

Testing examples in saemix 3.2 - continuous models

Emmanuelle

10/11/2022

Objective

Check saemix for continuous data models

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

```
if(testMode) cat("Testing package\n") else cat("Loading libraries\n")
```

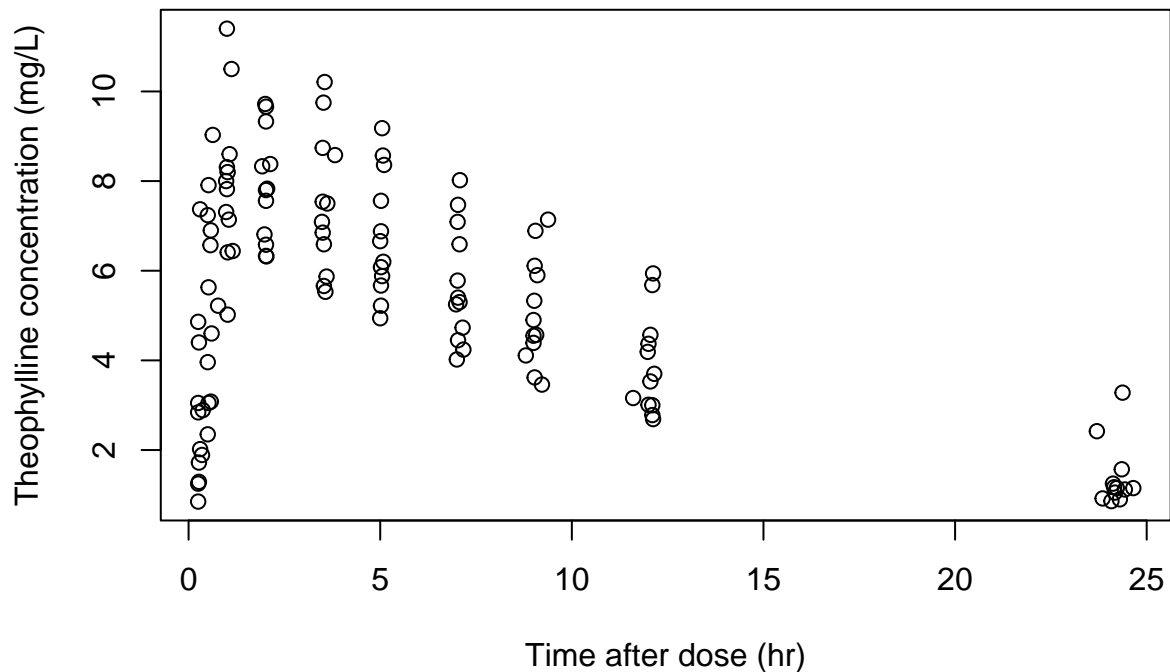
```
## Loading libraries
```

Continuous response model

Theophylline

```
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)")
```

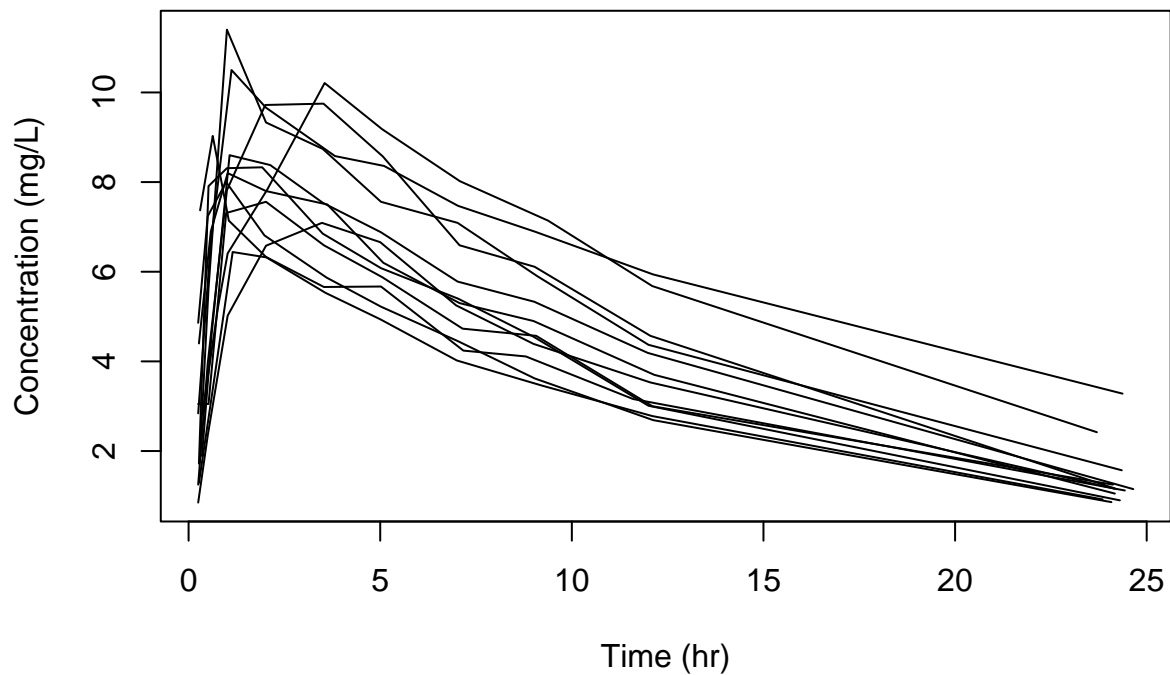


```
saemix.data<-saemixData(name.data=theo.saemix,header=TRUE,sep=" ",na=NA,
  name.group=c("Id"),name.predictors=c("Dose","Time"),
  name.response=c("Concentration"),name.covariates=c("Weight","Sex"),
  units=list(x="hr",y="mg/L",covariates=c("kg","-")), name.X="Time")
```

```
##
##
## 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
```

```
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)
}
```

```
plot(saemix.data)
```



```
# 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]

```

```

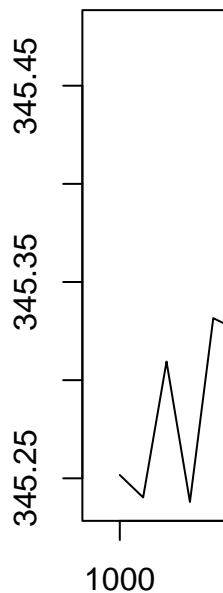
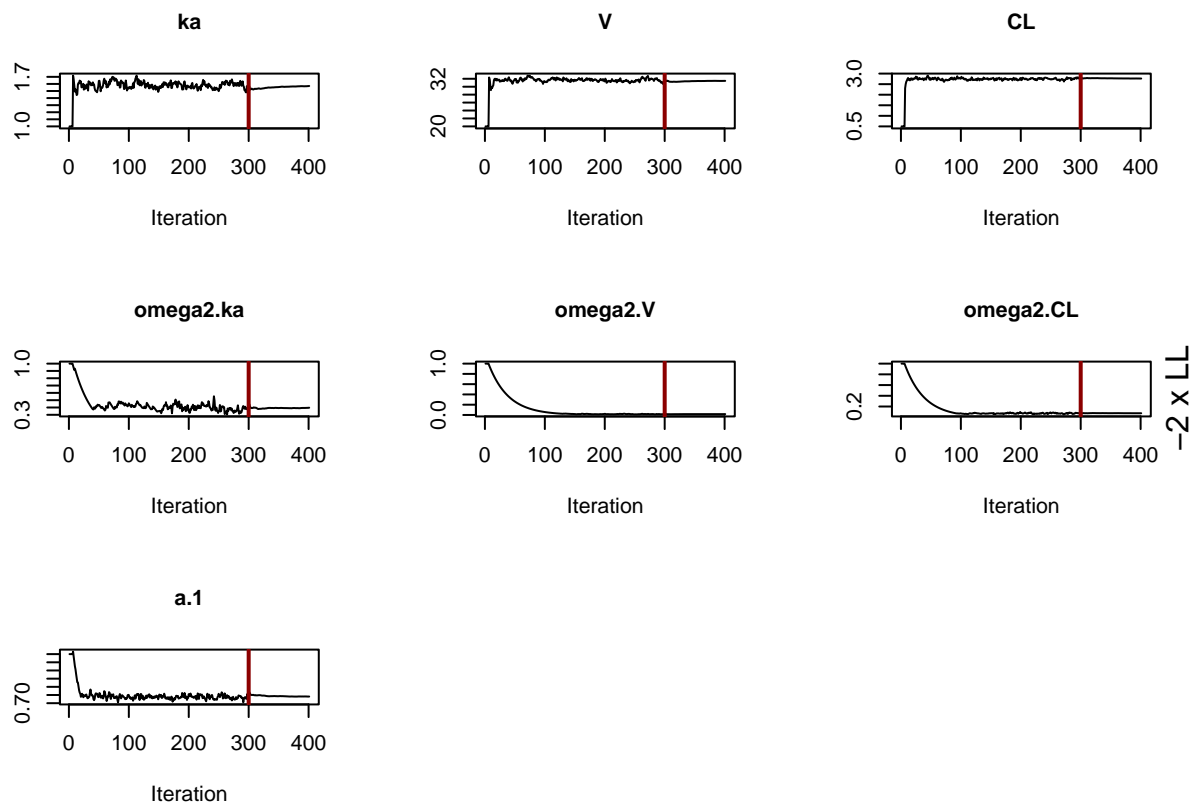
##      CL<-psi[id,3]
##      k<-CL/V
##      ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
##      return(ypred)
## }
## <bytecode: 0x55d7ffec7af8>
##      Nb of parameters: 3
##      parameter names:  ka V CL
##      distribution:
##      Parameter Distribution Estimated
## [1,] ka      log-normal  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

```

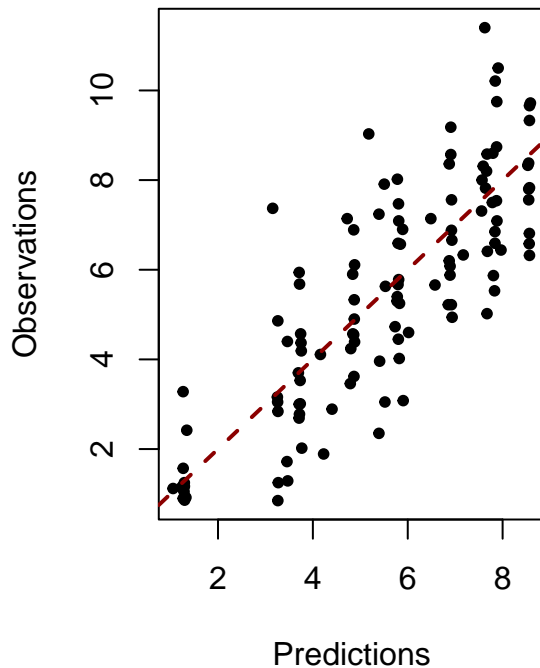
```
## V omega2.V 0.017 0.0096 58
## CL omega2.CL 0.074 0.0360 49
## -----
## ----- Correlation matrix of random effects -----
## -----
##          omega2.ka omega2.V omega2.CL
## omega2.ka 1          0          0
## omega2.V  0          1          0
## omega2.CL 0          0          1
## -----
## ----- 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
## -----
```

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

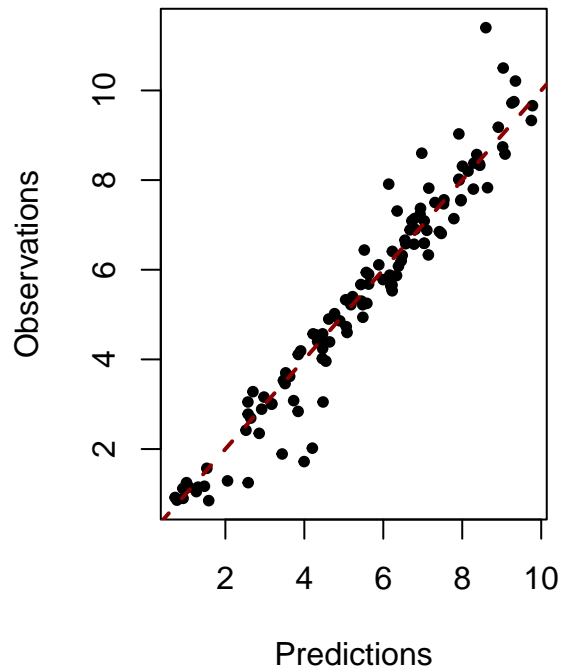
```
## Simulating data using nsim = 1000 simulated datasets
## Computing WRES and npde ..
```



Population predictions



Individual predictions, MAP



```
plot(saemix.fit, plot.type="observations")
```

```
## Simulating data using nsim = 1000 simulated datasets
## Computing WRES and npde ..
```

```
plot(saemix.fit, plot.type="npde")
```

```
## Simulating data using nsim = 1000 simulated datasets
## Computing WRES and npde ..
```

```
## Please use npdeSaemix to obtain VPC and npde
```

```
# 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="combined")
```

```
##
```

```
##
```

```
## 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: 0x55d7ffec7af8>
##      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: 0x55d7ffec7af8>
## 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 0.27
## [5,] a.1             0.5734 0.0935 16.3 -
## [6,] b.1             0.0748 0.0223 29.9 -
## -----
## ----- 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
## -----
# Warning message
plot(saemix.fit, plot.type="npde")

## Simulating data using nsim = 1000 simulated datasets
## Computing WRES and npde ..

## Please use npdeSaemix to obtain VPC and npde
# using npde instead
ynpde<-npdeSaemix(saemix.fit)

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits

