    0    0    0    0    1
##   Covariate model:
##      alp1 alp2 alp3 alp4 beta
## [1,]    1    0    0    0    0
## [2,]    0    0    0    0    0
## [3,]    0    0    0    0    0
##     Initial values
##              alp1 alp2 alp3 alp4 beta
## Pop.CondInit    0  0.2  0.6    3  0.2
## Cov.CondInit    0  0.0  0.0    0  0.0
```

```
saemix.options<-list(seed=632545,save=FALSE,save.graphs=FALSE, nb.chains=5, displayProgress=FALSE)

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

```
## ind.fix10= 2 3 4 ind.fix11= 1 5 ind.fix1= 1 2 3 4 5 ind.fix0=
## -7.091797 0.9158962 1.478287 1.911037 -0.8120353
## 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 knee.saemix
##     Structured data: y ~ y + time | id
##     X variable for graphs: time ()
##     covariates: Age (-), Sex (-), treatment (-)
##       reference class for covariate Sex :  0
##       reference class for covariate treatment :  0
## Dataset characteristics:
##     number of subjects:     127
##     number of observations: 508
##     average/min/max nb obs: 4.00  /  4  /  4
## First 10 lines of data:
##    id y time y.1 Age Sex treatment mdv cens occ ytype
## 1   1 4    0   4  -2   1         0   0    0   1     1
## 2   1 4    3   4  -2   1         0   0    0   1     1
## 3   1 4    7   4  -2   1         0   0    0   1     1
## 4   1 4   10   4  -2   1         0   0    0   1     1
## 5   2 4    0   4   2   1         0   0    0   1     1
## 6   2 4    3   4   2   1         0   0    0   1     1
## 7   2 4    7   4   2   1         0   0    0   1     1
## 8   2 4   10   4   2   1         0   0    0   1     1
## 9   3 3    0   3  11   1         0   0    0   1     1
## 10  3 3    3   3  11   1         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]
##   time<-xidep[,2]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   alp4<-psi[id,4]
##   beta<-psi[id,5]
##
##   logit1<-alp1 + beta*time
##   logit2<-logit1+alp2
##   logit3<-logit2+alp3
##   logit4<-logit3+alp4
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   pge4<-exp(logit4)/(1+exp(logit4))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==1)*pge1+(y==2)*(pge2 - pge1)+(y==3)*(pge3 - pge2)+(y==4)*(pge4 - pge3)+(y==5)*(1 - pge
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
## <bytecode: 0x56239fe4c218>
##   Nb of parameters: 5
##       parameter names:  alp1 alp2 alp3 alp4 beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
## [4,] alp4      log-normal   Estimated
## [5,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3 alp4 beta
## alp1    1    0    0    0    0
## alp2    0    0    0    0    0
## alp3    0    0    0    0    0
## alp4    0    0    0    0    0
## beta    0    0    0    0    1
##     No covariate in the model.
##     Initial values
##             alp1 alp2 alp3 alp4 beta
## Pop.CondInit    0  0.2  0.6    3  0.2
## ----------------------------------
## ----    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(%)
## [1,] alp1      -11.22   1.64 15
## [2,] alp2        4.67   1.57 34
## [3,] alp3        6.49   1.39 21
## [4,] alp4        9.49   2.71 29
## [5,] beta        0.64   0.14 22
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##      Parameter   Estimate SE CV(%)
## alp1 omega2.alp1 107.58   NA NA
## beta omega2.beta   0.54   NA NA
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##             omega2.alp1 omega2.beta
## omega2.alp1 1           0
## omega2.beta 0           1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 5968.801
##       AIC = 5984.801
##       BIC = 6007.554
##
## Likelihood computed by importance sampling
##       -2LL= 869.1684
##       AIC = 885.1684
##       BIC = 907.9219
## ----------------------------------------------------
```

```
ord.fit.cov1<-saemix(saemix.model.cov1,saemix.data,saemix.options)
```

```
## ind.fix10= 2 3 4 ind.fix11= 1 5 ind.fix1= 1 2 3 4 5 ind.fix0=
## -7.091797 0.9158962 1.478287 1.911037 -0.8120353
## 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 knee.saemix
##     Structured data: y ~ y + time | id
##     X variable for graphs: time ()
##     covariates: Age (-), Sex (-), treatment (-)
##       reference class for covariate Sex :  0
##       reference class for covariate treatment :  0
## Dataset characteristics:
##     number of subjects:     127
##     number of observations: 508
##     average/min/max nb obs: 4.00  /  4  /  4
## First 10 lines of data:
##    id y time y.1 Age Sex treatment mdv cens occ ytype
## 1   1 4    0   4  -2   1         0   0    0   1     1
## 2   1 4    3   4  -2   1         0   0    0   1     1
## 3   1 4    7   4  -2   1         0   0    0   1     1
## 4   1 4   10   4  -2   1         0   0    0   1     1
## 5   2 4    0   4   2   1         0   0    0   1     1
## 6   2 4    3   4   2   1         0   0    0   1     1
## 7   2 4    7   4   2   1         0   0    0   1     1
## 8   2 4   10   4   2   1         0   0    0   1     1
## 9   3 3    0   3  11   1         0   0    0   1     1
## 10  3 3    3   3  11   1         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]
##   time<-xidep[,2]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   alp4<-psi[id,4]
##   beta<-psi[id,5]
##
##   logit1<-alp1 + beta*time
##   logit2<-logit1+alp2
##   logit3<-logit2+alp3
##   logit4<-logit3+alp4
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   pge4<-exp(logit4)/(1+exp(logit4))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==1)*pge1+(y==2)*(pge2 - pge1)+(y==3)*(pge3 - pge2)+(y==4)*(pge4 - pge3)+(y==5)*(1 - pge
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
## <bytecode: 0x56239fe4c218>
##   Nb of parameters: 5
##       parameter names:  alp1 alp2 alp3 alp4 beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
## [4,] alp4      log-normal   Estimated
## [5,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3 alp4 beta
## alp1    1    0    0    0    0
## alp2    0    0    0    0    0
## alp3    0    0    0    0    0
## alp4    0    0    0    0    0
## beta    0    0    0    0    1
##     No covariate in the model.
##     Initial values
##              alp1 alp2 alp3 alp4 beta
## Pop.CondInit    0  0.2  0.6    3  0.2
## -----------------------------------
## ----    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(%)
## [1,] alp1      -11.22   1.64 15
## [2,] alp2        4.67   1.57 34
## [3,] alp3        6.49   1.39 21
## [4,] alp4        9.49   2.71 29
## [5,] beta        0.64   0.14 22
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##      Parameter   Estimate SE CV(%)
## alp1 omega2.alp1 107.58   NA NA
## beta omega2.beta   0.54   NA NA
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##             omega2.alp1 omega2.beta
## omega2.alp1 1           0
## omega2.beta 0           1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 5968.801
##       AIC = 5984.801
##       BIC = 6007.554
##
## Likelihood computed by importance sampling
##       -2LL= 869.1684
##       AIC = 885.1684
##       BIC = 907.9219
## ----------------------------------------------------
```

```
ord.fit.cov2<-saemix(saemix.model.cov2,saemix.data,saemix.options)
```

```
## ind.fix10= 2 3 4 ind.fix11= 1 5 ind.fix1= 1 2 3 4 5 ind.fix0=
## -7.091797 0.9158962 1.478287 1.911037 -0.8120353
## 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 knee.saemix
##     Structured data: y ~ y + time | id
##     X variable for graphs: time ()
##     covariates: Age (-), Sex (-), treatment (-)
##       reference class for covariate Sex :  0
##       reference class for covariate treatment :  0
## Dataset characteristics:
##     number of subjects:     127
##     number of observations: 508
##     average/min/max nb obs: 4.00  /  4  /  4
## First 10 lines of data:
##    id y time y.1 Age Sex treatment mdv cens occ ytype
## 1   1 4    0   4  -2   1         0   0    0   1     1
## 2   1 4    3   4  -2   1         0   0    0   1     1
## 3   1 4    7   4  -2   1         0   0    0   1     1
## 4   1 4   10   4  -2   1         0   0    0   1     1
## 5   2 4    0   4   2   1         0   0    0   1     1
## 6   2 4    3   4   2   1         0   0    0   1     1
## 7   2 4    7   4   2   1         0   0    0   1     1
## 8   2 4   10   4   2   1         0   0    0   1     1
## 9   3 3    0   3  11   1         0   0    0   1     1
## 10  3 3    3   3  11   1         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]
##   time<-xidep[,2]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   alp4<-psi[id,4]
##   beta<-psi[id,5]
##
##   logit1<-alp1 + beta*time
##   logit2<-logit1+alp2
##   logit3<-logit2+alp3
##   logit4<-logit3+alp4
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   pge4<-exp(logit4)/(1+exp(logit4))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==1)*pge1+(y==2)*(pge2 - pge1)+(y==3)*(pge3 - pge2)+(y==4)*(pge4 - pge3)+(y==5)*(1 - pge
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
## <bytecode: 0x56239fe4c218>
##   Nb of parameters: 5
##       parameter names:  alp1 alp2 alp3 alp4 beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
## [4,] alp4      log-normal   Estimated
## [5,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3 alp4 beta
## alp1    1    0    0    0    0
## alp2    0    0    0    0    0
## alp3    0    0    0    0    0
## alp4    0    0    0    0    0
## beta    0    0    0    0    1
##     No covariate in the model.
##     Initial values
##             alp1 alp2 alp3 alp4 beta
## Pop.CondInit    0  0.2  0.6    3  0.2
## ----------------------------------
## ----    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(%)
## [1,] alp1      -11.22   1.64 15
## [2,] alp2        4.67   1.57 34
## [3,] alp3        6.49   1.39 21
## [4,] alp4        9.49   2.71 29
## [5,] beta        0.64   0.14 22
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##      Parameter   Estimate SE CV(%)
## alp1 omega2.alp1 107.58   NA NA
## beta omega2.beta   0.54   NA NA
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##             omega2.alp1 omega2.beta
## omega2.alp1 1           0
## omega2.beta 0           1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 5968.801
##       AIC = 5984.801
##       BIC = 6007.554
##
## Likelihood computed by importance sampling
##       -2LL= 869.1684
##       AIC = 885.1684
##       BIC = 907.9219
## ----------------------------------------------------
```

```
ord.fit.cov3<-saemix(saemix.model.cov3,saemix.data,saemix.options)
```

```
## ind.fix10= 3 4 5 ind.fix11= 1 2 6 ind.fix1= 1 2 3 4 5 6 ind.fix0=
## -7.267067 0.133898 0.9609994 1.506956 1.910955 -0.8007291
## 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 knee.saemix
##     Structured data: y ~ y + time | id
##     X variable for graphs: time ()
##     covariates: Age (-), Sex (-), treatment (-)
##       reference class for covariate Sex :  0
##       reference class for covariate treatment :  0
## Dataset characteristics:
##     number of subjects:     127
##     number of observations: 508
##     average/min/max nb obs: 4.00  /  4  /  4
## First 10 lines of data:
##    id y time y.1 Age Sex treatment mdv cens occ ytype
## 1   1 4    0   4  -2   1         0   0    0   1     1
## 2   1 4    3   4  -2   1         0   0    0   1     1
## 3   1 4    7   4  -2   1         0   0    0   1     1
## 4   1 4   10   4  -2   1         0   0    0   1     1
## 5   2 4    0   4   2   1         0   0    0   1     1
## 6   2 4    3   4   2   1         0   0    0   1     1
## 7   2 4    7   4   2   1         0   0    0   1     1
## 8   2 4   10   4   2   1        0   0    0   1     1
## 9   3 3    0   3  11   1        0   0    0   1     1
## 10  3 3    3   3  11   1        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]
##   time<-xidep[,2]
##   alp1<-psi[id,1]
##   alp2<-psi[id,2]
##   alp3<-psi[id,3]
##   alp4<-psi[id,4]
##   beta<-psi[id,5]
##
##   logit1<-alp1 + beta*time
##   logit2<-logit1+alp2
##   logit3<-logit2+alp3
##   logit4<-logit3+alp4
##   pge1<-exp(logit1)/(1+exp(logit1))
##   pge2<-exp(logit2)/(1+exp(logit2))
##   pge3<-exp(logit3)/(1+exp(logit3))
##   pge4<-exp(logit4)/(1+exp(logit4))
##   logpdf<-rep(0,length(y))
##   P.obs = (y==1)*pge1+(y==2)*(pge2 - pge1)+(y==3)*(pge3 - pge2)+(y==4)*(pge4 - pge3)+(y==5)*(1 - pge
##   logpdf <- log(P.obs)
##
##   return(logpdf)
## }
## <bytecode: 0x56239fe4c218>
##   Nb of parameters: 5
##       parameter names:  alp1 alp2 alp3 alp4 beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alp1      normal       Estimated
## [2,] alp2      log-normal   Estimated
## [3,] alp3      log-normal   Estimated
## [4,] alp4      log-normal   Estimated
## [5,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##      alp1 alp2 alp3 alp4 beta
## alp1    1    0    0    0    0
## alp2    0    0    0    0    0
## alp3    0    0    0    0    0
## alp4    0    0    0    0    0
## beta    0    0    0    0    1
##   Covariate model:
##     [,1] [,2] [,3] [,4] [,5]
## Age    1    0    0    0    0
##     Initial values
##              alp1 alp2 alp3 alp4 beta
## Pop.CondInit    0  0.2  0.6    3  0.2
## Cov.CondInit    0  0.0  0.0    0  0.0
## -----------------------------------
## ----    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,] alp1           -10.96   1.698 15    -
## [2,] beta_Age(alp1)   0.19   0.084 45    0.013
## [3,] alp2             4.72   1.622 34    -
## [4,] alp3             6.39   1.372 21    -
## [5,] alp4             9.10   2.585 28    -
## [6,] beta             0.62   0.135 22    -
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##      Parameter   Estimate SE CV(%)
## alp1 omega2.alp1 99.28    NA NA
## beta omega2.beta  0.54    NA NA
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##             omega2.alp1 omega2.beta
## omega2.alp1 1           0
## omega2.beta 0           1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 5955.436
##       AIC = 5973.436
##       BIC = 5999.034
##
## Likelihood computed by importance sampling
##       -2LL= 866.072
##       AIC = 884.072
##       BIC = 909.6697
## ----------------------------------------------------
```

```
# Comparing the 3 covariate models - model with Age on alp1 and treatment on beta best
compare.saemix(ord.fit.cov1,ord.fit.cov2, ord.fit.cov3)
```

```
## Likelihoods calculated by importance sampling

##        AIC      BIC  BIC.cov
## 1 885.1684 907.9219 897.5482
## 2 885.1684 907.9219 897.5482
## 3 884.0720 909.6697 899.2960
```

Count data model

- Vraies données ??? (difficile à trouver :-/)
  - contacté David Atkins (tutorial in 2013 on analysing count data with GLMM and GEE): dataset on gender differences in drinking patterns that would be great to use as an example in saemix ⇒ accepted ! lovely :-)
  - Salamanders data from the glmmTMB package
    * fit successful when using only the data for one species
    * but error when using more than one species with a recurrent error message (solve.default…) **TODO** investigate
    * note: error in the previous version of Poisson model (factorial(y) instead of log(factorial(y))) ?
- Epilepsy
- Drinking patterns

```
epilepsy<-MASS::epil
saemix.data<-saemixData(name.data=epilepsy, name.group=c("subject"),
                        name.predictors=c("period","y"),name.response=c("y"),
                        name.covariates=c("trt","base", "age"),
                        units=list(x="2-week",y="",covariates=c("","","yr")))
```

```
## [1] "trt"  "base" "age"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset epilepsy
##     Structured data: y ~ period + y | subject
##     X variable for graphs: period (2-week)
##     covariates: trt (), base (), age (yr)
##       reference class for covariate trt :  placebo
```

```
## Poisson model with one parameter
countmodel.poisson<-function(psi,id,xidep) {
  y<-xidep[,2]
  lambda<-psi[id,1]
  logp <- -lambda + y*log(lambda) - log(factorial(y))
  return(logp)
}

# Adding a period effect
countmodel.periodpoi<-function(psi,id,xidep) {
  tim <- xidep[,1]
  y<-xidep[,2]
  lam<-psi[id,1]
  betaT<-psi[id,2]
  lambda<-lam*exp(beta*log(tim))
  logp <- -lambda + y*log(lambda) - log(factorial(y))
  return(logp)
}

## Generalised Poisson model
countmodel.genpoisson<-function(psi,id,xidep) {
  y<-xidep[,1]
  delta<-psi[id,1]
  lambda<-psi[id,2]
  logp <- -lambda
  pos.ind <- which(y>0)
  logp[pos.ind] <- log(lambda) + (y-1)*log(lambda+y*delta) - (lambda+y*delta) - log(factorial(y))
  return(logp)
}

## Poisson model wtih Zero-Inflation
countmodel.zip<-function(psi,id,xidep) {
  y<-xidep[,2]
  lambda<-psi[id,1]
  p0<-psi[id,2]
  logp <- log(1-p0) -lambda + y*log(lambda) - log(factorial(y))
  logp0 <- log(p0+(1-p0)*exp(-lambda))
  logp[y==0]<-logp0[y==0]
  return(logp)
}

saemix.model.poi<-saemixModel(model=countmodel.poisson,description="count model Poisson",modeltype="lik
                          psi0=matrix(c(0.5),ncol=1,byrow=TRUE,dimnames=list(NULL, c("lambda"))),
                          transform.par=c(1))
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  count model Poisson  Model type:  likelihood
## function(psi,id,xidep) {
##   y<-xidep[,2]
##   lambda<-psi[id,1]
##   logp <- -lambda + y*log(lambda) - log(factorial(y))
##   return(logp)
## }
##   Nb of parameters: 1
##       parameter names:  lambda
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
##   Variance-covariance matrix:
##        lambda
## lambda      1
##     No covariate in the model.
##     Initial values
##               lambda
## Pop.CondInit    0.5
```

```
saemix.model.zip<-saemixModel(model=countmodel.zip,description="count model ZIP",modeltype="likelihood"
                              psi0=matrix(c(0.5,0.2),ncol=2,byrow=TRUE,dimnames=list(NULL, c("lambda","
                              transform.par=c(1,3), #omega.init=matrix(c(0.5,0,0,0.3),ncol=2,byrow=TRUE
                              covariance.model=matrix(c(1,0,0,0),ncol=2,byrow=TRUE))
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  count model ZIP  Model type:  likelihood
## function(psi,id,xidep) {
##   y<-xidep[,2]
##   lambda<-psi[id,1]
##   p0<-psi[id,2]
##   logp <- log(1-p0) -lambda + y*log(lambda) - log(factorial(y))
##   logp0 <- log(p0+(1-p0)*exp(-lambda))
##   logp[y==0]<-logp0[y==0]
##   return(logp)
## }
##   Nb of parameters: 2
##       parameter names:  lambda p0
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] p0        logit        Estimated
##   Variance-covariance matrix:
##        lambda p0
## lambda      1  0
## p0          0  0
##     No covariate in the model.
##     Initial values
##               lambda  p0
## Pop.CondInit    0.5 0.2
```

```
saemix.model.gp<-saemixModel(model=countmodel.zip,description="Generalised Poisson model",modeltype="li
                              psi0=matrix(c(0.5,0.2),ncol=2,byrow=TRUE,dimnames=list(NULL, c("delta","l
                              transform.par=c(1,1), #omega.init=matrix(c(0.5,0,0,0.3),ncol=2,byrow=TRUE
                              covariance.model=matrix(c(1,0,0,0),ncol=2,byrow=TRUE))
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Generalised Poisson model  Model type:  likelihood
## function(psi,id,xidep) {
##   y<-xidep[,2]
##   lambda<-psi[id,1]
##   p0<-psi[id,2]
##   logp <- log(1-p0) -lambda + y*log(lambda) - log(factorial(y))
##   logp0 <- log(p0+(1-p0)*exp(-lambda))
##   logp[y==0]<-logp0[y==0]
##   return(logp)
## }
##   Nb of parameters: 2
##       parameter names:  delta lambda
##       distribution:
##      Parameter Distribution Estimated
## [1,] delta     log-normal   Estimated
## [2,] lambda    log-normal   Estimated
##   Variance-covariance matrix:
##        delta lambda
## delta      1      0
## lambda     0      0
##     No covariate in the model.
##     Initial values
##              delta lambda
## Pop.CondInit   0.5    0.2
```