```

```

## lors de la conversion automatique

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

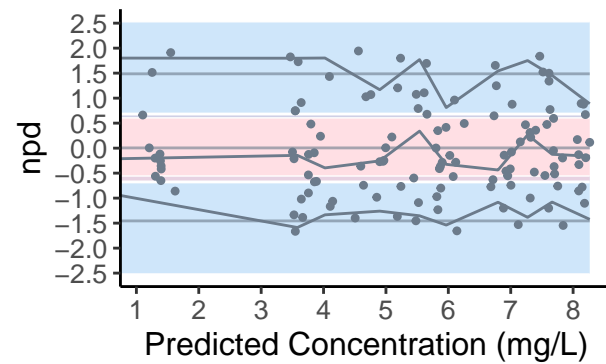
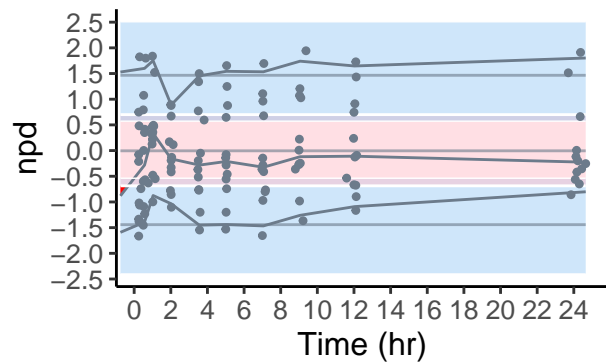
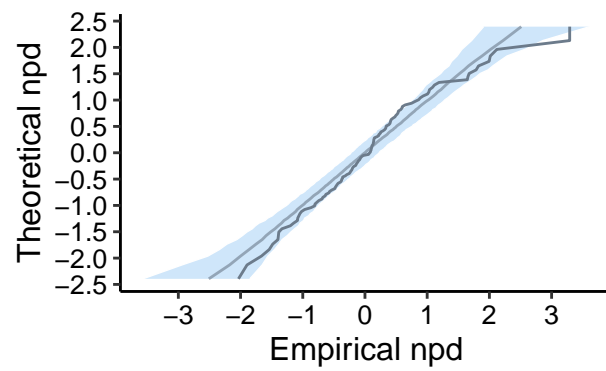
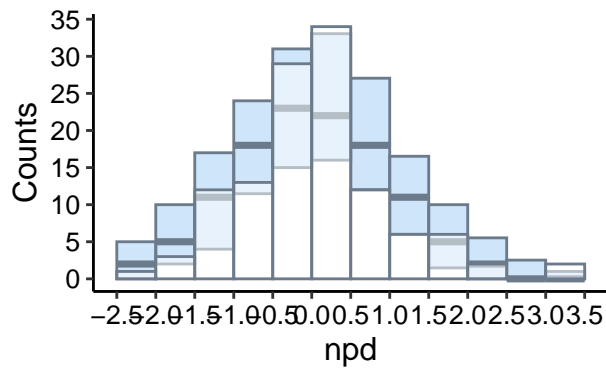
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

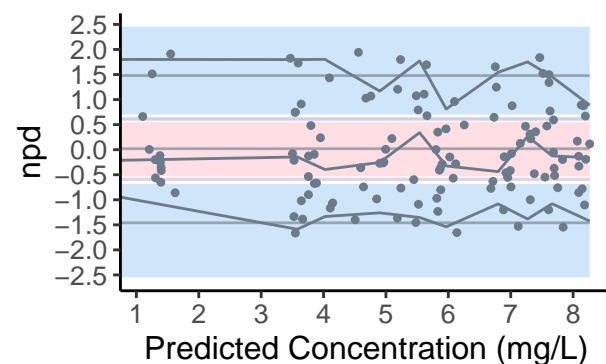
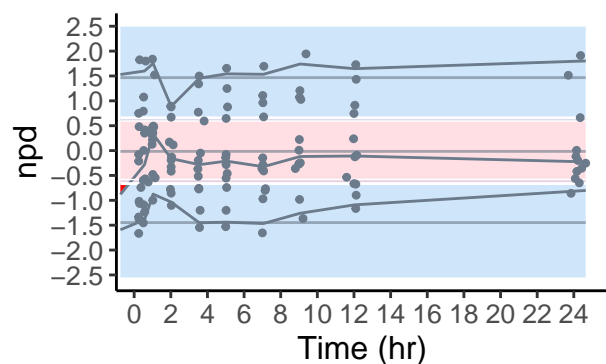
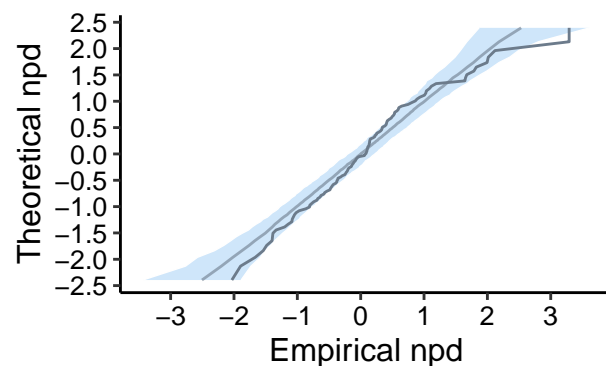
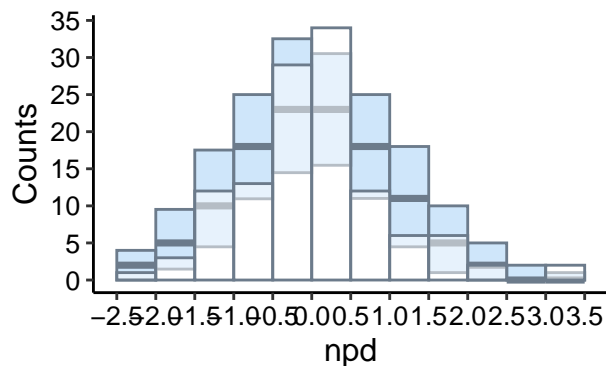
## Warning in which(!is.na(as.integer(object@name.covariates))): NAs introduits
## lors de la conversion automatique

## -----
## Distribution of npde :
##      nb of obs: 120
##      mean= 0.04778   (SE= 0.085 )
##      variance= 0.8765   (SE= 0.11 )
##      skewness= 0.6982
##      kurtosis= 1.474
## -----
## Statistical tests (adjusted p-values):
##      t-test           : 1
##      Fisher variance test : 1
##      SW test of normality : 0.00516 **
##      Global test       : 0.00516 **
## ---
## Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
## -----

```



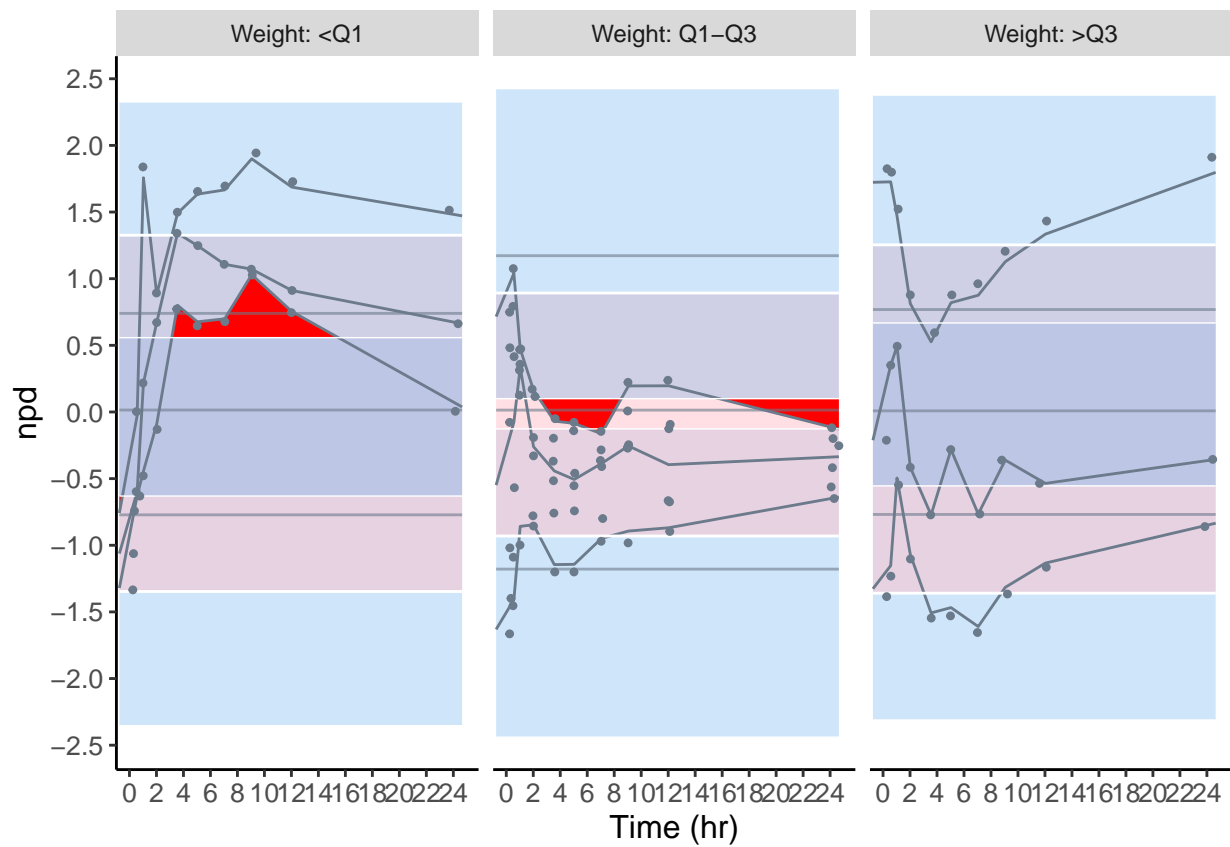
```
plot(ynpde)
```



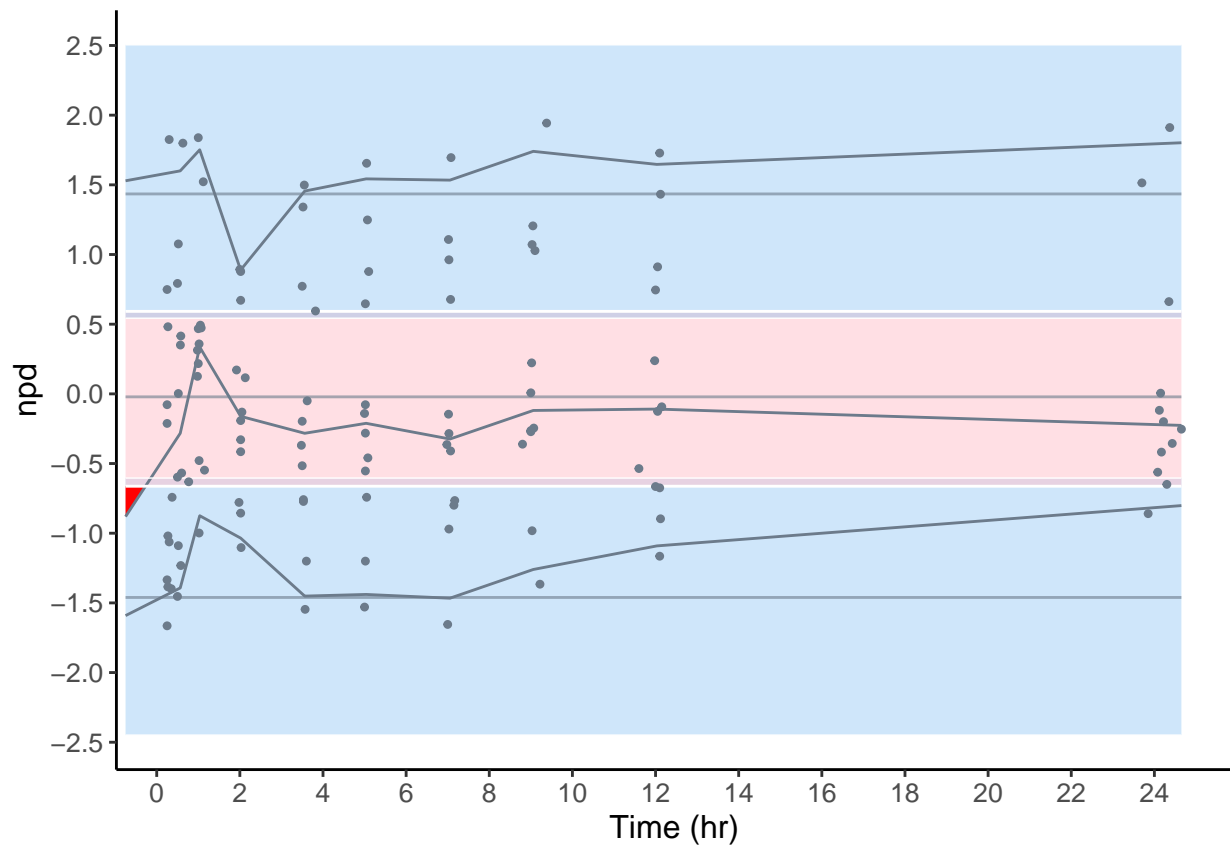
```
# individual npde plots
```

```
plot(ynpde, plot.type="x.scatter", covsplit=TRUE, which.cov=c("Weight", "Sex"))
```

```
## [[1]]
```

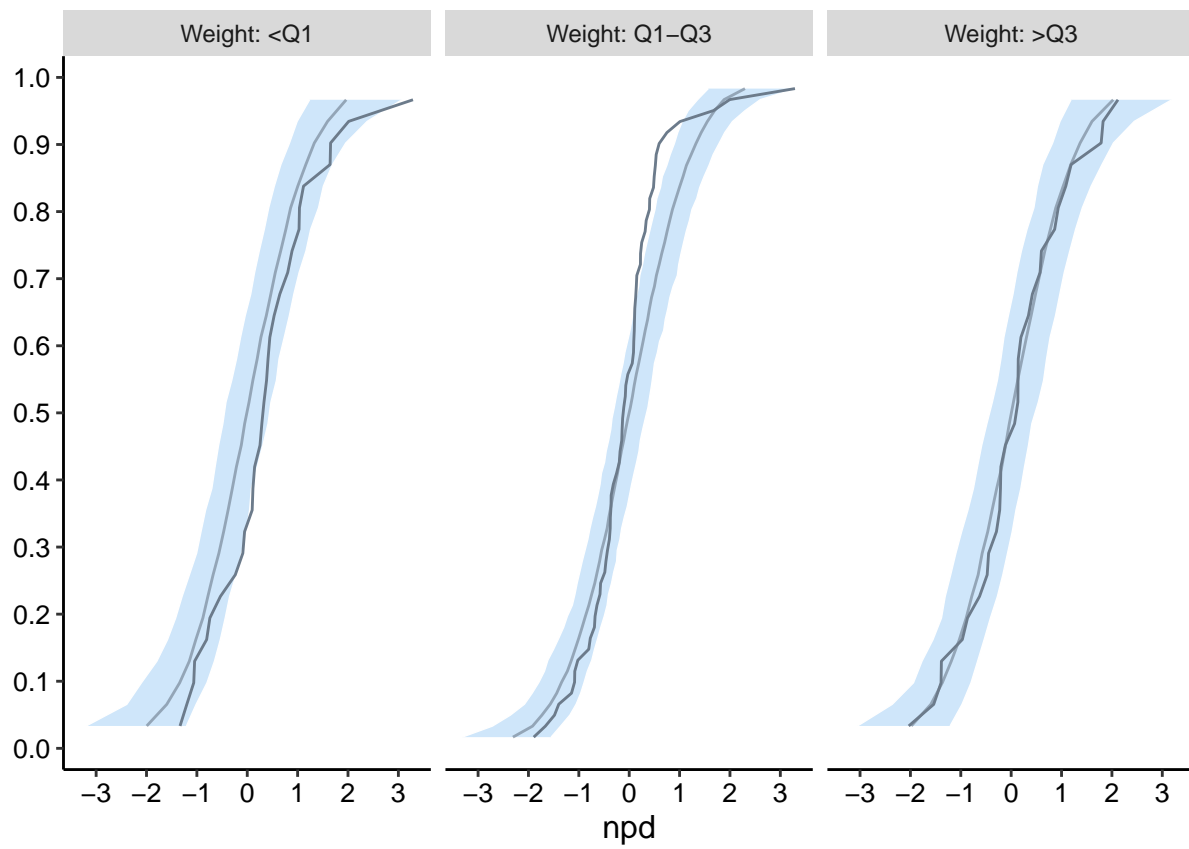


[[2]]