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

poisson.fit<-saemix(saemix.model.poi,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 epilepsy
##     Structured data: y ~ period + y | subject
##     X variable for graphs: period (2-week)
##     covariates: trt (), base (), age (yr)
##       reference class for covariate trt :  placebo
## Dataset characteristics:
##     number of subjects:     59
##     number of observations: 236
##     average/min/max nb obs: 4.00  /  4  /  4
## First 10 lines of data:
##    subject period y y.1     trt base age mdv cens occ ytype
## 1        1      1 5   5 placebo   11  31   0    0   1     1
## 2        1      2 3   3 placebo   11  31   0    0   1     1
## 3        1      3 3   3 placebo   11  31   0    0   1     1
## 4        1      4 3   3 placebo   11  31   0    0   1     1
## 5        2      1 3   3 placebo   11  30   0    0   1     1
## 6        2      2 5   5 placebo   11  30   0    0   1     1
## 7        2      3 3   3 placebo   11  30   0    0   1     1
## 8        2      4 3   3 placebo   11  30   0    0   1     1
## 9        3      1 2   2 placebo    6  25   0    0   1     1
## 10       3      2 4   4 placebo    6  25   0    0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  count model Poisson  Model type:  likelihood
## function(psi,id,xidep) {
##   y<-xidep[,2]
##   lambda<-psi[id,1]
##   logp <- -lambda + y*log(lambda) - log(factorial(y))
##   return(logp)
## }
## <bytecode: 0x56239cff3490>
##   Nb of parameters: 1
##       parameter names:  lambda
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
##   Variance-covariance matrix:
##        lambda
## lambda      1
##     No covariate in the model.
##     Initial values
##             lambda
## Pop.CondInit    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,] lambda    5.1      0.71 14
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##        Parameter     Estimate SE   CV(%)
## lambda omega2.lambda 0.9      0.21 23
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##               omega2.lambda
## omega2.lambda 1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 60096.92
##       AIC = 60102.92
##       BIC = 60109.15
##
## Likelihood computed by importance sampling
##       -2LL= 1402.095
##       AIC = 1408.095
##       BIC = 1414.327
## ----------------------------------------------------
```

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

```
## ind.fix10= 2 ind.fix11= 1 ind.fix1= 1 2 ind.fix0=
## 1.659812 -3.239705
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data              ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset epilepsy
##     Structured data: y ~ period + y | subject
##     X variable for graphs: period (2-week)
##     covariates: trt (), base (), age (yr)
##       reference class for covariate trt :  placebo
## Dataset characteristics:
##     number of subjects:     59
##     number of observations: 236
##     average/min/max nb obs: 4.00  /  4  /  4
## First 10 lines of data:
##    subject period y y.1     trt base age mdv cens occ ytype
## 1        1      1 5   5 placebo   11  31   0    0   1     1
## 2        1      2 3   3 placebo   11  31   0    0   1     1
## 3        1      3 3   3 placebo   11  31   0    0   1     1
## 4        1      4 3   3 placebo   11  31   0    0   1     1
## 5        2      1 3   3 placebo   11  30   0    0   1     1
## 6        2      2 5   5 placebo   11  30   0    0   1     1
## 7        2      3 3   3 placebo   11  30   0    0   1     1
## 8        2      4 3   3 placebo   11  30   0    0   1     1
## 9        3      1 2   2 placebo    6  25   0    0   1     1
## 10       3      2 4   4 placebo    6  25   0    0   1     1
## -----------------------------------
## ----          Model             ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  Generalised Poisson model  Model type:  likelihood
## function(psi,id,xidep) {
##   y<-xidep[,2]
##   lambda<-psi[id,1]
##   p0<-psi[id,2]
##   logp <- log(1-p0) -lambda + y*log(lambda) - log(factorial(y))
##   logp0 <- log(p0+(1-p0)*exp(-lambda))
##   logp[y==0]<-logp0[y==0]
##   return(logp)
## }
## <bytecode: 0x56239c93ddd8>
##   Nb of parameters: 2
##       parameter names:  delta lambda
##       distribution:
##      Parameter Distribution Estimated
## [1,] delta     log-normal   Estimated
## [2,] lambda    log-normal   Estimated
##   Variance-covariance matrix:
##        delta lambda
## delta      1      0
## lambda     0      0
##     No covariate in the model.
##     Initial values
##              delta lambda
## Pop.CondInit   0.5    0.2
## ----------------------------------
## ----    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,] delta     5.314    0.747 14
## [2,] lambda    0.041    0.024 58
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##       Parameter    Estimate SE   CV(%)
## delta omega2.delta 0.86     0.21 24
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##              omega2.delta
## omega2.delta 1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 60647.88
##       AIC = 60655.88
##       BIC = 60664.19
##
## Likelihood computed by importance sampling
##       -2LL= 1381.329
##       AIC = 1389.329
##       BIC = 1397.639
## ----------------------------------------------------
```

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

```
## ind.fix10= 2 ind.fix11= 1 ind.fix1= 1 2 ind.fix0=
## 1.59917 -3.019205
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data             ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset epilepsy
##     Structured data: y ~ period + y | subject
##     X variable for graphs: period (2-week)
##     covariates: trt (), base (), age (yr)
##       reference class for covariate trt :  placebo
## Dataset characteristics:
##     number of subjects:     59
##     number of observations: 236
##     average/min/max nb obs: 4.00  /  4  /  4
## First 10 lines of data:
##    subject period y y.1     trt base age mdv cens occ ytype
## 1        1      1 5   5 placebo   11  31   0    0   1     1
## 2        1      2 3   3 placebo   11  31   0    0   1     1
## 3        1      3 3   3 placebo   11  31   0    0   1     1
## 4        1      4 3   3 placebo   11  31   0    0   1     1
## 5        2      1 3   3 placebo   11  30   0    0   1     1
## 6        2      2 5   5 placebo   11  30   0    0   1     1
## 7        2      3 3   3 placebo   11  30   0    0   1     1
## 8        2      4 3   3 placebo   11  30   0    0   1     1
## 9        3      1 2   2 placebo    6  25   0    0   1     1
## 10       3      2 4   4 placebo    6  25   0    0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  count model ZIP  Model type:  likelihood
## function(psi,id,xidep) {
##   y<-xidep[,2]
##   lambda<-psi[id,1]
##   p0<-psi[id,2]
##   logp <- log(1-p0) -lambda + y*log(lambda) - log(factorial(y))
##   logp0 <- log(p0+(1-p0)*exp(-lambda))
##   logp[y==0]<-logp0[y==0]
##   return(logp)
## }
## <bytecode: 0x56239c93ddd8>
##   Nb of parameters: 2
##       parameter names:  lambda p0
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] p0        logit        Estimated
##   Variance-covariance matrix:
##        lambda p0
## lambda      1  0
## p0          0  0
##     No covariate in the model.
##     Initial values
##              lambda  p0
## Pop.CondInit    0.5 0.2
## ----------------------------------
## ----    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,] lambda    5.320    0.748 14
## [2,] p0        0.041    0.024 58
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##        Parameter     Estimate SE   CV(%)
## lambda omega2.lambda 0.86     0.21 24
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##               omega2.lambda
## omega2.lambda 1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 61045.94
##       AIC = 61053.94
##       BIC = 61062.25
##
## Likelihood computed by importance sampling
##       -2LL= 1381.314
##       AIC = 1389.314
##       BIC = 1397.624
## ----------------------------------------------------
```

- Meantimes, simulated data

```
# Settings
param <- c(39.1, 0.0388, 0.1 )
omega<-c(0.5, 0.5) # SD=50%
paramSimul<-c(param, omega)
parnam<-c("alpha","beta","risk","omega.alpha","omega.beta")

nsuj<-40
xtim<-c(0.0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100)

partab<-as.data.frame(matrix(data=0,nrow=nsuj,ncol=2,dimnames=list(NULL,parnam[1:2])))
for(i in 1:2) partab[,i]<-rnorm(nsuj,mean=log(param[i]),sd=omega[i])
partab[(1+nsuj/2):nsuj,2]<-partab[(1+nsuj/2):nsuj,2]+param[3]
for(i in 1:2) partab[,i]<-exp(partab[,i])

psim<-data.frame()
for(itim in xtim) {
  lambda<-partab[,1]*exp(-partab[,2]*itim)
  psim<-rbind(psim,lambda)
}
datsim<-data.frame(id=rep(1:nsuj,each=length(xtim)),time=rep(xtim,nsuj),lambda=unlist(psim))
rownames(datsim)<-NULL
ysim<-rpois(dim(datsim)[1], lambda=datsim$lambda)
#  summary(datsim)
datsim$y<-ysim
datsim$risk<-ifelse(datsim$id>(nsuj/2),1,0)
```

```
saemix.data<-saemixData(name.data=datsim,name.group=c("id"),name.predictors=c("time","y"), name.covaria
```

```
## 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
```

```r
# Model
countData.model<-function(psi,id,xidep) {
  tim <- xidep[,1]
  y <- xidep[,2]
  alpha <- psi[id,1]
  beta <- psi[id,2]
  lambda <- alpha*exp(-beta*tim)

  logpdf <- rep(0,length(tim))
  logpdf <- -lambda + y*( (log(alpha) - beta*tim )) - log(factorial(y))
  return(logpdf)
}
saemix.model.true<-saemixModel(model=countData.model,description="Count data model", modeltype="likelih
                               psi0=matrix(c(param[1:2],0,param[3]),ncol=2,byrow=TRUE,dimnames=list(NUL
                               covariate.model=matrix(c(0,1),ncol=2), omega.init = diag(c(0.5,0.5)),
                               transform.par=c(1,1),covariance.model=matrix(c(1,0,0,1),ncol=2))
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Count data model  Model type:  likelihood
## function(psi,id,xidep) {
##   tim <- xidep[,1]
##   y <- xidep[,2]
##   alpha <- psi[id,1]
##   beta <- psi[id,2]
##   lambda <- alpha*exp(-beta*tim)
##
##   logpdf <- rep(0,length(tim))
##   logpdf <- -lambda + y*( (log(alpha) - beta*tim )) - log(factorial(y))
##   return(logpdf)
## }
##   Nb of parameters: 2
##       parameter names:  alpha beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alpha     log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##       alpha beta
## alpha     1    0
## beta      0    1
##   Covariate model:
##      alpha beta
## [1,]     0    1
##     Initial values
##              alpha   beta
## Pop.CondInit  39.1 0.0388
## Cov.CondInit   0.0 0.1000
```

```r
# Running saemix

saemix.options<-list(seed=123456,save=FALSE,save.graphs=FALSE, fim=FALSE, displayProgress=FALSE)
count.fit<-try(saemix(saemix.model.true,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 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:     40
##     number of observations: 840
##     average/min/max nb obs: 21.00  /  21  /  21
## First 10 lines of data:
##    id time  y y.1 risk mdv cens occ ytype
## 1   1    0 40  40    0   0    0   1     1
## 2   1    5 24  24    0   0    0   1     1
## 3   1   10 16  16    0   0    0   1     1
## 4   1   15 18  18    0   0    0   1     1
## 5   1   20 10  10    0   0    0   1     1
## 6   1   25  7   7    0   0    0   1     1
## 7   1   30  1   1    0   0    0   1     1
## 8   1   35  3   3    0   0    0   1     1
## 9   1   40  5   5    0   0    0   1     1
## 10  1   45  1   1    0   0    0   1     1
## -----------------------------------
## ----          Model             ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  Count data model  Model type:  likelihood
## function(psi,id,xidep) {
##   tim <- xidep[,1]
##   y <- xidep[,2]
##   alpha <- psi[id,1]
##   beta <- psi[id,2]
##   lambda <- alpha*exp(-beta*tim)
##
##   logpdf <- rep(0,length(tim))
##   logpdf <- -lambda + y*( (log(alpha) - beta*tim )) - log(factorial(y))
##   return(logpdf)
## }
## <bytecode: 0x56239d922a20>
##   Nb of parameters: 2
##       parameter names:  alpha beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] alpha     log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##       alpha beta
## alpha     1    0
## beta      0    1
##   Covariate model:
##      [,1] [,2]
## risk    0    1
##     Initial values
##              alpha   beta
## Pop.CondInit  39.1 0.0388
## Cov.CondInit   0.0 0.1000
## ----------------------------------
## ----    Key algorithm options  ----
## ----------------------------------
##     Estimation of individual parameters (MAP)
##     Estimation of log-likelihood by importance sampling
##     Number of iterations:  K1=300, K2=100
##     Number of chains:  2
##     Seed:  123456
##     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
## [1,] alpha           35.716
## [2,] beta             0.047
## [3,] beta_risk(beta) -0.064
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##       Parameter    Estimate
## alpha omega2.alpha 0.21
## beta  omega2.beta  0.22
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##              omega2.alpha omega2.beta
## omega2.alpha 1            0
## omega2.beta  0            1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
##
## Likelihood computed by importance sampling
##       -2LL= 3590.901
##       AIC = 3602.901
##       BIC = 3613.034
## ----------------------------------------------------
```

Time-to-event

TTE model - simulated data

TTE data simulated according to a Weibull model, hazard defined by shape (β) and scale (λ) as:

$$h(t) = \frac{\beta}{\lambda}\left(\frac{t}{\lambda}\right)^{\beta-1}$$

```r
# Simulating TTE data
set.seed(12345)

nsuj<-50
xtim<-c(0)
tte.data<-data.frame(id=rep(1:nsuj,each=length(xtim)),time=rep(xtim,nsuj))
psiM<-data.frame(lambda=seq(1.6,2,length.out=length(unique(tte.data$id))),beta = 2)

simul.tte<-function(psi,id,xidep) {
  T<-xidep
  N <- nrow(psi)
  Nj <- length(T)
  censoringtime = 3
  lambda <- psi[id,1]
  beta <- psi[id,2]
  obs <-rep(0,length(T))
  for (i in (1:N)){
    obs[id==i] <- rweibull(n=length(id[id==i]), shape=beta[i], scale=lambda[i])
  }
  obs[obs>censoringtime]<-censoringtime
  return(obs)
}

preds <- simul.tte(psiM, tte.data$id, tte.data[,c("time")])
tte.data$y<-0
tte.data$tlat<-preds
dat1<-tte.data[,c("id","time","y")]
dat2<-tte.data[,c("id","tlat","y")]
dat2$y<-as.integer(dat2$tlat>0 & dat2$tlat<3)
colnames(dat2)[2]<-"time"
tte.data<-rbind(dat1,dat2)
tte.data<-tte.data[order(tte.data$id, tte.data$time),]
tte.psiM<-psiM

# Simulate T from Weibull (check)
if(FALSE) {
  lambda<-2
  beta<-2
  nsim<-5000
  # By hand
  q1<-runif(nsim)
  #  tevent<-lambda*exp(log(q1)/beta)
  tevent<-lambda*exp(log(-log(q1))/beta)
  tevent<-sort(tevent)
#  plot(tevent, exp(-(tevent/lambda)^beta))
  tevent2<-sort(rweibull(nsim, shape=beta, scale=lambda))
  plot(tevent, tevent2)
  abline(0,1)

}
```

```
saemix.data<-saemixData(name.data=tte.data, name.group=c("id"),
  name.predictors=c("time"), name.response="y")
```

```
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset tte.data
##     Structured data: y ~ time | id
##     Predictor: time ()
```

```
tte.model<-function(psi,id,xidep) {
  T<-xidep[,1]
  N <- nrow(psi)
  Nj <- length(T)
  # censoringtime = 6
  censoringtime = max(T)
  lambda <- psi[id,1]
  beta <- psi[id,2]
  init <- which(T==0)
  cens <- which(T==censoringtime)
  ind <- setdiff(1:Nj, append(init,cens))
  hazard <- (beta/lambda)*(T/lambda)^(beta-1)
  H <- (T/lambda)^beta
  logpdf <- rep(0,Nj)
  logpdf[cens] <- -H[cens] + H[cens-1]
  logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind])
  return(logpdf)
}

saemix.model<-saemixModel(model=tte.model,description="time model",modeltype="likelihood",
  psi0=matrix(c(1,2),ncol=2,byrow=TRUE,dimnames=list(NULL,  c("lambda","beta"))),
  transform.par=c(1,1),covariance.model=matrix(c(1,0,0,1),ncol=2, byrow=TRUE))
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  time model  Model type:  likelihood
## function(psi,id,xidep) {
##   T<-xidep[,1]
##   N <- nrow(psi)
##   Nj <- length(T)
##   # censoringtime = 6
##   censoringtime = max(T)
##   lambda <- psi[id,1]
##   beta <- psi[id,2]
##   init <- which(T==0)
##   cens <- which(T==censoringtime)
##   ind <- setdiff(1:Nj, append(init,cens))
##   hazard <- (beta/lambda)*(T/lambda)^(beta-1)
##   H <- (T/lambda)^beta
##   logpdf <- rep(0,Nj)
##   logpdf[cens] <- -H[cens] + H[cens-1]
##   logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind])
##   return(logpdf)
## }
##   Nb of parameters: 2
##       parameter names:  lambda beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##        lambda beta
## lambda      1    0
## beta        0    1
##     No covariate in the model.
##     Initial values
##              lambda beta
## Pop.CondInit      1    2
```