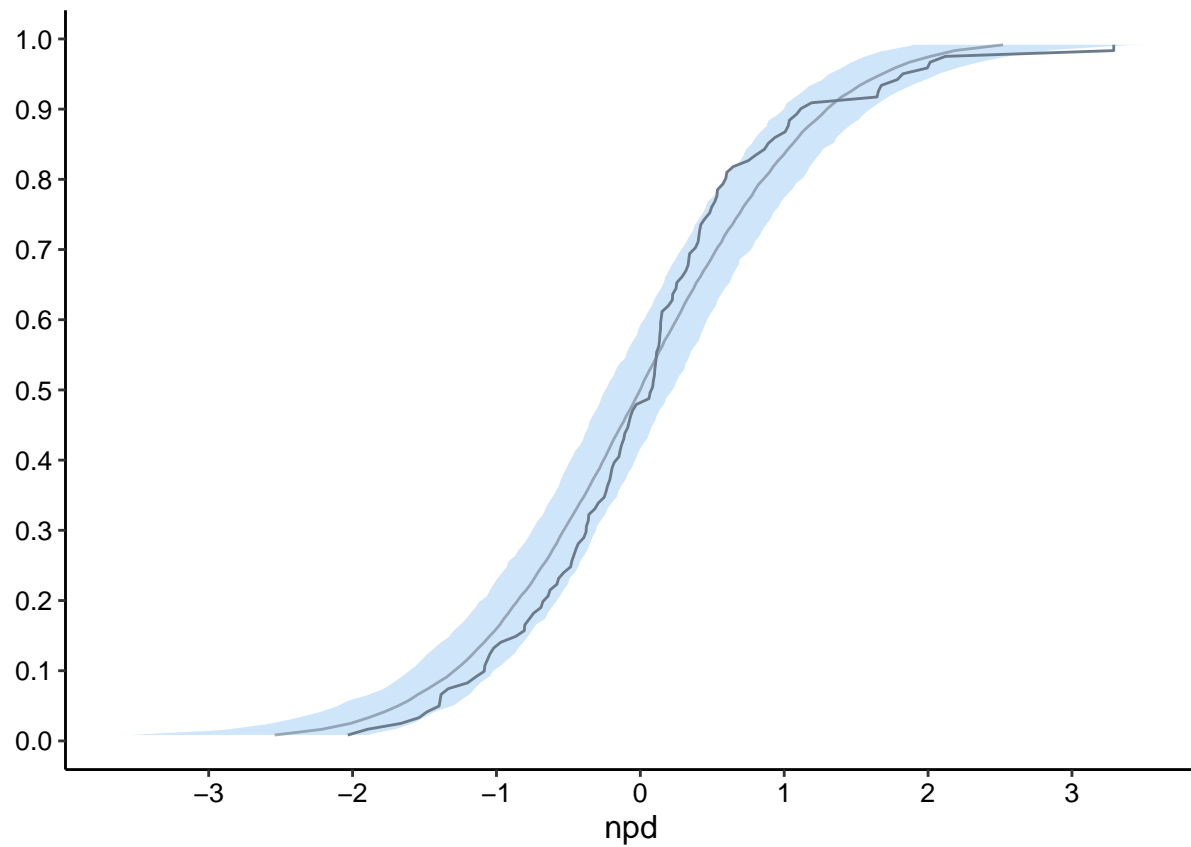


```
plot(ynpde, plot.type="ecdf", covsplit=TRUE, which.cov=c("Weight", "Sex"))
```

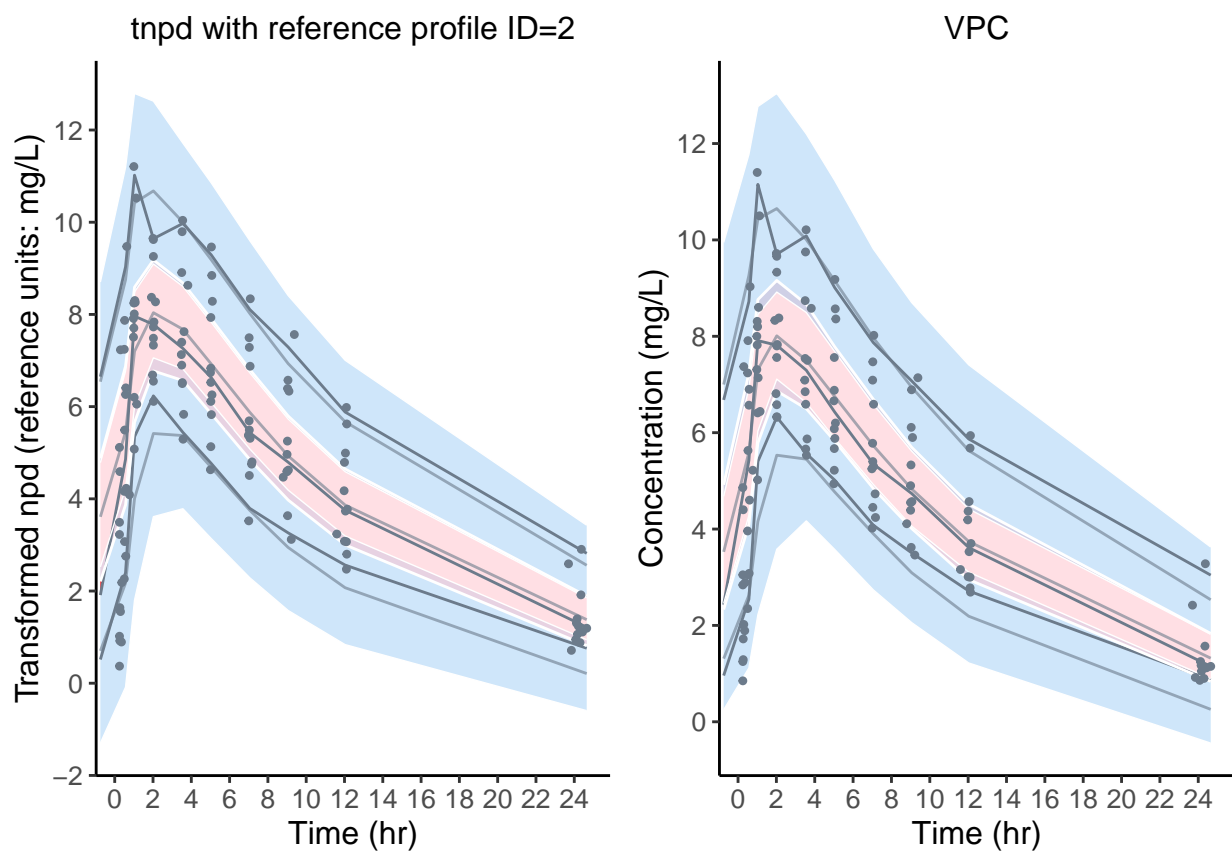
```
## [[1]]
```



```
##  
## [[2]]
```

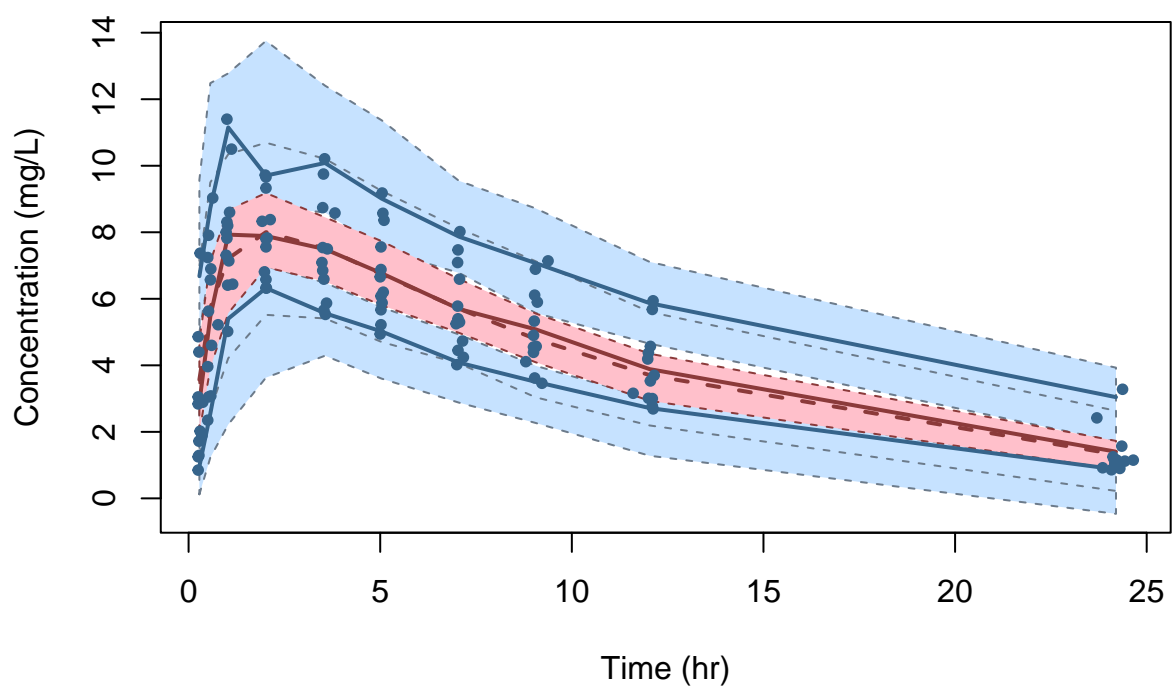


```
plot.tnpde<-plot(ynpde, plot.type="x.scatter", ref.prof=list(Id=2), main="tnpd with reference profile I")
plot.vpc<-plot(ynpde, plot.type="vpc", main="VPC")
grid.arrange(grobs=list(plot.tnpde, plot.vpc), nrow=1, ncol=2)
```

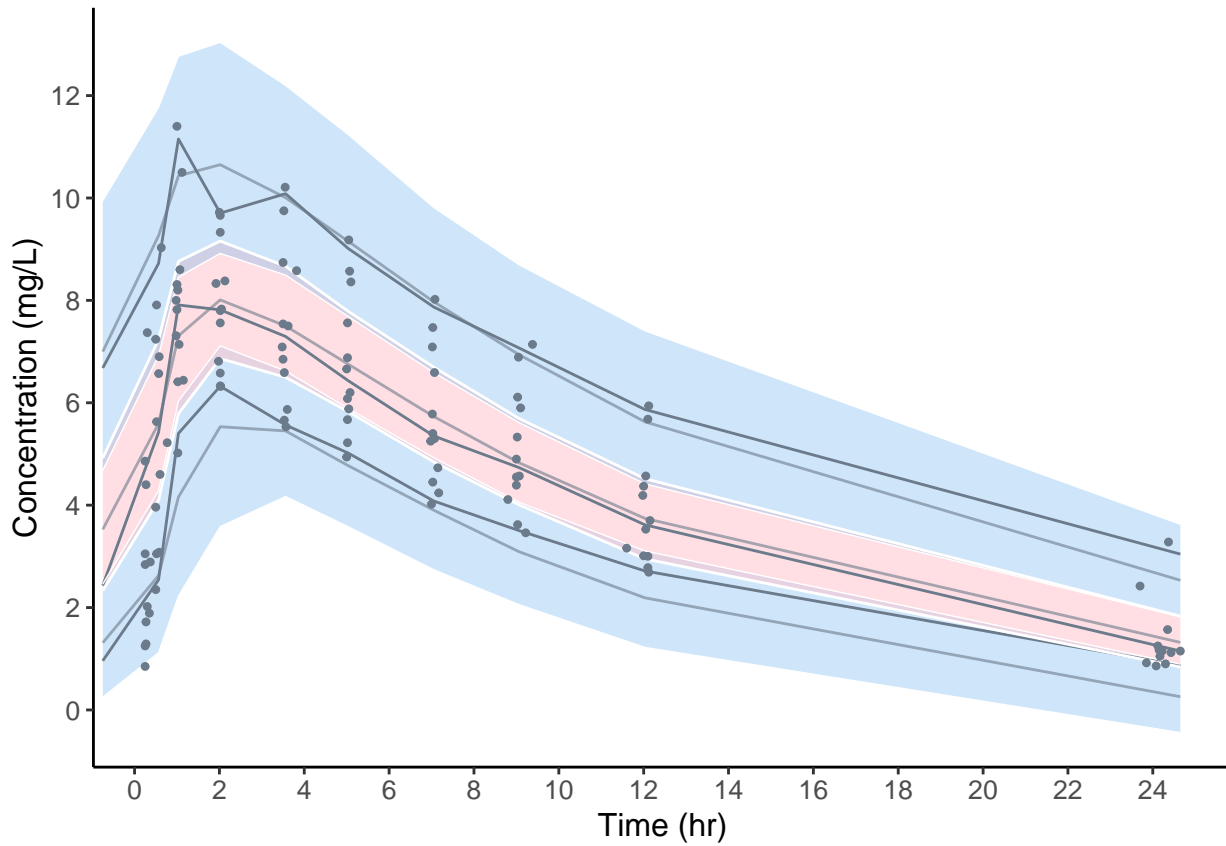



```
# VPC
plot(saemix.fit, plot.type="vpc")
```