```r
saemix.model<-saemixModel(model=tte.model,description="time model",modeltype="likelihood",
  psi0=matrix(c(1,2),ncol=2,byrow=TRUE,dimnames=list(NULL,  c("lambda","beta"))),
  transform.par=c(1,1),covariance.model=matrix(c(1,0,0,0),ncol=2, byrow=TRUE))
```

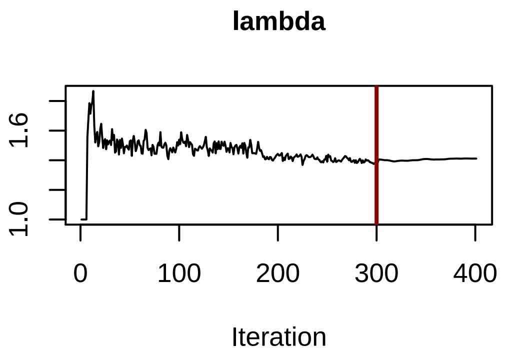
```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  time model  Model type:  likelihood
## function(psi,id,xidep) {
##   T<-xidep[,1]
##   N <- nrow(psi)
##   Nj <- length(T)
##   # censoringtime = 6
##   censoringtime = max(T)
##   lambda <- psi[id,1]
##   beta <- psi[id,2]
##   init <- which(T==0)
##   cens <- which(T==censoringtime)
##   ind <- setdiff(1:Nj, append(init,cens))
##   hazard <- (beta/lambda)*(T/lambda)^(beta-1)
##   H <- (T/lambda)^beta
##   logpdf <- rep(0,Nj)
##   logpdf[cens] <- -H[cens] + H[cens-1]
##   logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind])
##   return(logpdf)
## }
##   Nb of parameters: 2
##       parameter names:  lambda beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##        lambda beta
## lambda      1    0
## beta        0    0
##     No covariate in the model.
##     Initial values
##               lambda beta
## Pop.CondInit      1    2
```

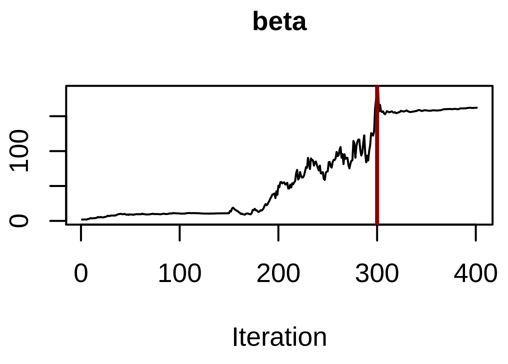
```
saemix.options<-list(seed=632545,save=FALSE,save.graphs=FALSE, displayProgress=FALSE)
tte.fit<-saemix(saemix.model,saemix.data,saemix.options)
```

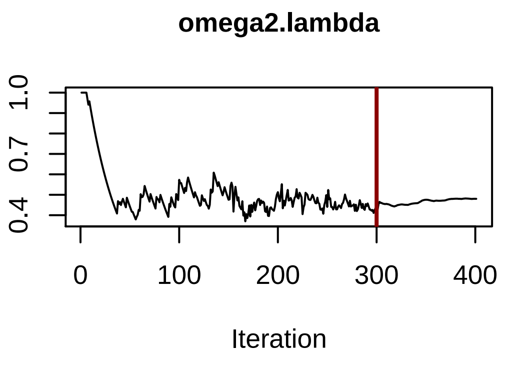
```
## ind.fix10= 2 ind.fix11= 1 ind.fix1= 1 2 ind.fix0=
## 0.3733829 2.393744
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data              ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset tte.data
##     Structured data: y ~ time | id
##     Predictor: time ()
## Dataset characteristics:
##     number of subjects:     50
##     number of observations: 100
##     average/min/max nb obs: 2.00  /  2  /  2
## First 10 lines of data:
##    id      time y mdv cens occ ytype
## 1   1 0.0000000 0   0    0   1     1
## 51  1 0.9152915 1   0    0   1     1
## 2   2 0.0000000 0   0    0   1     1
## 52  2 0.5857074 1   0    0   1     1
## 3   3 0.0000000 0   0    0   1     1
## 53  3 0.8447454 1   0    0   1     1
## 4   4 0.0000000 0   0    0   1     1
## 54  4 0.5648408 1   0    0   1     1
## 5   5 0.0000000 0   0    0   1     1
## 55  5 1.4458047 1   0    0   1     1
## -----------------------------------
## ----          Model             ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  time model  Model type:  likelihood
## function(psi,id,xidep) {
```

```
##   T<-xidep[,1]
##   N <- nrow(psi)
##   Nj <- length(T)
##   # censoringtime = 6
##   censoringtime = max(T)
##   lambda <- psi[id,1]
##   beta <- psi[id,2]
##   init <- which(T==0)
##   cens <- which(T==censoringtime)
##   ind <- setdiff(1:Nj, append(init,cens))
##   hazard <- (beta/lambda)*(T/lambda)^(beta-1)
##   H <- (T/lambda)^beta
##   logpdf <- rep(0,Nj)
##   logpdf[cens] <- -H[cens] + H[cens-1]
##   logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind])
##   return(logpdf)
## }
## <bytecode: 0x56239f9731e0>
##   Nb of parameters: 2
##       parameter names:  lambda beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##        lambda beta
## lambda      1    0
## beta        0    0
##     No covariate in the model.
##     Initial values
##              lambda beta
## Pop.CondInit      1    2
## ----------------------------------
## ----    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,] lambda      1.4       0.58   41
## [2,] beta      162.2    5675.89 3500
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##        Parameter     Estimate SE   CV(%)
## lambda omega2.lambda 0.48     0.21 44
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##               omega2.lambda
## omega2.lambda 1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 573.0104
##       AIC = 581.0104
##       BIC = 588.6585
##
## Likelihood computed by importance sampling
##       -2LL= 122.2899
##       AIC = 130.2899
##       BIC = 137.938
## ----------------------------------------------------
```

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







TTE model - lung cancer

- create dataset for saemix (once) using the lung data from the survival package
- changes
  - saemix format: added time=0
  - created one column for status (dead or alive, recoded as 1/0) and one for censoring (0/1)
  - removed subjects with missing age, institution, sex, or ph scores (ecog and karno)

```r
if(FALSE) {
  library(survival)
  data(cancer)
  cancer$cens<-as.integer(cancer$status==1) # censored=1, non-censored=0
  cancer$status<-cancer$status-1 # dead=1, alive=0
  cancer<-cbind(id=1:dim(cancer)[1],cancer)
  cancer2<-cancer
  cancer2$time<-0
  cancer2$status<-0
  cancer2$cens<-0
  lung.saemix<-rbind(cancer2, cancer)
  lung.saemix<-lung.saemix[order(lung.saemix$id, lung.saemix$time),]
  lung.saemix$sex<-lung.saemix$sex-1
  lung.saemix<-lung.saemix[,c("id","time","status","cens","inst","age", "sex", "ph.ecog", "ph.karno", "
  hasnoNA<-function(xmat)
    apply(xmat,1,function(x) sum(is.na(x))==0)
  lung.saemix<-lung.saemix[hasnoNA(lung.saemix[,5:9]),]
  write.table(lung.saemix, file.path(datDir, "lung.saemix.tab"), quote=F, row.names=F)
}
```

Checks

- The `Surv` function from the `survival` package creates a survival object for use as the response in a model formula.
  - one entry for each subject that is the survival time, which is followed by a `+` if the subject was censored
  - transform lung.saemix in the Surv format to check the survival function w/r saemix fit
- Weibull model
  - parametric survival function

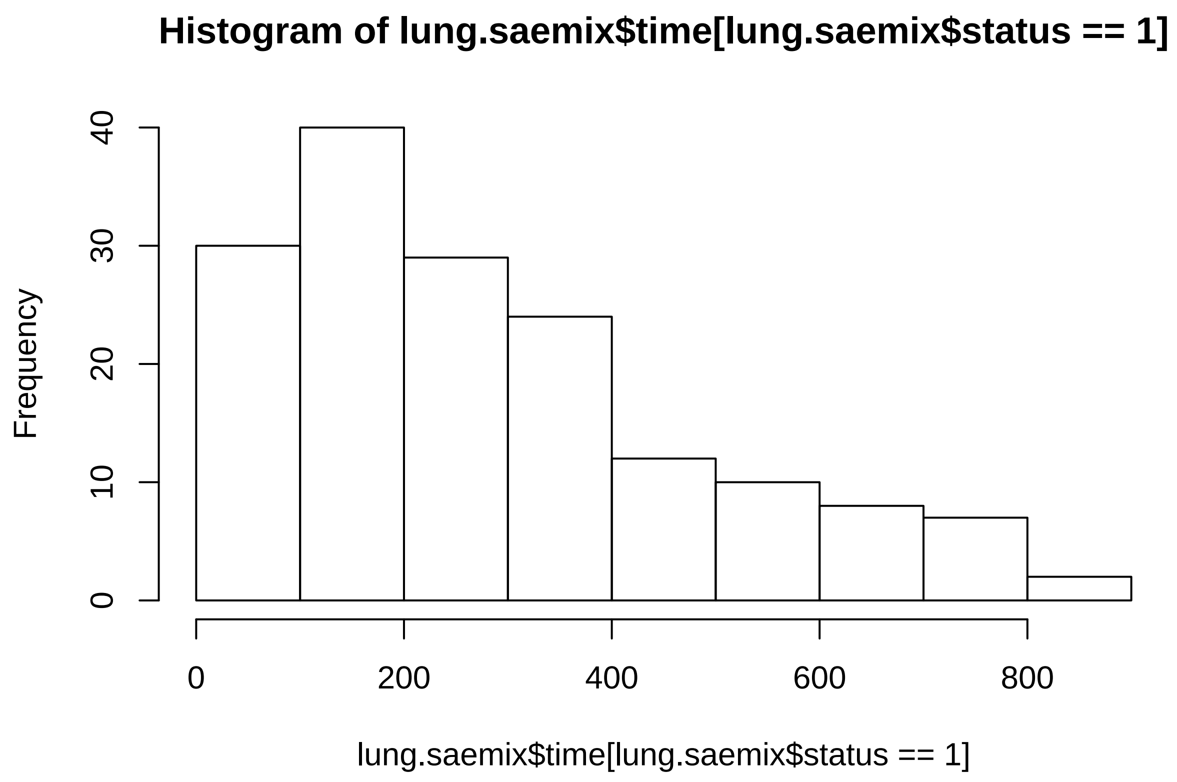
$$S(t) = e^{-\left(\frac{t}{\lambda}\right)^{\beta}}$$

- Also tried computing a SE for $S(t)$ using the delta-method where the vector of derivatives for the survival function of Weibull model are:

$$\begin{pmatrix} \frac{\delta S}{\delta \lambda} \\ \frac{\delta S}{\delta \beta} \end{pmatrix} = \begin{pmatrix} \frac{\beta}{\lambda} \left(\frac{t}{\lambda}\right)^{\beta} e^{-\left(\frac{t}{\lambda}\right)^{\beta}} \\ -\ln\left(\frac{t}{\lambda}\right) \left(\frac{t}{\lambda}\right)^{\beta} e^{-\left(\frac{t}{\lambda}\right)^{\beta}} \end{pmatrix}$$

- works pretty well compared to the non-parametric KM estimate

```r
if(testMode)
  data(lung.saemix) else
    lung.saemix<-read.table(file.path(datDir, "lung.saemix.tab"), header=TRUE)

hist(lung.saemix$time[lung.saemix$status==1])
```



```r
# Note: missing data in pat.karno, wt.loss and meal.cal
if(FALSE)
    print(summary(lung.saemix))

saemix.data<-saemixData(name.data=lung.saemix,header=TRUE,name.group=c("id"),
      name.predictors=c("time","status","cens"),name.response=c("status"),
      name.covariates=c("age", "sex", "ph.ecog", "ph.karno", "pat.karno", "wt.loss","meal.cal"),
      units=list(x="days",y="",covariates=c("yr","","-","%","%","cal","pounds")))
```

```
## [1] "age"       "sex"       "ph.ecog"   "ph.karno"  "pat.karno" "wt.loss"
## [7] "meal.cal"
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset lung.saemix
##     Structured data: status ~ time + status + cens | id
##     X variable for graphs: time (days)
##     covariates: age (yr), sex (), ph.ecog (-), ph.karno (%), pat.karno (%), wt.loss (cal), meal.cal
##       reference class for covariate sex :  0
```

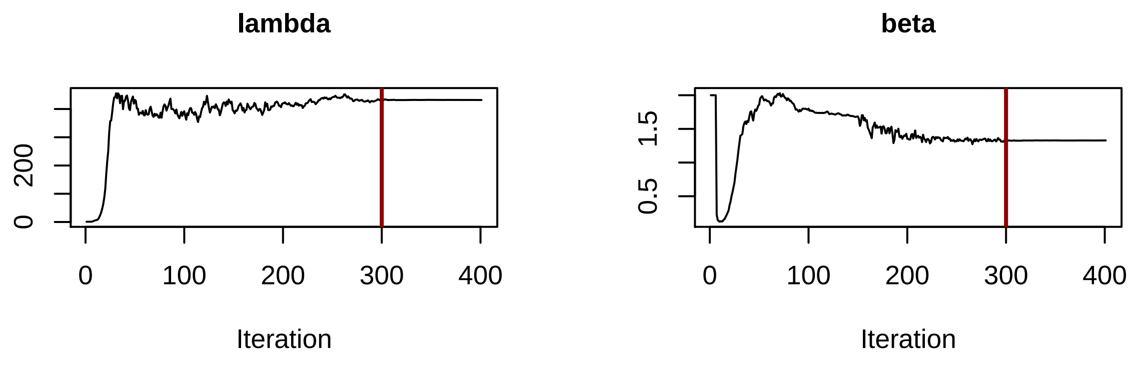
```r
weibulltte.model<-function(psi,id,xidep) {
  T<-xidep[,1]
  y<-xidep[,2] # events (1=event, 0=no event)
  cens<-which(xidep[,3]==1) # censoring times (subject specific)
  init <- which(T==0)
  lambda <- psi[id,1] # Parameters of the Weibull model
  beta <- psi[id,2]
  Nj <- length(T)

  ind <- setdiff(1:Nj, append(init,cens)) # indices of events
  hazard <- (beta/lambda)*(T/lambda)^(beta-1) # ln(H')
  H <- (T/lambda)^beta # ln(H)
  logpdf <- rep(0,Nj) # ln(l(T=0))=0
  logpdf[cens] <- -H[cens] + H[cens-1] # ln(l(T=censoring time))
  logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind]) # ln(l(T=event time))
  return(logpdf)
}