Visual Predictive Check

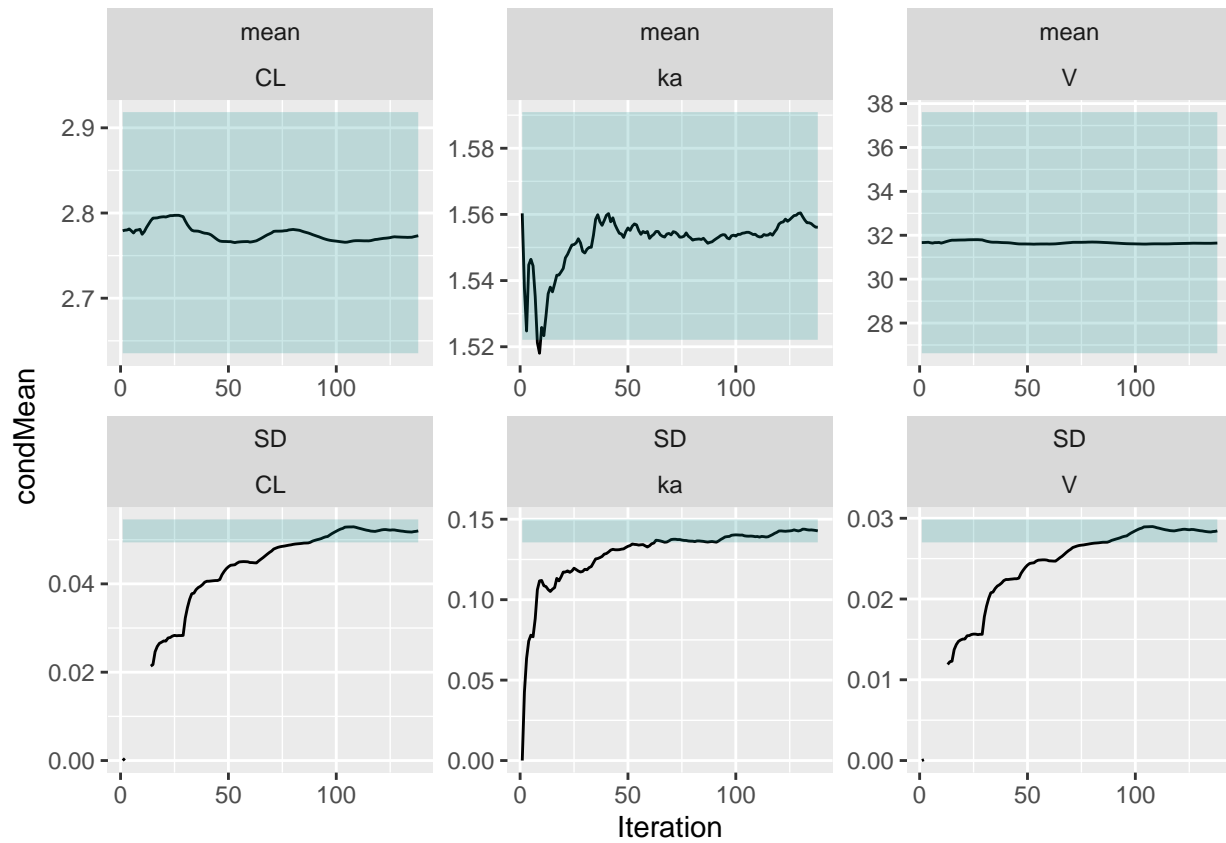


```
plot(ynpde, plot.type="vpc")
```



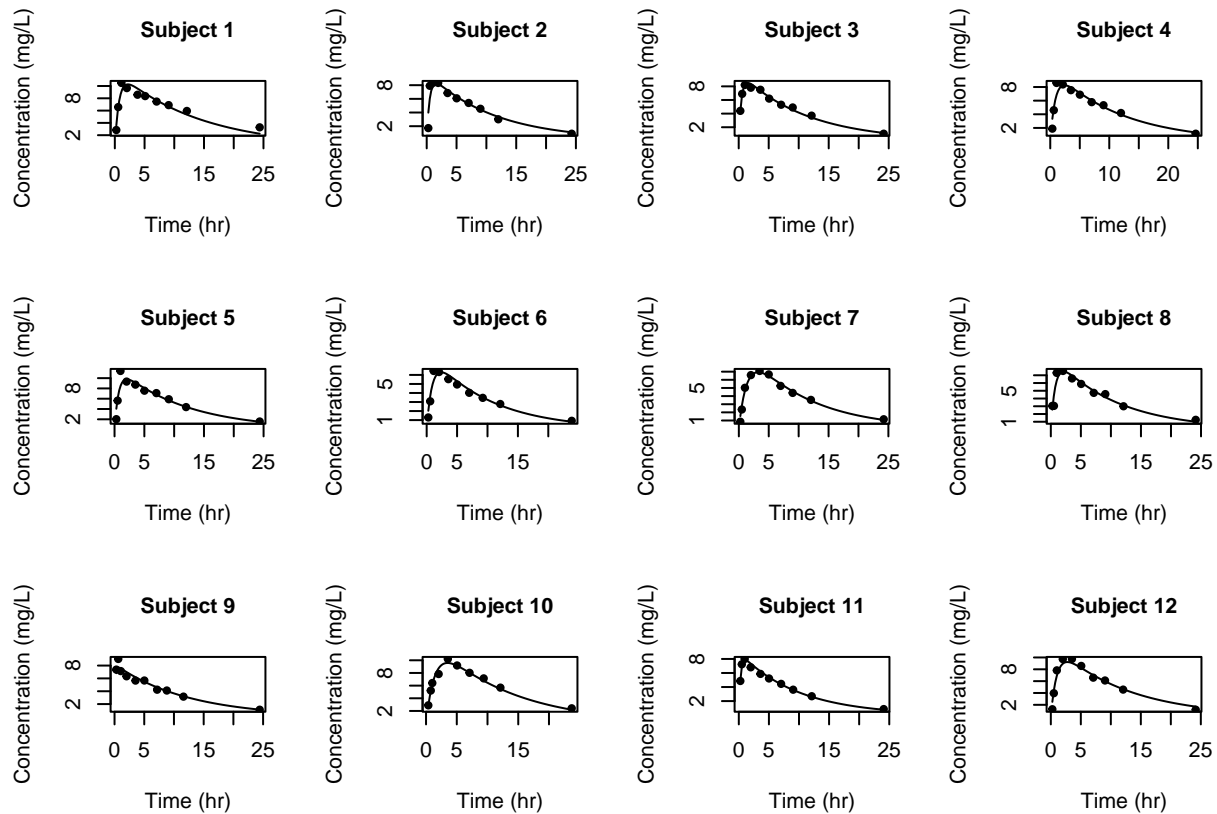
```
#saemix.fit<-condidist.saemix(saemix.fit)
saemix.fit<-condidist.saemix(saemix.fit, plot=TRUE)
```

```
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
## Warning in sqrt(varik): Production de NaN
```



```
plot(saemix.fit, plot.type="individual", smooth=TRUE)
```

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



Using predict.SaemixModel to evaluate the initial estimates

We can use the `predict.SaemixModel()` function to quickly check the initial fixed effects, either

```
# Predictors given as a dataframe
ypred <- predict.SaemixModel(saemix.model, saemix.data@data[,saemix.data@name.predictors, drop=FALSE])

# Predictors extracted from a saemixData object
ypred2 <- predict.SaemixModel(saemix.model, saemix.data)

# Summary of
cat("Check that the predictions don't give any NA and are reasonable compared to the observations\n")
```

Check that the predictions don't give any NA and are reasonable compared to the observations

```
summary(ypred$predictions)
```

```
##      id      Dose      Time      pred
##  Min.   :1  Min.   :267.8  Min.   : 0.250  Min.   : 3.457
## 1st Qu.:1 1st Qu.:319.4 1st Qu.: 1.020 1st Qu.: 8.944
## Median :1 Median :319.8 Median : 4.410 Median :12.132
## Mean   :1 Mean   :315.4 Mean   : 6.484 Mean   :10.955
## 3rd Qu.:1 3rd Qu.:320.0 3rd Qu.: 9.030 3rd Qu.:13.717
## Max.   :1 Max.   :320.6 Max.   :24.650 Max.   :14.572
```

```
summary(saemix.data@data[,saemix.data@name.response])
```

```
##      Min. 1st Qu. Median      Mean 3rd Qu.      Max.
##    0.850  3.513  5.665  5.447  7.325 11.400
```

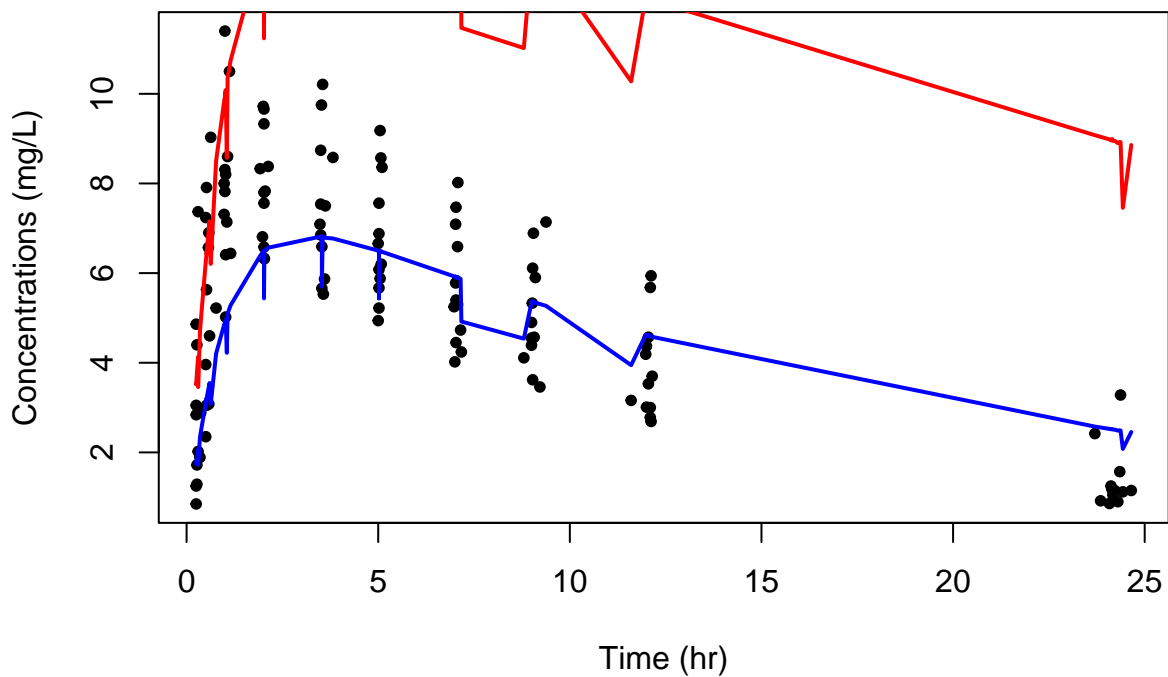
```
summary(ypred2$predictions$pred-ypred$predictions$pred)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##           0         0         0         0         0         0

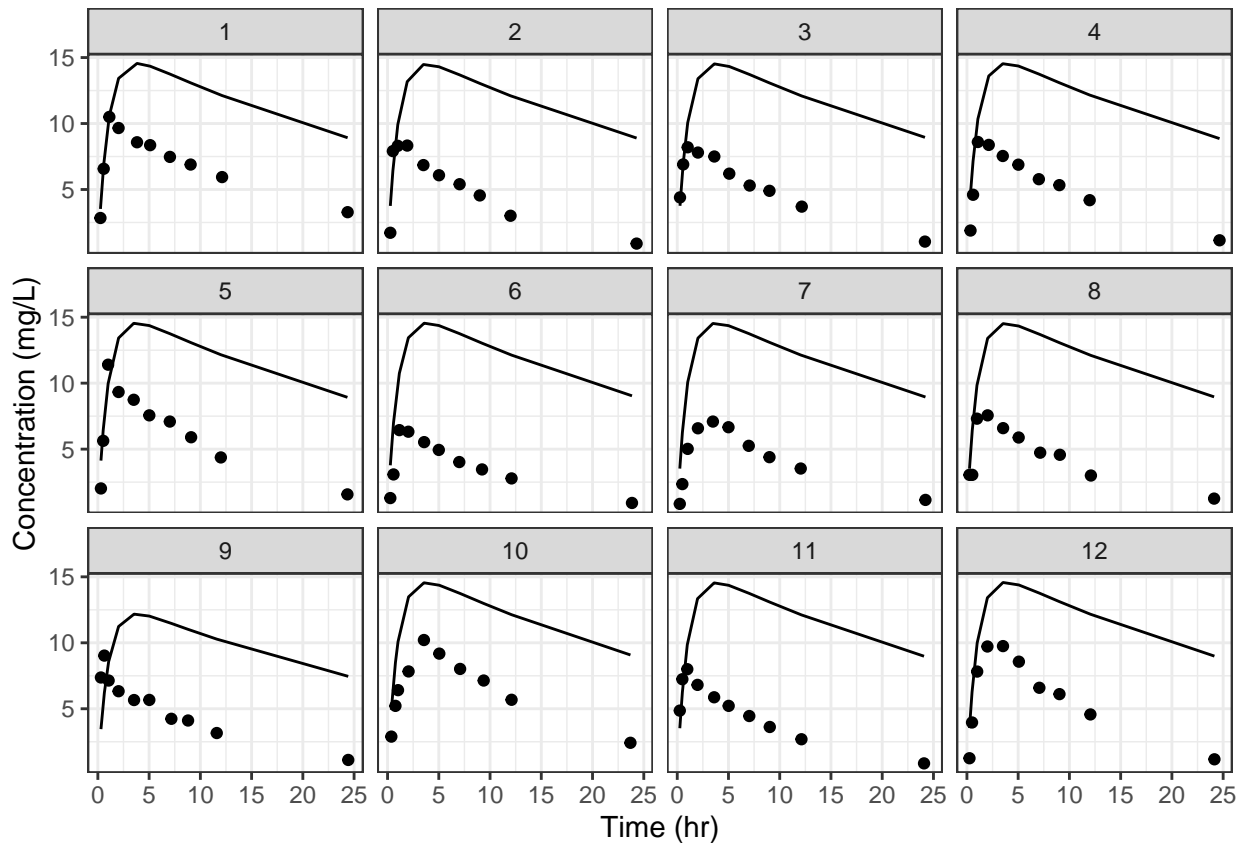
cat("Overlay the predictions and the data (roughly, not accounting for different doses...)\n")

## Overlay the predictions and the data (roughly, not accounting for different doses...)
plot(saemix.data@data$Time, saemix.data@data$Concentration, pch=20, xlab="Time (hr)", ylab="Concentration (mg/L)")
#for(i in 1:saemix.data@N) lines(ypred1$Time[ypred1$id==i], ypred1$pred[ypred1$id==i])
ypred1 <- ypred2$predictions[order(ypred2$predictions$Time),]
lines(loess(pred ~ Time, data=ypred1), col="Red", lwd=2)

# Better increasing V and CL
ypred3 <- predict.SaemixModel(saemix.model, saemix.data, psi=c(1,40,2))
ypred1 <- ypred3$predictions[order(ypred2$predictions$Time),]
lines(loess(pred ~ Time, data=ypred1), col="Blue", lwd=2)
```



```
checkInitialFixedEffects(saemix.model, saemix.data)
```



```
summary(predict(saemix.fit))
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.7692  3.8717  5.4989  5.4619  7.0435 10.2498
```

One random effect

Note: sort the message “one-dimensional optimization by Nelder-Mead is unreliable”

```
model1cpt.1<-function(psi,id,xidep) {
  dose<-xidep[,1]
  tim<-xidep[,2]
  ka<-psi[id,1]
  V<-2
  # V<-psi[id,2]
  k<-0.5
  CL<-k*V
  ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
  return(ypred)
}
saemix.model<-saemixModel(model=model1cpt.1,description="warfarin",modeltype="structural",
  psi0=matrix(c(1),ncol=1,byrow=TRUE, dimnames=list(NULL, c("ka"))),
  transform.par=c(1),omega.init=matrix(c(1),ncol=1,byrow=TRUE),
  covariance.model=matrix(c(1),ncol=1,byrow=TRUE))
```

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

```

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

```

```

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:  warfarin
## Model type:  structural
## function(psi,id,xidep) {
##   dose<-xidep[,1]
##   tim<-xidep[,2]
##   ka<-psi[id,1]
##   V<-2
##   # V<-psi[id,2]
##   k<-0.5
##   CL<-k*V
##   ypred<-dose*ka/(V*(ka-k))*(exp(-k*tim)-exp(-ka*tim))
##   return(ypred)
## }
## <bytecode: 0x55d7fee4b880>
## Nb of parameters: 1
##   parameter names:  ka
##   distribution:
##   Parameter Distribution Estimated
## [1,] ka          log-normal Estimated
## Variance-covariance matrix:
##   ka
## ka  1
## Error model: constant , initial values: a.1=1
## No covariate in the model.
## Initial values
##   ka
## Pop.CondInit  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:  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(%)
## [1,] ka          0.027   0.0015 5.8
## [2,] a.1         2.927   0.1991 6.8
## -----
## ----- Variance of random effects -----
## -----
##      Parameter Estimate SE      CV(%)
## ka omega2.ka 1.3e-05  0.017 130416
## -----
## ----- Correlation matrix of random effects -----
## -----
##      omega2.ka
## omega2.ka 1
## -----
## ----- Statistical criteria -----
## -----
## Likelihood computed by linearisation
##      -2LL= 598.2676
##      AIC = 604.2676
##      BIC = 605.7223
##
## Likelihood computed by importance sampling
##      -2LL= 598.2609
##      AIC = 604.2609
##      BIC = 605.7157
## -----
# Alternate, fixing V and CL
saemix.model2<-saemixModel(model=model1cpt,
                           description="One-compartment model with first-order absorption",
                           psi0=matrix(c(1.,20,1),ncol=3,byrow=TRUE, dimnames=list(NULL, c("ka","V","CL")),
                           transform.par=c(1,1,1), fixed.estim=c(1,0,0),
                           covariance.model = diag(c(1,0,0)),
                           omega.init=diag(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)
## }
## <bytecode: 0x55d7ffec7af8>
## Nb of parameters: 3

```

```

##      parameter names: ka V CL
##      distribution:
##      Parameter Distribution Estimated
## [1,] ka      log-normal Estimated
## [2,] V      log-normal Fixed
## [3,] CL      log-normal Fixed
##      Variance-covariance matrix:
##      ka V CL
## ka  1 0 0
## V   0 0 0
## CL  0 0 0
##      Error model: constant , initial values: a.1=1
##      No covariate in the model.
##      Initial values
##      ka V CL
## Pop.CondInit 1 20 1
saemix.fit2<-saemix(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 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: 0x55d7ffec7af8>
##      Nb of parameters: 3
##      parameter names:  ka V CL
##      distribution:
##      Parameter Distribution Estimated
## [1,] ka      log-normal  Estimated
## [2,] V      log-normal  Fixed
## [3,] CL      log-normal  Fixed
##      Variance-covariance matrix:
##      ka V CL
## ka  1 0 0
## V   0 0 0
## CL  0 0 0
##      Error model: constant , initial values: a.1=1
##      No covariate in the model.
##      Initial values
##      ka V CL
## Pop.CondInit  1 20  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:  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(%)
## [1,] ka      0.16    0.018 11.8
## [2,] V      20.00    -      -
## [3,] CL      1.00    -      -
## [4,] a.1     4.09    0.278 6.8
## -----
## -----      Variance of random effects      -----
## -----

```

```
##      Parameter Estimate SE      CV(%)
## ka omega2.ka 0.0023    0.072 3143
## -----
## ----- Correlation matrix of random effects -----
## -----
##              omega2.ka
## omega2.ka 1
## -----
## ----- Statistical criteria -----
## -----
## Likelihood computed by linearisation
##      -2LL= 678.9567
##      AIC = 684.9567
##      BIC = 686.4114
##
## Likelihood computed by importance sampling
##      -2LL= 678.84
##      AIC = 684.84
##      BIC = 686.2948
## -----
```

```
# Checking estimates are close (yes)
saemix.fit@results
```

```
## Fixed effects
##      Parameter Estimate SE      CV(%)
## ka      0.0266    0.00153 5.77
## a.1      2.9266    0.19905 6.80
##
## Variance of random effects
##      Parameter Estimate SE      CV(%)
## omega2.ka 1.31e-05 0.0171 130416
##
## Statistical criteria
## Likelihood computed by linearisation
##      -2LL= 598.2676
##      AIC= 604.2676
##      BIC= 605.7223
## Likelihood computed by importance sampling
##      -2LL= 598.2609
##      AIC= 604.2609
##      BIC= 605.7157
```

```
saemix.fit2@results
```

```
## Fixed effects
## Warning in methods::show(x): NAs introduits lors de la conversion automatique
## Warning in methods::show(x): NAs introduits lors de la conversion automatique
##      Parameter Estimate SE      CV(%)
## ka      0.155    0.0183 11.8
## V      20.000    -      -
## CL      1.000    -      -
## a.1      4.093    0.2784 6.8
##
```

```
## Variance of random effects
## Parameter Estimate SE CV(%)
## omega2.ka 0.00228 0.0716 3143
##
## Statistical criteria
## Likelihood computed by linearisation
## -2LL= 678.9567
## AIC= 684.9567
## BIC= 686.4114
## Likelihood computed by importance sampling
## -2LL= 678.84
## AIC= 684.84
## BIC= 686.2948
```

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)
}

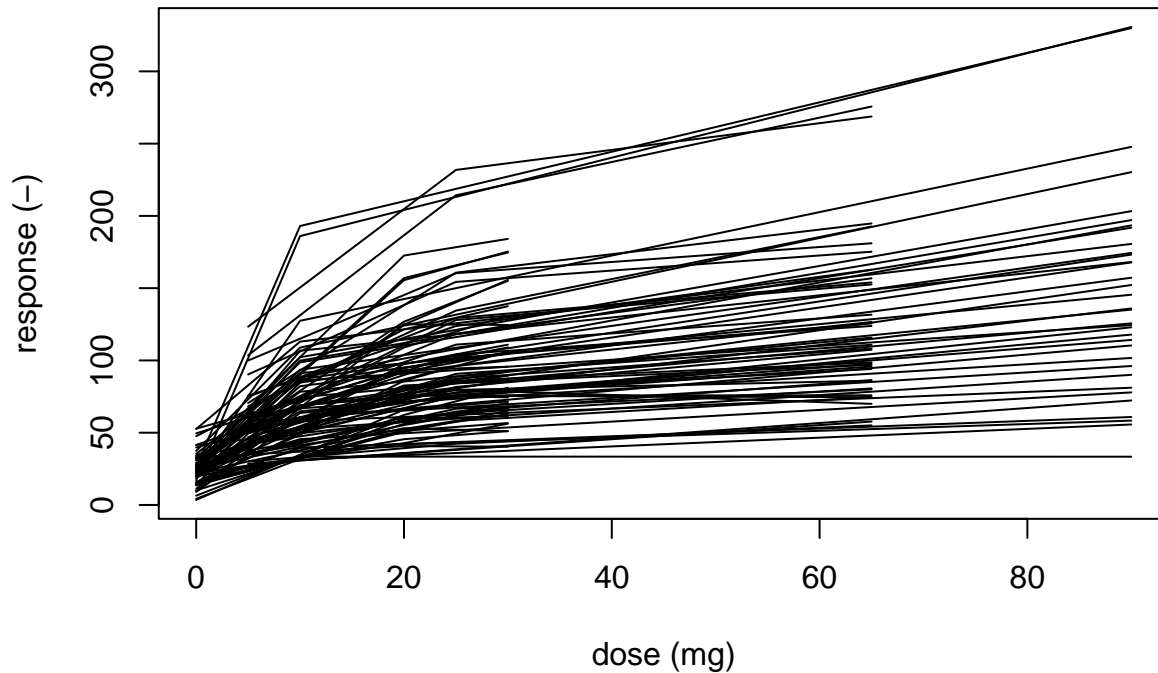
saemix.data<-saemixData(name.data=PD1.saemix,header=TRUE,name.group=c("subject"),
  name.predictors=c("dose"),name.response=c("response"),
  name.covariates=c("gender"), units=list(x="mg",y="-",covariates=c("-")))

##
##
## 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")
```

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: 0x55d7ffb7a6a8>
##   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: 0x55d7fffb7a6a8>
##   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:

```

```

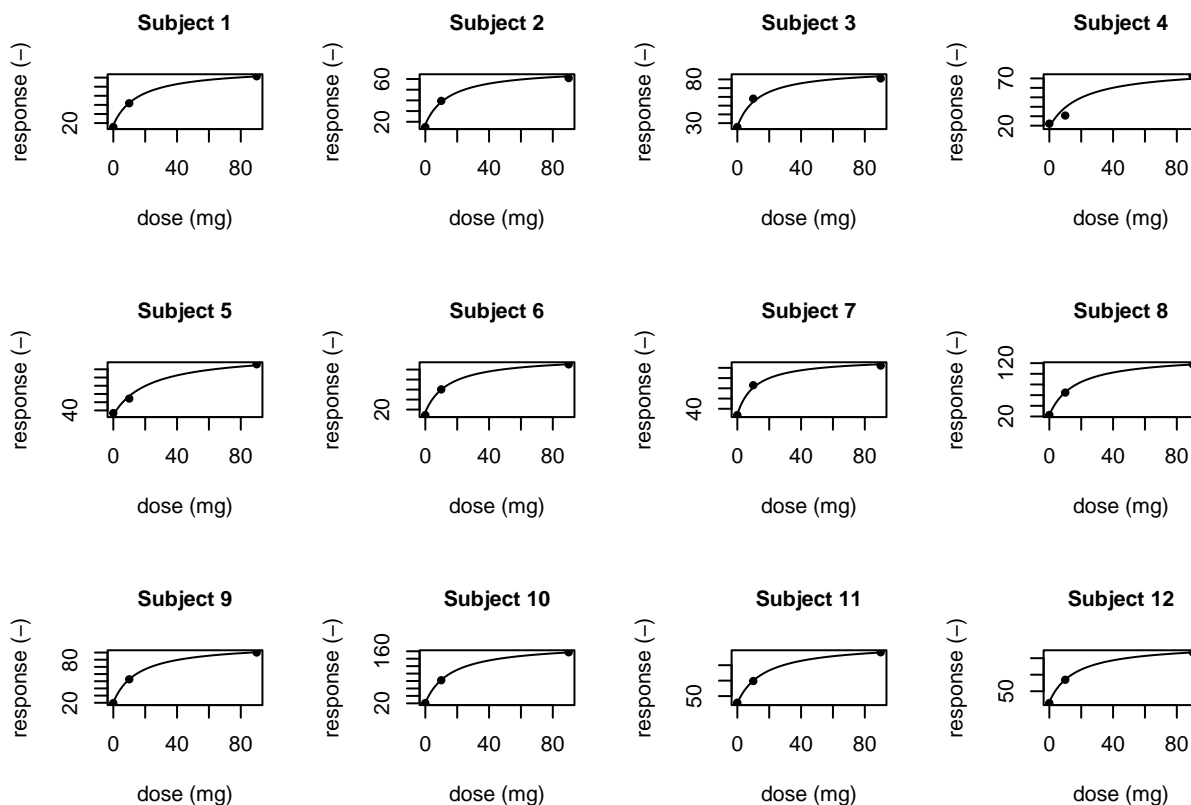
##          nb of simulated datasets used for npde: 1000
##          nb of simulated datasets used for VPC: 100
##      Input/output
##          save the results to a file: FALSE
##          save the graphs to files: FALSE
## -----
## ----- Results -----
## -----
## ----- Fixed effects -----
## -----
##      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 9.3e-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
plot(fit1, plot.type="individual", smooth=T, ilist=1:12)

## Simulating data using nsim = 1000 simulated datasets
## Computing WRES and npde .....

```



```
if(FALSE) {
  plot(model1, saemix.data)
  plot(model1, saemix.data, psi=c(0, 200, 50))
}
```

```
# Diagnostics
```

```
ynpde<-npdeSaemix(fit2)
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique
```

```
## Warning in which(!is.na(as.integer(object@name.covariates))): NAs introduits
## lors de la conversion automatique
```

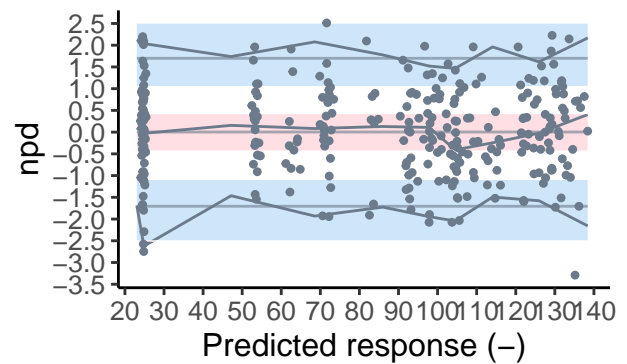
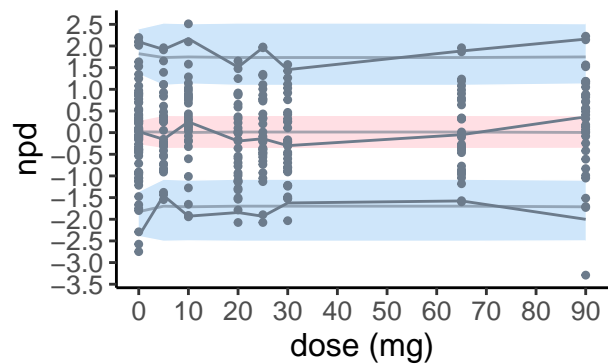
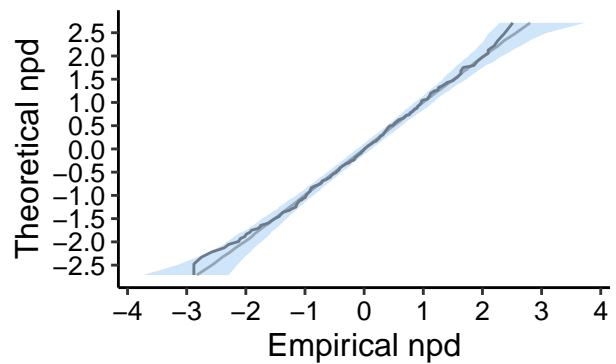
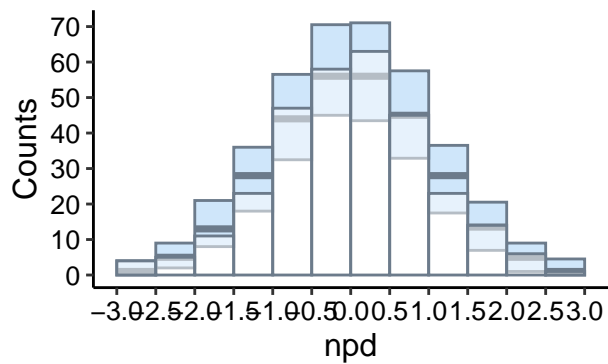
```
## -----
```

```
## Distribution of npde :
```