saemix.model<-saemixModel(model=weibulltte.model,description="time model",modeltype="likelihood",
  psi0=matrix(c(1,2),ncol=2,byrow=TRUE,dimnames=list(NULL,  c("lambda","beta"))),
  transform.par=c(1,1),covariance.model=matrix(c(1,0,0,0),ncol=2, byrow=TRUE))
```

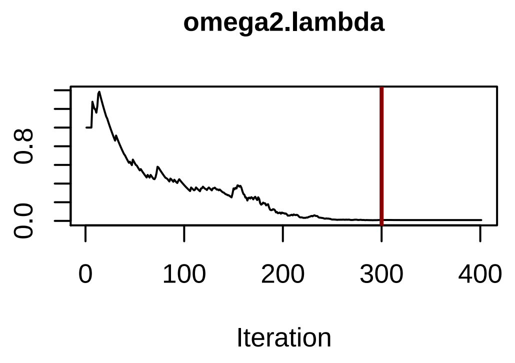
```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  time model  Model type:  likelihood
## function(psi,id,xidep) {
##   T<-xidep[,1]
##   y<-xidep[,2] # events (1=event, 0=no event)
##   cens<-which(xidep[,3]==1) # censoring times (subject specific)
##   init <- which(T==0)
##   lambda <- psi[id,1] # Parameters of the Weibull model
##   beta <- psi[id,2]
##   Nj <- length(T)
##
##   ind <- setdiff(1:Nj, append(init,cens)) # indices of events
##   hazard <- (beta/lambda)*(T/lambda)^(beta-1) # ln(H')
##   H <- (T/lambda)^beta # ln(H)
##   logpdf <- rep(0,Nj) # ln(l(T=0))=0
##   logpdf[cens] <- -H[cens] + H[cens-1] # ln(l(T=censoring time))
##   logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind]) # ln(l(T=event time))
##   return(logpdf)
## }
##   Nb of parameters: 2
##       parameter names:  lambda beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##        lambda beta
## lambda      1    0
## beta        0    0
##     No covariate in the model.
##     Initial values
##              lambda beta
## Pop.CondInit      1    2
```

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

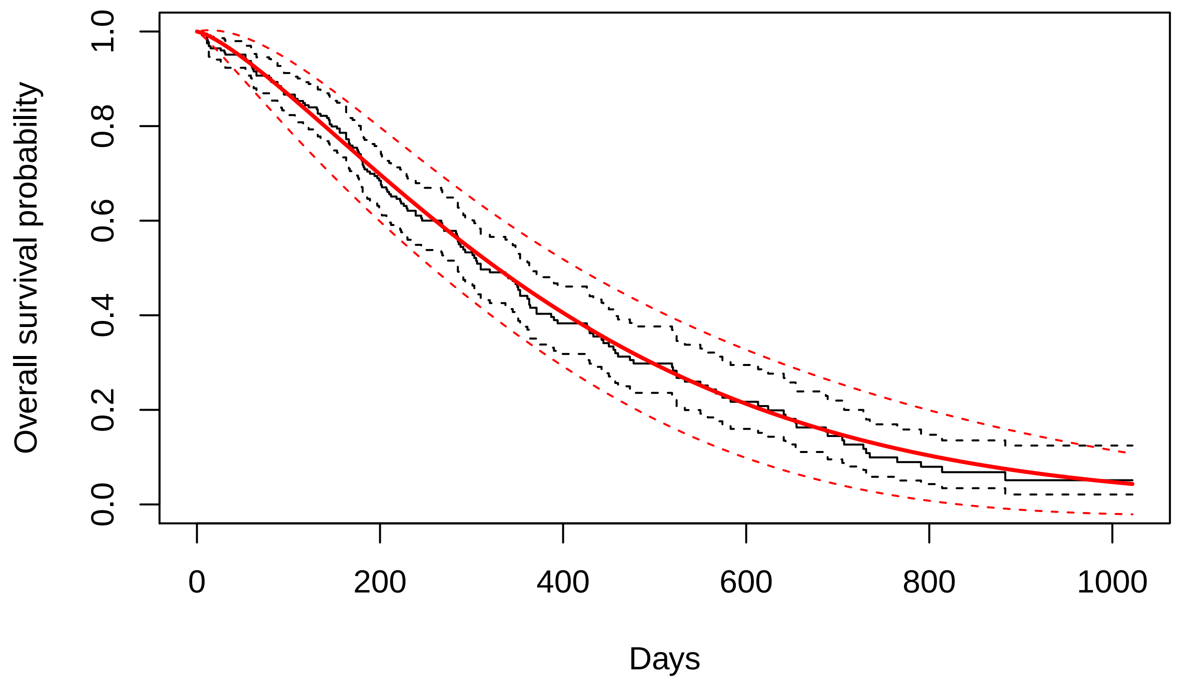
```
## ind.fix10= 2 ind.fix11= 1 ind.fix1= 1 2 ind.fix0=
## 5.952352 0.5214079
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data             ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset lung.saemix
##     Structured data: status ~ time + status + cens | id
##     X variable for graphs: time (days)
##     covariates: age (yr), sex (), ph.ecog (-), ph.karno (%), pat.karno (%), wt.loss (cal), meal.cal
##       reference class for covariate sex :  0
## Dataset characteristics:
##     number of subjects:     225
##     number of observations: 450
##     average/min/max nb obs: 2.00  /  2  /  2
## First 10 lines of data:
##    id time status cens status.1 age sex ph.ecog ph.karno pat.karno wt.loss
## 1   1    0      0    0        0  74   0       1       90       100      NA
## 2   1  306      1    0        1  74   0       1       90       100      NA
## 3   2    0      0    0        0  68   0       0       90        90      15
## 4   2  455      1    0        1  68   0       0       90        90      15
## 5   3    0      0    0        0  56   0       0       90        90      15
## 6   3 1010      0    1        0  56   0       0       90        90      15
## 7   4    0      0    0        0  57   0       1       90        60      11
## 8   4  210      1    0        1  57   0       1       90        60      11
## 9   5    0      0    0        0  60   0       0      100        90       0
## 10  5  883      1    0        1  60   0       0      100        90       0
##    meal.cal mdv cens.1 occ ytype
## 1      1175   0      0   1     1
## 2      1175   0      0   1     1
## 3      1225   0      0   1     1
## 4      1225   0      0   1     1
## 5        NA   0      0   1     1
## 6        NA   0      0   1     1
## 7      1150   0      0   1     1
## 8      1150   0      0   1     1
## 9        NA   0      0   1     1
## 10       NA   0      0   1     1
## -----------------------------------
## ----          Model            ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  time model  Model type:  likelihood
## function(psi,id,xidep) {
##   T<-xidep[,1]
##   y<-xidep[,2] # events (1=event, 0=no event)
##   cens<-which(xidep[,3]==1) # censoring times (subject specific)
##   init <- which(T==0)
##   lambda <- psi[id,1] # Parameters of the Weibull model
##   beta <- psi[id,2]
##   Nj <- length(T)
##
##   ind <- setdiff(1:Nj, append(init,cens)) # indices of events
##   hazard <- (beta/lambda)*(T/lambda)^(beta-1) # ln(H')
##   H <- (T/lambda)^beta # ln(H)
##   logpdf <- rep(0,Nj) # ln(l(T=0))=0
##   logpdf[cens] <- -H[cens] + H[cens-1] # ln(l(T=censoring time))
##   logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind]) # ln(l(T=event time))
##   return(logpdf)
## }
## <bytecode: 0x56239c4f35b0>
##   Nb of parameters: 2
##       parameter names:  lambda beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##        lambda beta
## lambda      1    0
## beta        0    0
##     No covariate in the model.
##     Initial values
##             lambda beta
## Pop.CondInit      1    2
## ----------------------------------
## ----    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,] lambda    431.8    51.60 12
## [2,] beta        1.3     0.19 14
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##        Parameter     Estimate SE   CV(%)
## lambda omega2.lambda 0.009    0.17 1858
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##              omega2.lambda
## omega2.lambda 1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 5189.352
##       AIC = 5197.352
##       BIC = 5211.017
##
## Likelihood computed by importance sampling
##       -2LL= 2269.357
##       AIC = 2277.357
##       BIC = 2291.021
## ----------------------------------------------------
```

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





```
ypred<-predict(tte.fit)

# Use survival package to assess Survival curve
if(TRUE) {
  library(survival)
  lung.surv<-lung.saemix[lung.saemix$time>0,]
  lung.surv$status<-lung.surv$status+1
  Surv(lung.surv$time, lung.surv$status) # 1=censored, 2=dead
  f1 <- survfit(Surv(time, status) ~ 1, data = lung.surv)
  xtim<-seq(0,max(lung.saemix$time), length.out=200)
  estpar<-tte.fit@results@fixed.effects
  estse<-tte.fit@results@se.fixed
  ypred<-exp(-(xtim/estpar[1])^(estpar[2]))

# Computing SE for the survival curve based on linearised FIM (probably not a good idea) through the de
  invfim<-solve(tte.fit@results@fim[1:2,1:2])
  xcal<- (xtim/estpar[1])^estpar[2]
  dsdbeta<- -log(xtim/estpar[1]) * xcal *exp(-xcal)
  dsdalpha<- estpar[2]/estpar[1] * xcal *exp(-xcal)
  xmat<-rbind(dsdalpha, dsdbeta)
  #    x1<-t(xmat[,1:3]) %*% invfim %*% xmat[,1:3]
  sesurv<-rep(0,length(xcal))
  for(i in 1:length(xcal))
    sesurv[i]<-sqrt(t(xmat[,i]) %*% invfim %*% xmat[,i])
  plot(f1, xlab = "Days", ylab = "Overall survival probability")
  lines(xtim,ypred, col="red",lwd=2)
  lines(xtim,ypred+1.96*sesurv, col="red",lwd=1, lty=2)
  lines(xtim,ypred-1.96*sesurv, col="red",lwd=1, lty=2)

  # ypred2<-exp(-(xtim/(estpar[1]-1.96*sqrt(invfim[1,1])))^(estpar[2]+1.96*sqrt(invfim[2,2])))
  # ypred3<-exp(-(xtim/(estpar[1]+1.96*sqrt(invfim[1,1])))^(estpar[2]+1.96*sqrt(invfim[2,2])))
  # lines(xtim,ypred2, col="blue",lwd=1, lty=2)
  # lines(xtim,ypred3, col="blue",lwd=1, lty=2)
}
```



RTTE model

- again difficult to find real data
- simulated data

    - Exemple simulé de Belhal **TODO**
    - data from the Monolix documentation: absolutely no indication where the data comes from (weibull_data.txt for the weibullRTTE.mlxtran project in the demo)
- search for real data
    - asked Ulrika Simonsson for the RTTE data on post-operative pain (Pain Medicine 2015)
    - data on events in Gaucher disease used for the ENSAI workshops (but few events)
    - discretised PCA events during warfarin treatment ? (from the warfarin PK/PD) (but threshold ?)

```r
# Simulating TTE data
set.seed(12345)

nsuj<-50
xtim<-c(0)
tte.data<-data.frame(id=rep(1:nsuj,each=length(xtim)),time=rep(xtim,nsuj))
psiM<-data.frame(lambda=seq(1.6,2,length.out=length(unique(tte.data$id))),beta = 2)

simul.tte<-function(psi,id,xidep) {
  T<-xidep
  N <- nrow(psi)
  Nj <- length(T)
  censoringtime = 3
  lambda <- psi[id,1]
  beta <- psi[id,2]
  obs <-rep(0,length(T))
  for (i in (1:N)){
    obs[id==i] <- rweibull(n=length(id[id==i]), shape=beta[i], scale=lambda[i])
  }
  obs[obs>censoringtime]<-censoringtime
  return(obs)
}

preds <- simul.tte(psiM, tte.data$id, tte.data[,c("time")])
tte.data$y<-0
tte.data$tlat<-preds
dat1<-tte.data[,c("id","time","y")]
dat2<-tte.data[,c("id","tlat","y")]
dat2$y<-as.integer(dat2$tlat>0 & dat2$tlat<3)
colnames(dat2)[2]<-"time"
tte.data<-rbind(dat1,dat2)
tte.data<-tte.data[order(tte.data$id, tte.data$time),]
tte.psiM<-psiM