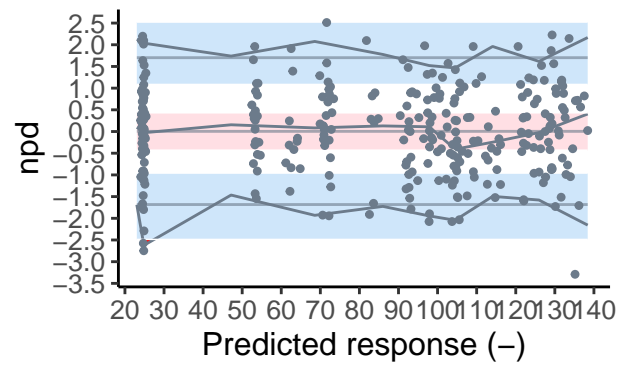
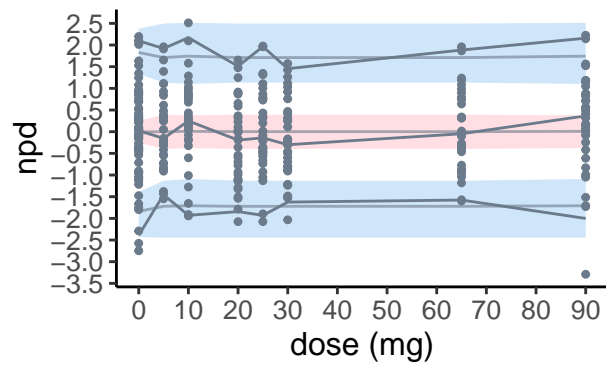
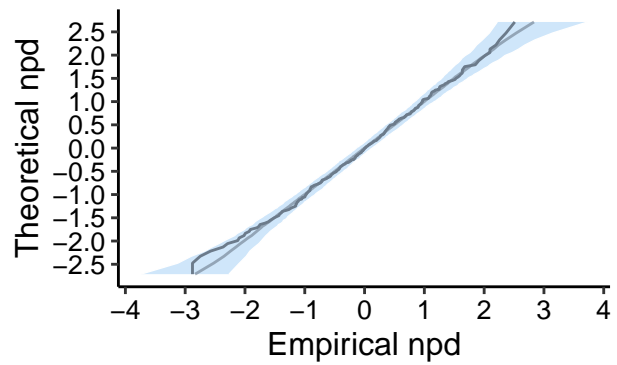
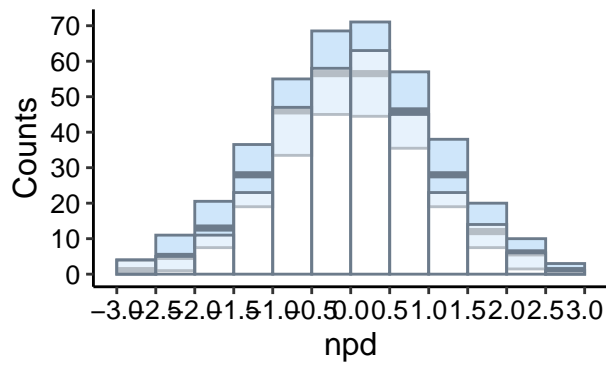
```
##      nb of obs: 300
```

```
##      mean= -0.007638   (SE= 0.057 )
```

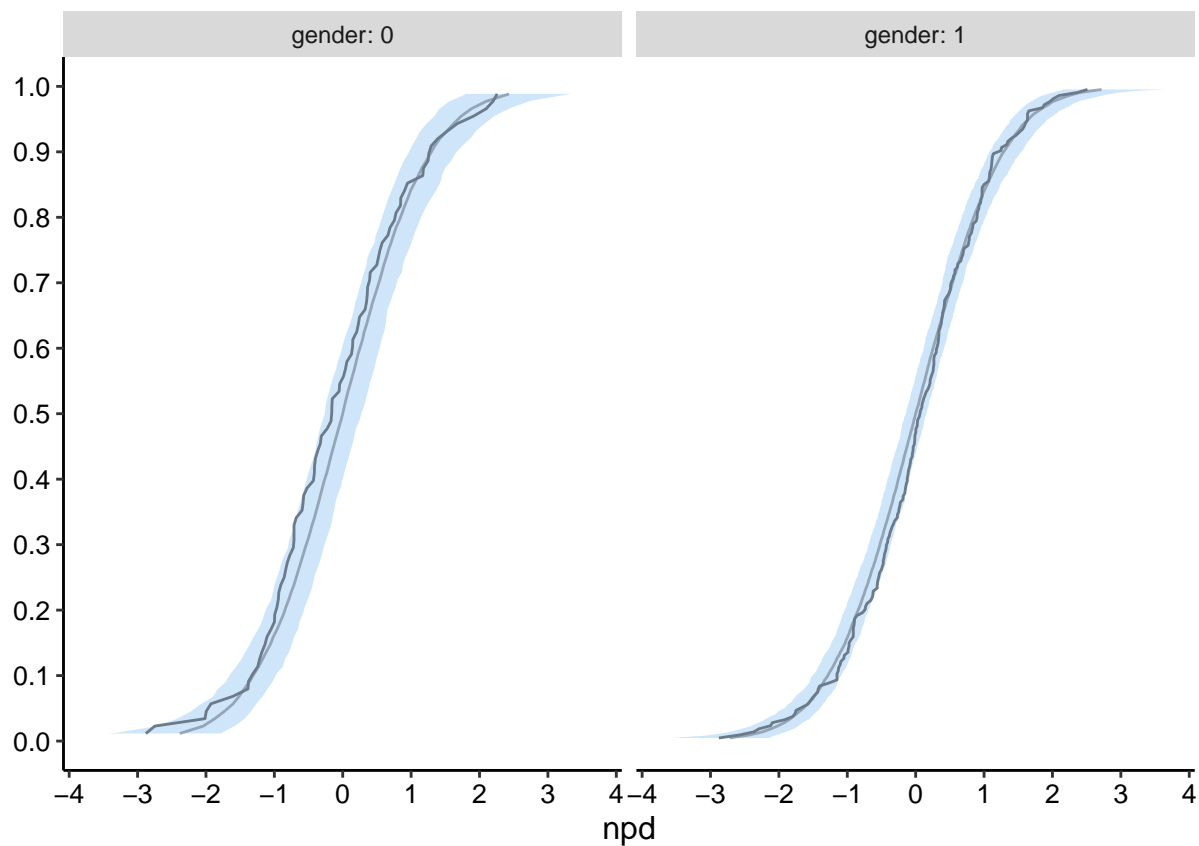
```
##      variance= 0.9914   (SE= 0.081 )
##      skewness= -0.1594
##      kurtosis= 0.05948
## -----
## Statistical tests (adjusted p-values):
##   t-test      : 1
##   Fisher variance test : 1
##   SW test of normality : 1
##   Global test  : 1
## ---
## Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
## -----
```



```
plot(ynpde)
```



```
# Splitting by covariates
plot(ynpde, plot.type="ecdf", which.cov="gender", covsplit=T)
```



```
# Better than the fit without covariates
```

```
ynpde1<-npdeSaemix(fit1)
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits  
## lors de la conversion automatique
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits  
## lors de la conversion automatique
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits  
## lors de la conversion automatique
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits  
## lors de la conversion automatique
```

```
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits  
## lors de la conversion automatique
```

```
## Warning in which(!is.na(as.integer(object@name.covariates))): NAs introduits  
## lors de la conversion automatique
```

```
## -----
```

```
## Distribution of npde :
```

```
##      nb of obs: 300
```

```
##      mean= -0.01346   (SE= 0.058 )
```

```
##      variance= 1.021   (SE= 0.083 )
```

```
##      skewness= -0.02993
```

```
##      kurtosis= -0.06712
```

```
## -----
```

```
## Statistical tests (adjusted p-values):
```

```
##      t-test           : 1
```

```
##      Fisher variance test : 1
```

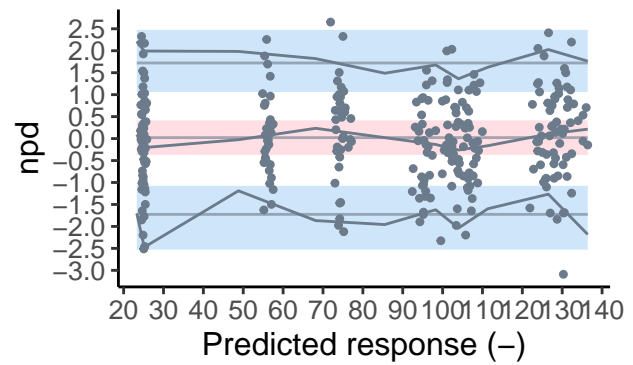
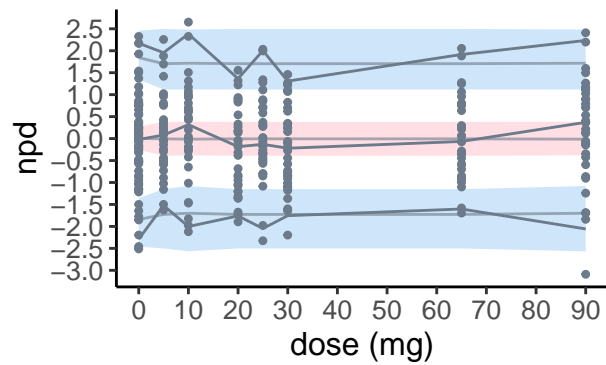
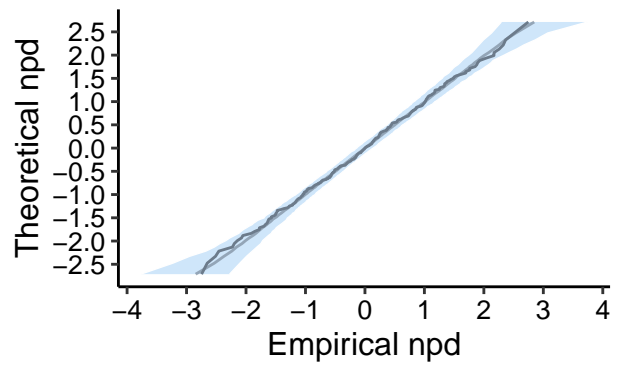
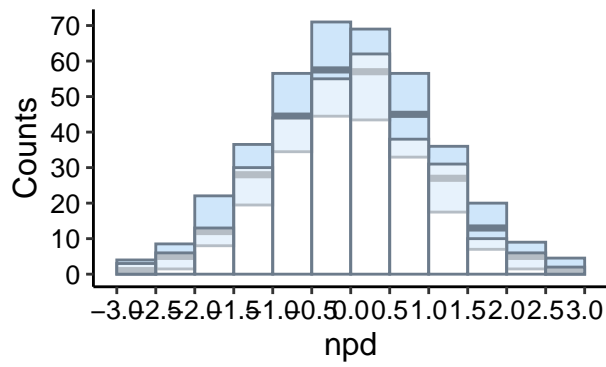
```
##      SW test of normality : 1
```

```
##      Global test        : 1
```

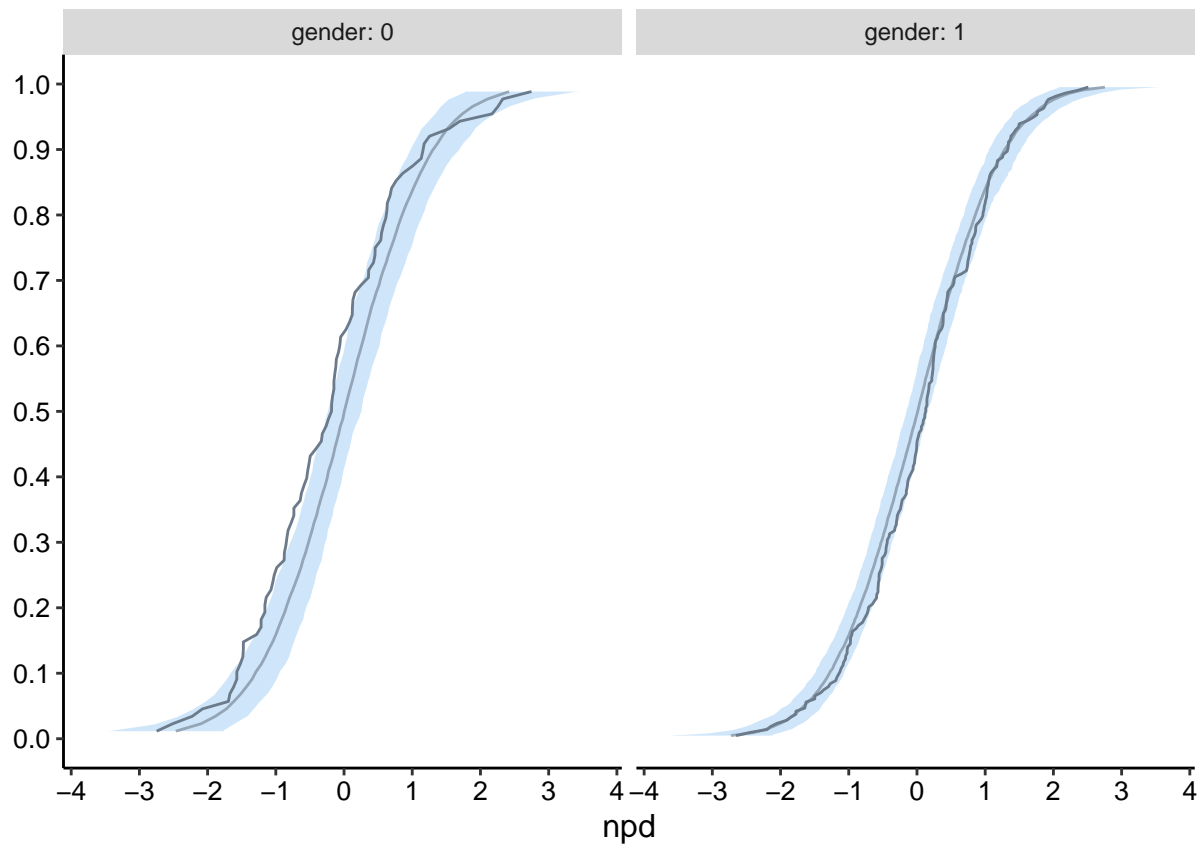
```
## ---
```

```
## Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
```

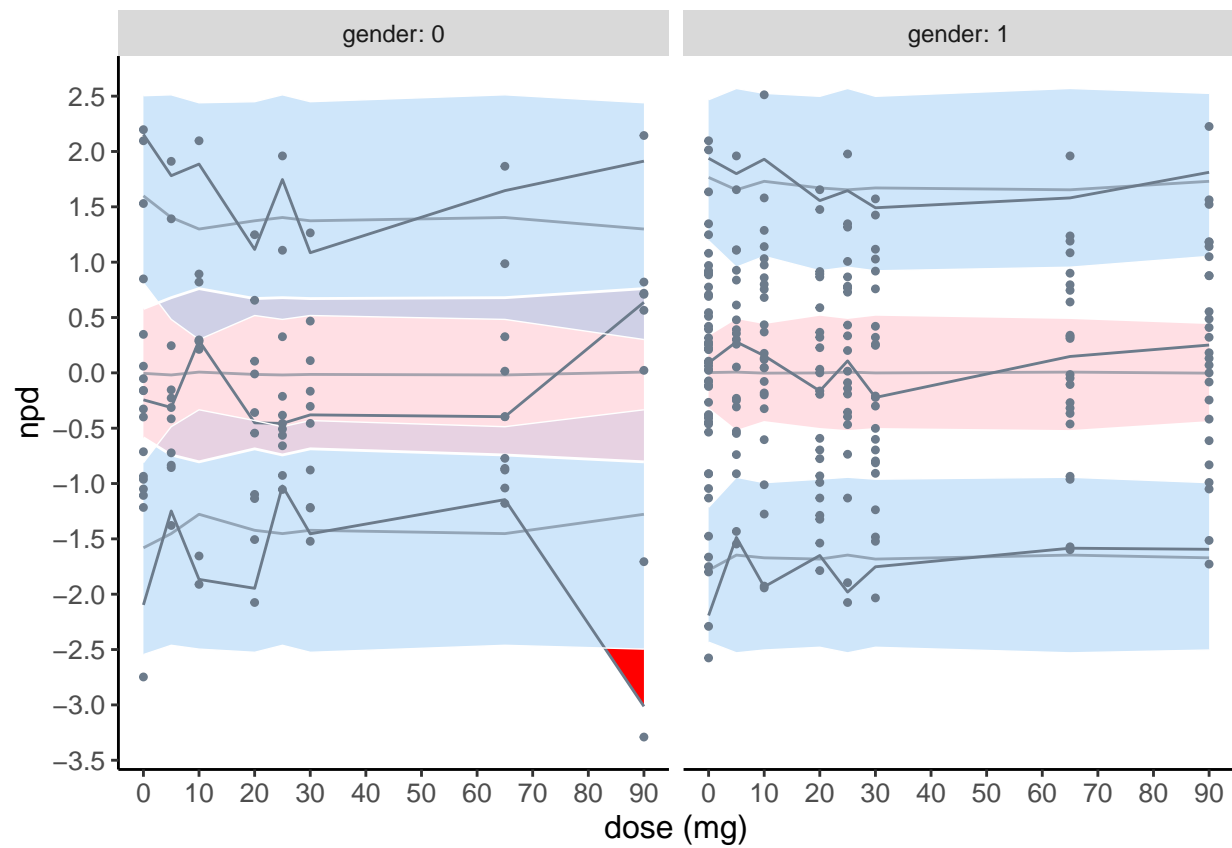
```
## -----
```

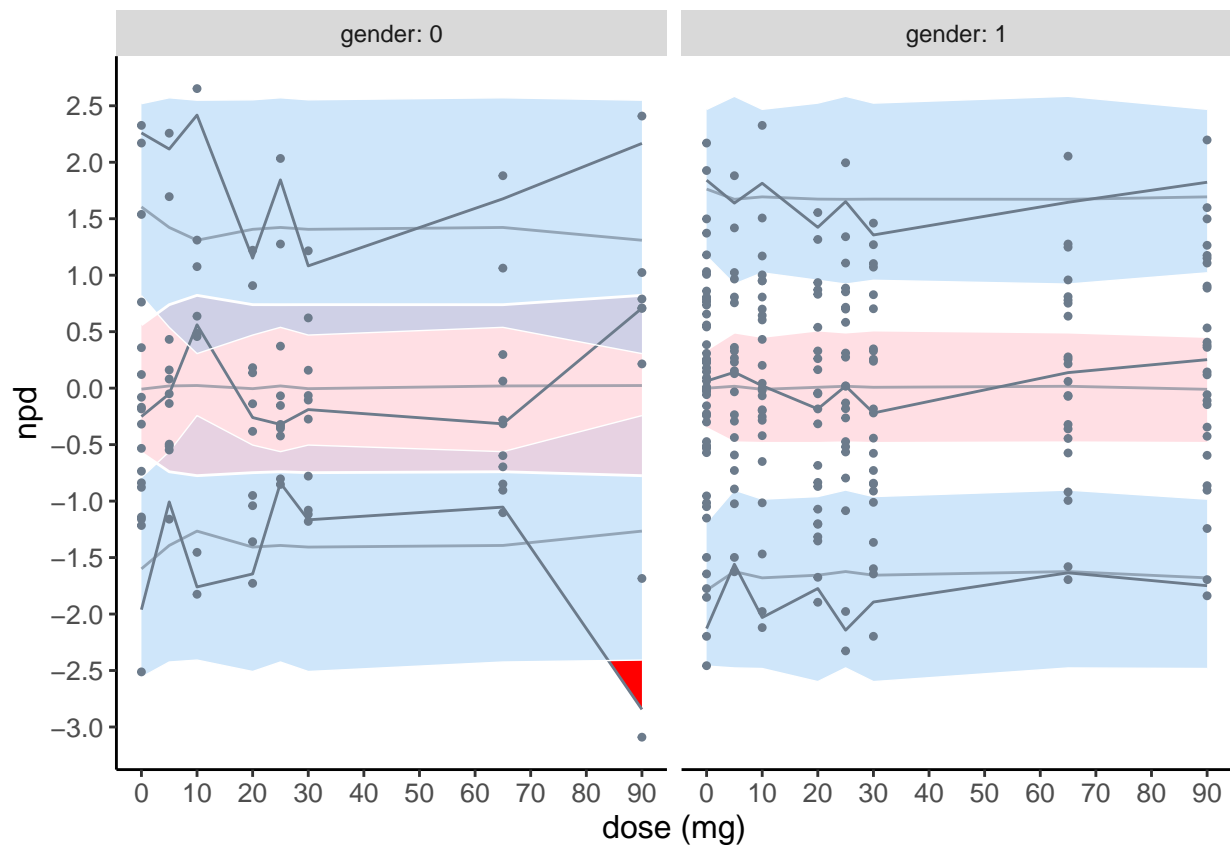
```
plot(ynpde1, plot.type="ecdf", which.cov="gender", covsplit=T)
```



```
# Similar scatterplots for both models
plot(ynpde, plot.type="x.scatter", which.cov="gender", covsplit=T)
```



```
plot(ynpde1, plot.type="x.scatter", which.cov="gender", covsplit=T)
```



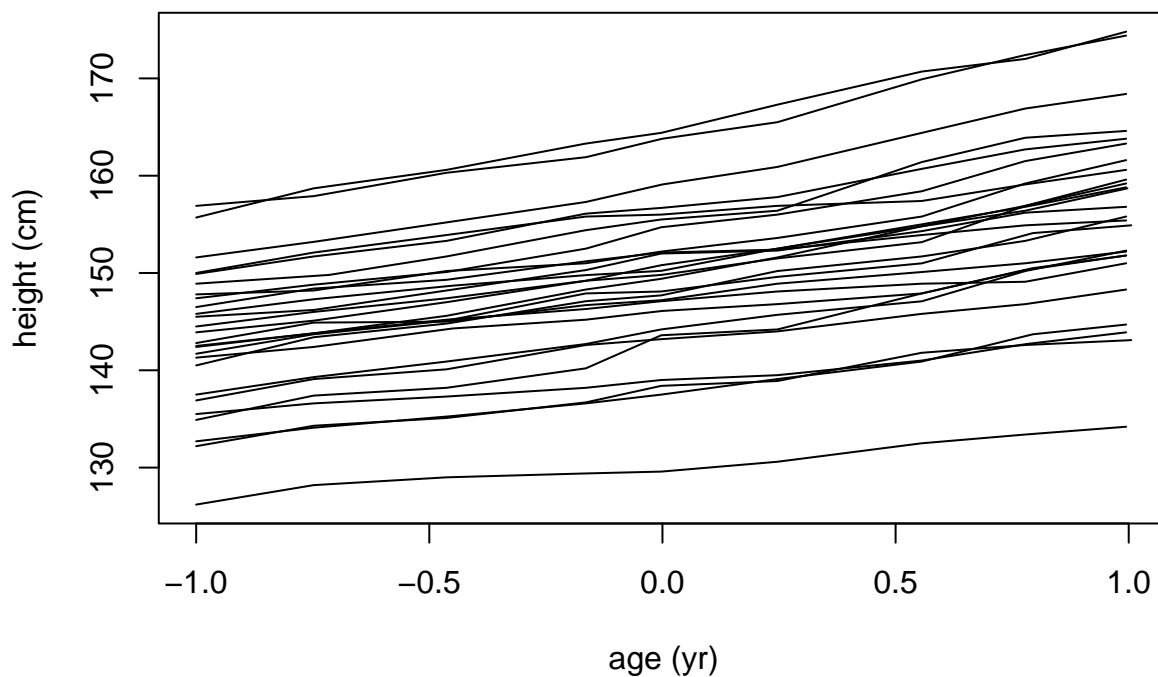
Oxford boys

```
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: 0x55d801eea588>

```

```

##   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
## -----

ynpde<-npdeSaemix(saemix.fit)

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

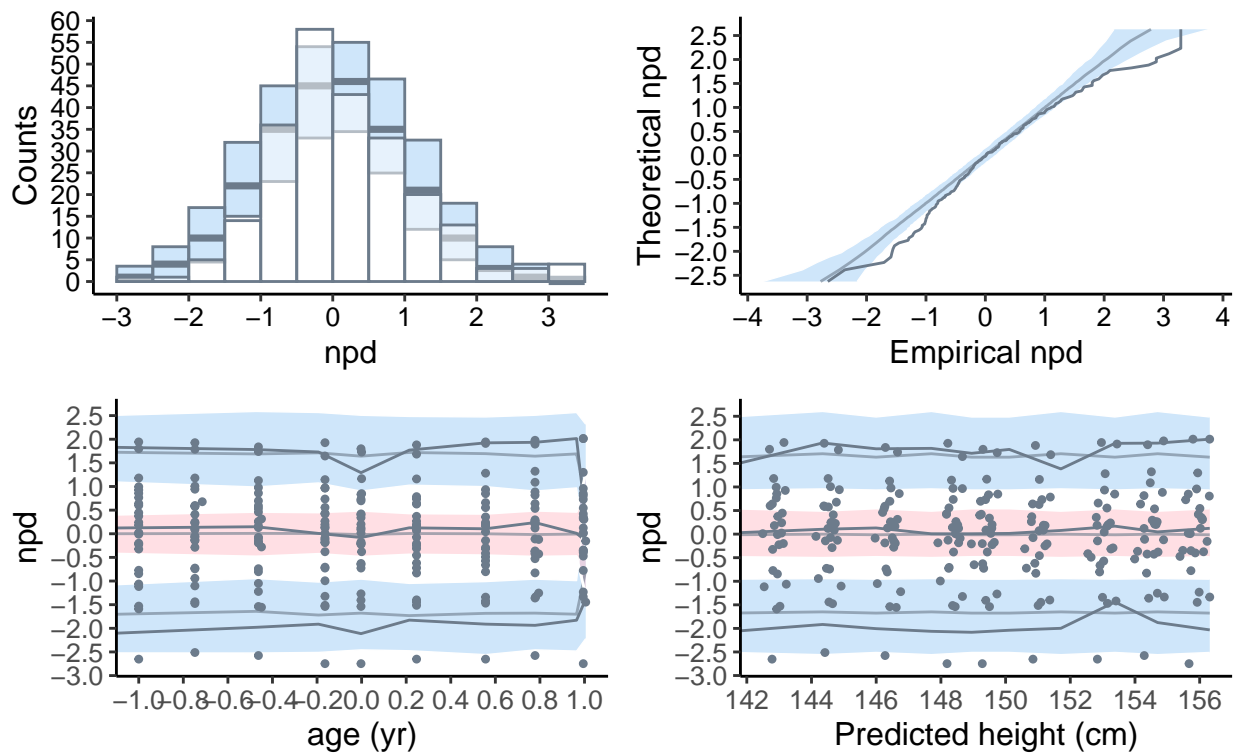
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

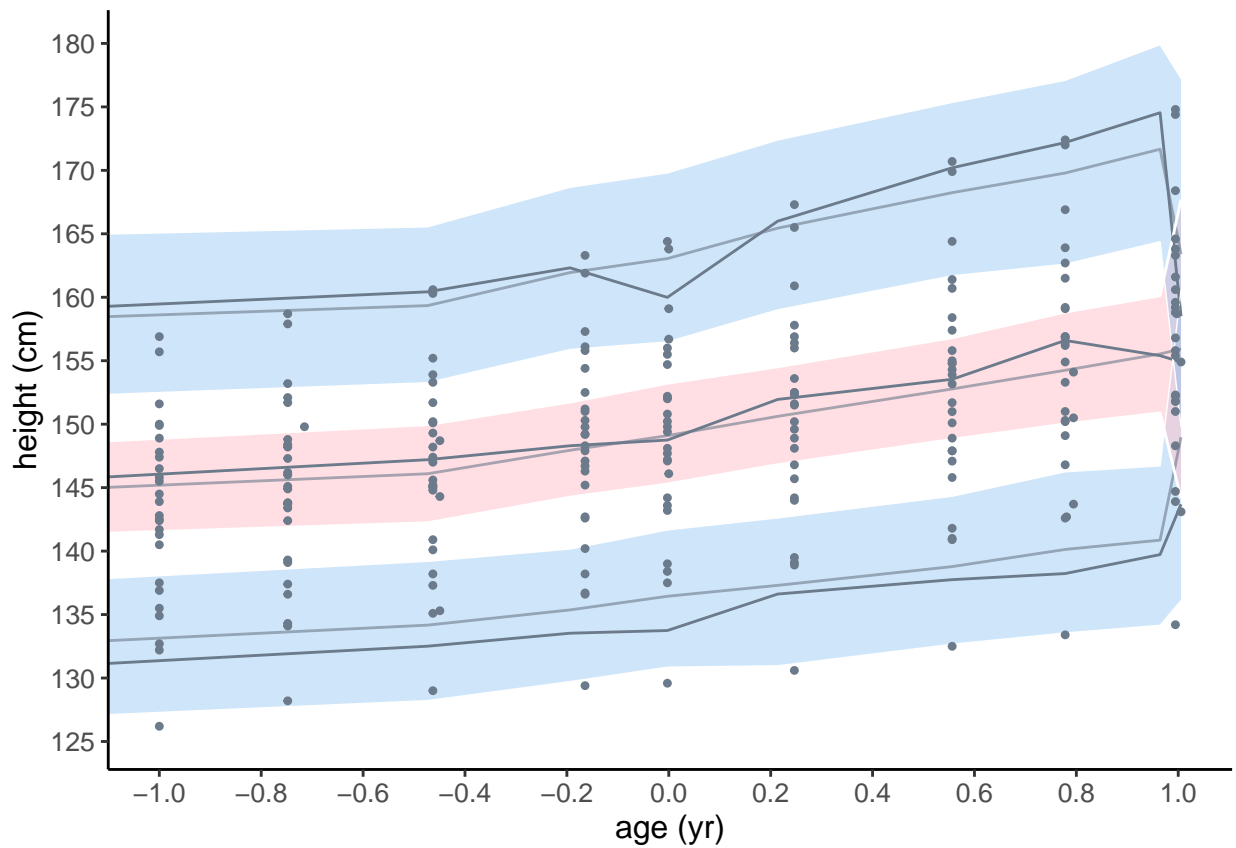
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

## -----
## Distribution of npde :
##      nb of obs: 234
##      mean= 0.1538 (SE= 0.066 )
##      variance= 1.014 (SE= 0.094 )
##      skewness= 0.6498
##      kurtosis= 0.8492
## -----
## Statistical tests (adjusted p-values):
##      t-test          : 0.0609 .
##      Fisher variance test : 1
##      SW test of normality : 0.000177 ***
##      Global test       : 0.000177 ***
## ---
## Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
## -----