# Simulate T from Weibull (check)
if(FALSE) {
  lambda<-2
  beta<-2
  nsim<-5000
  # By hand
  q1<-runif(nsim)
  #  tevent<-lambda*exp(log(q1)/beta)
  tevent<-lambda*exp(log(-log(q1))/beta)
  tevent<-sort(tevent)
#  plot(tevent, exp(-(tevent/lambda)^beta))
  tevent2<-sort(rweibull(nsim, shape=beta, scale=lambda))
  plot(tevent, tevent2)
  abline(0,1)

}
```

```
saemix.data<-saemixData(name.data=tte.data, name.group=c("id"),
  name.predictors=c("time"), name.response="y")
```

```
##
##
## The following SaemixData object was successfully created:
##
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset tte.data
##     Structured data: y ~ time | id
##     Predictor: time ()
```

```
tte.model<-function(psi,id,xidep) {
  T<-xidep[,1]
  N <- nrow(psi)
  Nj <- length(T)
  # censoringtime = 6
  censoringtime = max(T)
  lambda <- psi[id,1]
  beta <- psi[id,2]
  init <- which(T==0)
  cens <- which(T==censoringtime)
  ind <- setdiff(1:Nj, append(init,cens))
  hazard <- (beta/lambda)*(T/lambda)^(beta-1)
  H <- (T/lambda)^beta
  logpdf <- rep(0,Nj)
  logpdf[cens] <- -H[cens] + H[cens-1]
  logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind])
  return(logpdf)
}

saemix.model<-saemixModel(model=tte.model,description="time model",modeltype="likelihood",
  psi0=matrix(c(1,2),ncol=2,byrow=TRUE,dimnames=list(NULL,  c("lambda","beta"))),
  transform.par=c(1,1),covariance.model=matrix(c(1,0,0,1),ncol=2, byrow=TRUE))
```

```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  time model  Model type:  likelihood
## function(psi,id,xidep) {
##   T<-xidep[,1]
##   N <- nrow(psi)
##   Nj <- length(T)
##   # censoringtime = 6
##   censoringtime = max(T)
##   lambda <- psi[id,1]
##   beta <- psi[id,2]
##   init <- which(T==0)
##   cens <- which(T==censoringtime)
##   ind <- setdiff(1:Nj, append(init,cens))
##   hazard <- (beta/lambda)*(T/lambda)^(beta-1)
##   H <- (T/lambda)^beta
##   logpdf <- rep(0,Nj)
##   logpdf[cens] <- -H[cens] + H[cens-1]
##   logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind])
##   return(logpdf)
## }
##   Nb of parameters: 2
##       parameter names:  lambda beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##        lambda beta
## lambda      1    0
## beta        0    1
##     No covariate in the model.
##     Initial values
##              lambda beta
## Pop.CondInit      1    2
```

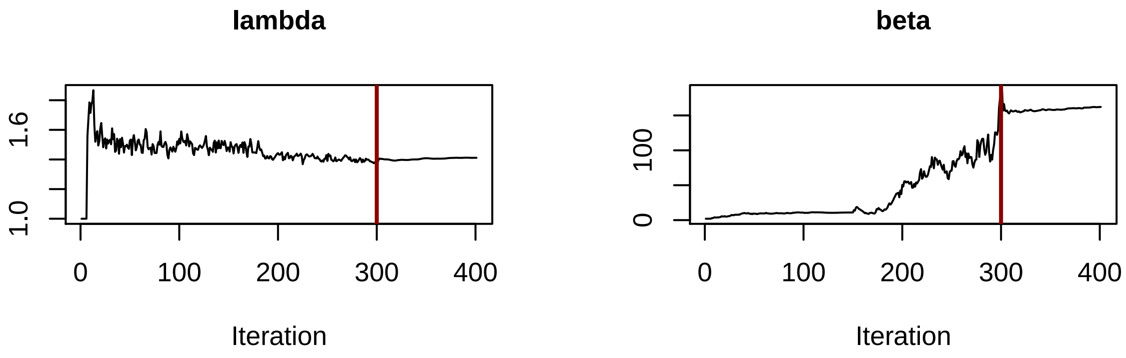
```
saemix.model<-saemixModel(model=tte.model,description="time model",modeltype="likelihood",
  psi0=matrix(c(1,2),ncol=2,byrow=TRUE,dimnames=list(NULL,  c("lambda","beta"))),
  transform.par=c(1,1),covariance.model=matrix(c(1,0,0,0),ncol=2, byrow=TRUE))
```

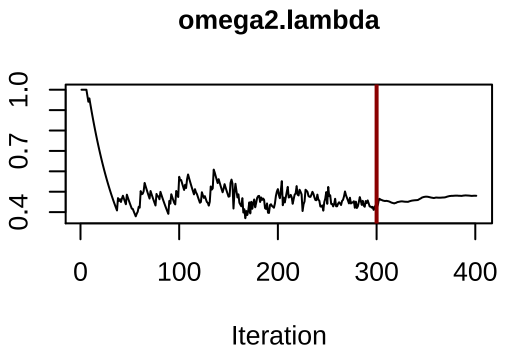
```
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  time model  Model type:  likelihood
## function(psi,id,xidep) {
##   T<-xidep[,1]
##   N <- nrow(psi)
##   Nj <- length(T)
##   # censoringtime = 6
##   censoringtime = max(T)
##   lambda <- psi[id,1]
##   beta <- psi[id,2]
##   init <- which(T==0)
##   cens <- which(T==censoringtime)
##   ind <- setdiff(1:Nj, append(init,cens))
##   hazard <- (beta/lambda)*(T/lambda)^(beta-1)
##   H <- (T/lambda)^beta
##   logpdf <- rep(0,Nj)
##   logpdf[cens] <- -H[cens] + H[cens-1]
##   logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind])
##   return(logpdf)
## }
##   Nb of parameters: 2
##       parameter names:  lambda beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##        lambda beta
## lambda      1    0
## beta        0    0
##     No covariate in the model.
##     Initial values
##               lambda beta
## Pop.CondInit      1    2
```

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

```
## ind.fix10= 2 ind.fix11= 1 ind.fix1= 1 2 ind.fix0=
## 0.3733829 2.393744
## Nonlinear mixed-effects model fit by the SAEM algorithm
## -----------------------------------
## ----          Data              ----
## -----------------------------------
## Object of class SaemixData
##     longitudinal data for use with the SAEM algorithm
## Dataset tte.data
##     Structured data: y ~ time | id
##     Predictor: time ()
## Dataset characteristics:
##     number of subjects:     50
##     number of observations: 100
##     average/min/max nb obs: 2.00  /  2  /  2
## First 10 lines of data:
##    id      time y mdv cens occ ytype
## 1   1 0.0000000 0   0    0   1     1
## 51  1 0.9152915 1   0    0   1     1
## 2   2 0.0000000 0   0    0   1     1
## 52  2 0.5857074 1   0    0   1     1
## 3   3 0.0000000 0   0    0   1     1
## 53  3 0.8447454 1   0    0   1     1
## 4   4 0.0000000 0   0    0   1     1
## 54  4 0.5648408 1   0    0   1     1
## 5   5 0.0000000 0   0    0   1     1
## 55  5 1.4458047 1   0    0   1     1
## -----------------------------------
## ----          Model             ----
## -----------------------------------
## Nonlinear mixed-effects model
##   Model function:  time model  Model type:  likelihood
## function(psi,id,xidep) {
##   T<-xidep[,1]
##   N <- nrow(psi)
##   Nj <- length(T)
##   # censoringtime = 6
##   censoringtime = max(T)
##   lambda <- psi[id,1]
##   beta <- psi[id,2]
##   init <- which(T==0)
##   cens <- which(T==censoringtime)
##   ind <- setdiff(1:Nj, append(init,cens))
##   hazard <- (beta/lambda)*(T/lambda)^(beta-1)
##   H <- (T/lambda)^beta
##   logpdf <- rep(0,Nj)
##   logpdf[cens] <- -H[cens] + H[cens-1]
##   logpdf[ind] <- -H[ind] + H[ind-1] + log(hazard[ind])
##   return(logpdf)
## }
## <bytecode: 0x5623a4228830>
##   Nb of parameters: 2
##       parameter names:  lambda beta
##       distribution:
##      Parameter Distribution Estimated
## [1,] lambda    log-normal   Estimated
## [2,] beta      log-normal   Estimated
##   Variance-covariance matrix:
##        lambda beta
## lambda      1    0
## beta        0    0
##     No covariate in the model.
##     Initial values
##              lambda beta
## Pop.CondInit      1    2
## ----------------------------------
## ----    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,] lambda      1.4       0.58   41
## [2,] beta      162.2    5675.89 3500
## ----------------------------------------------------
## -----------  Variance of random effects  -----------
## ----------------------------------------------------
##        Parameter     Estimate SE   CV(%)
## lambda omega2.lambda 0.48     0.21 44
## ----------------------------------------------------
## ------  Correlation matrix of random effects  ------
## ----------------------------------------------------
##               omega2.lambda
## omega2.lambda 1
## ----------------------------------------------------
## ---------------  Statistical criteria  -------------
## ----------------------------------------------------
## Likelihood computed by linearisation
##       -2LL= 573.0104
##       AIC = 581.0104
##       BIC = 588.6585
##
## Likelihood computed by importance sampling
##       -2LL= 122.2899
##       AIC = 130.2899
##       BIC = 137.938
## ----------------------------------------------------
```

```r
plot(tte.fit, plot.type="convergence")
```


lambda
Iteration
beta
Iteration


omega2.lambda
Iteration