```



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




```

# Predictors given as a dataframe
ypred <- predict.SaemixModel(saemix.model, saemix.data@data[,saemix.data@name.predictors, drop=FALSE])

# Predictors extracted from a saemixData object
ypred2 <- predict.SaemixModel(saemix.model, saemix.data)

# Summary of
cat("Check that the predictions don't give any NA and are reasonable compared to the observations\n")

## Check that the predictions don't give any NA and are reasonable compared to the observations
summary(ypred$predictions)

##           id           age           pred
##  Min.      :1    Min.     :-1.00000    Min.      :139.0
## 1st Qu.:1    1st Qu.: -0.46300    1st Qu.:139.5
##  Median :1    Median : -0.00270    Median :140.0
##   Mean  :1     Mean  : 0.02263     Mean  :140.0
## 3rd Qu.:1    3rd Qu.: 0.55620     3rd Qu.:140.6
##   Max.  :1     Max.   : 1.00550     Max.   :141.0

summary(saemix.data@data[,saemix.data@name.response])

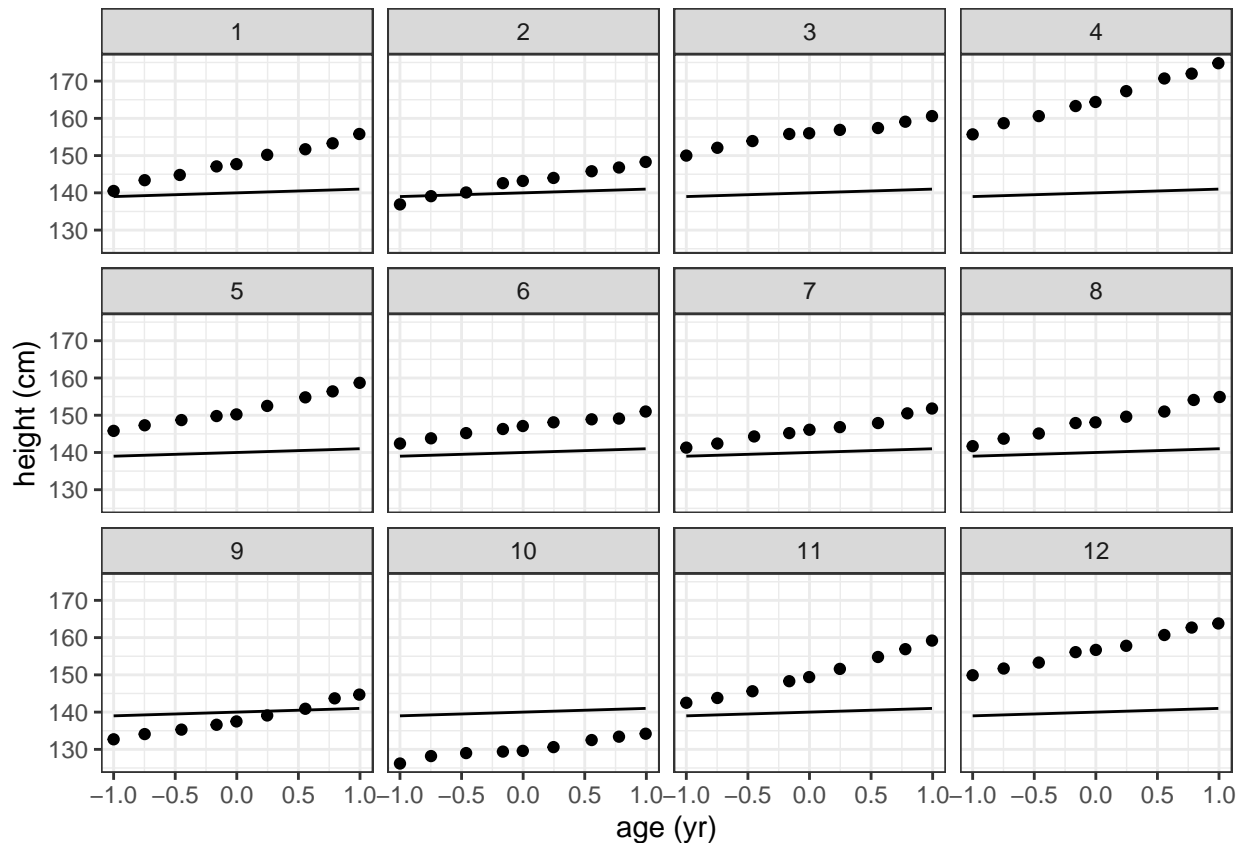
##      Min. 1st Qu.  Median      Mean 3rd Qu.      Max.
##    126.2  143.8   149.5   149.5   155.5   174.8

summary(ypred2$predictions$pred-ypred$predictions$pred)

##      Min. 1st Qu.  Median      Mean 3rd Qu.      Max.
##         0         0         0         0         0         0

# Graphs
checkInitialFixedEffects(saemix.model, saemix.data)

```



```
summary(predict(saemix.fit))
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      126.5  143.6   149.8   149.5   155.4   174.3
```

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","-","-")))
```

```
##
##
## 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")

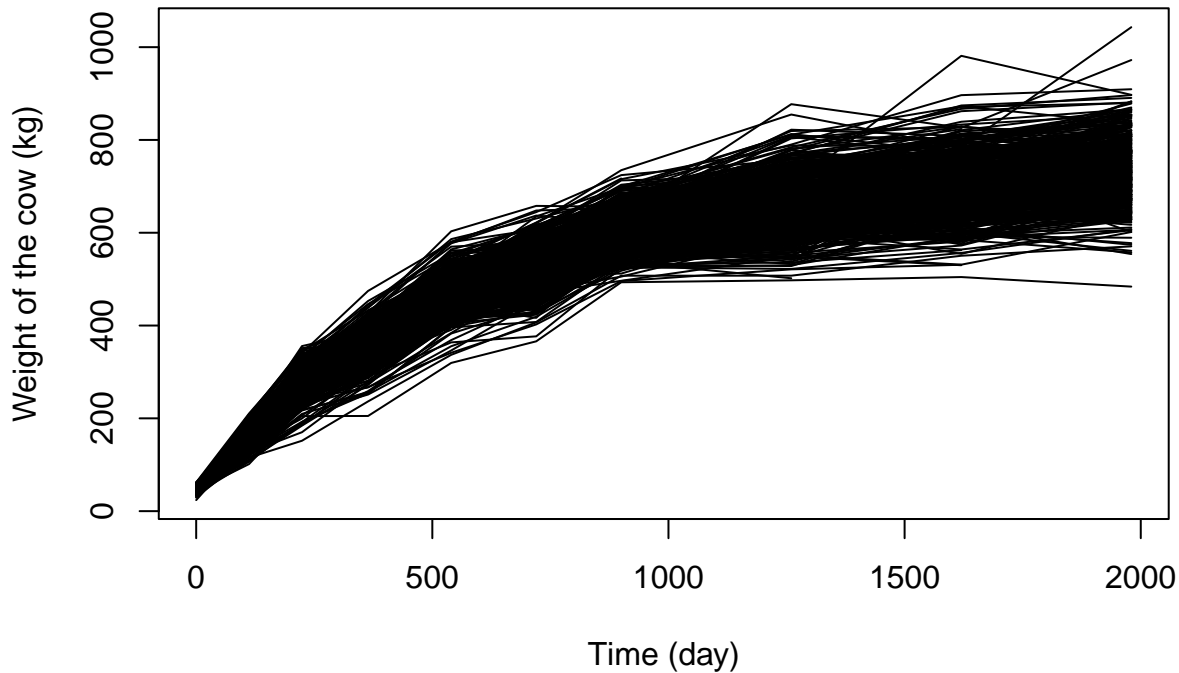
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
## Model function: Exponential growth model
## Model type: structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   a<-psi[id,1]
##   b<-psi[id,2]
##   k<-psi[id,3]
##   f<-a*(1-b*exp(-k*x))
##   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: 0x55d7fc497df8>
## 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")))

##
##
## 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")

```

```

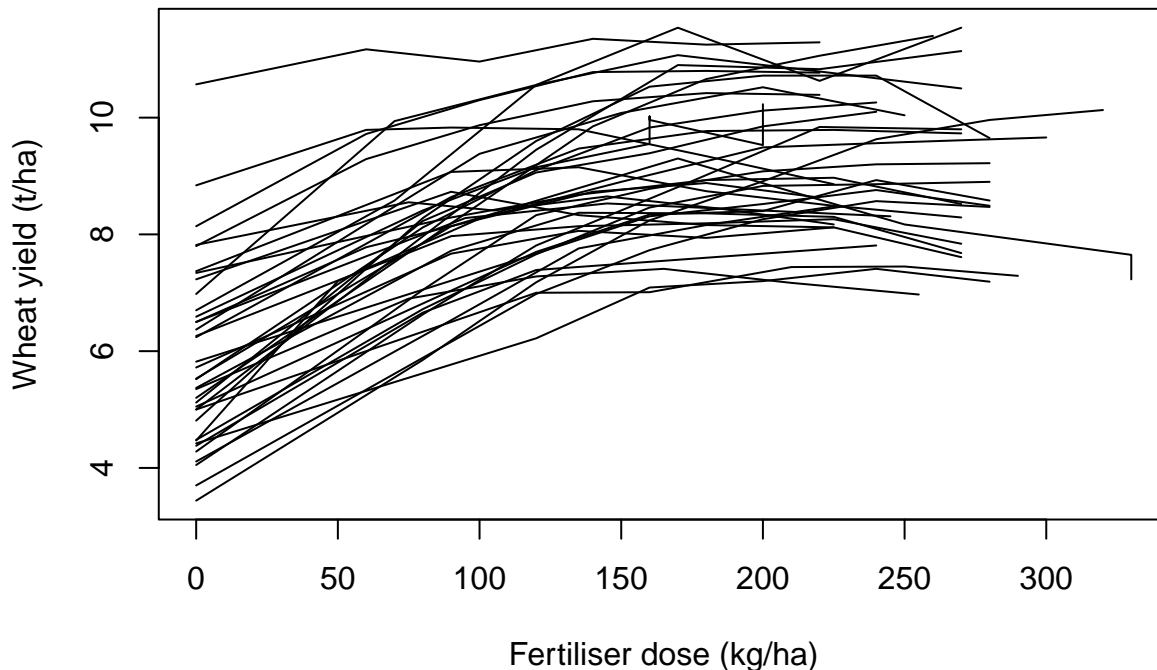
##
##
## The following SaemixModel object was successfully created:
##
## Nonlinear mixed-effects model
##   Model function:  Linear plus plateau model
##   Model type:     structural
## function(psi,id,xidep) {
##   x<-xidep[,1]
##   ymax<-psi[id,1]
##   xmax<-psi[id,2]
##   slope<-psi[id,3]
##   f<-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
```

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



```
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
## 145 931 0 5.12 105 0 0 1 1
## 146 931 80 8.23 105 0 0 1 1
## 147 931 120 9.06 105 0 0 1 1
## 148 931 160 9.39 105 0 0 1 1
## 149 931 200 9.85 105 0 0 1 1
```



```

## 150  931  240 10.10          105  0  0  1  1
## 151  932   0  4.47           88  0  0  1  1
## 152  932   50  7.20           88  0  0  1  1
## 153  932  100  8.27           88  0  0  1  1
## 154  932  150  8.90           88  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: 0x55d7fd4a2490>
##   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          8.88  0.175  2.0
## [2,] Xmax         13.41  3.265 24.3
## [3,] slope          0.22  0.056 25.4
## [4,] a.1           0.70  0.040  5.8
## -----
## ----- Variance of random effects -----
## -----
##      Parameter      Estimate SE      CV(%)
## Ymax  omega2.Ymax  1.0335    0.2622   25
## Xmax  omega2.Xmax  0.0716   13.9406 19465
## slope omega2.slope 0.0067    0.0048   71
## -----
## ----- 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= 616.5778
##      AIC = 630.5778
##      BIC = 641.8542
##
## Likelihood computed by importance sampling
##      -2LL= 616.4578
##      AIC = 630.4578
##      BIC = 641.7342
## -----
# 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= -308.2289
## LL by linearisation, LL_lin= -308.2889
## LL by Gaussian Quadrature, LL_GQ= -308.3099
# 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: 0x55d7fd4a2490>
## 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
## 145 931 0 5.12 105 0 0 1 1
## 146 931 80 8.23 105 0 0 1 1
## 147 931 120 9.06 105 0 0 1 1
## 148 931 160 9.39 105 0 0 1 1
## 149 931 200 9.85 105 0 0 1 1
## 150 931 240 10.10 105 0 0 1 1
## 151 932 0 4.47 88 0 0 1 1
## 152 932 50 7.20 88 0 0 1 1
## 153 932 100 8.27 88 0 0 1 1
## 154 932 150 8.90 88 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: 0x55d7fd4a2490>
## 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                   9.179   0.1908  2.1  -
## [2,] Xmax                   217.787  15.6758  7.2  -
## [3,] beta_soil.nitrogen(Xmax) -1.104   0.1712 15.5  1.1e-10
## [4,] slope                   0.026   0.0013  4.8  -
## [5,] a.1                     0.303   0.0193  6.4  -
## -----
## ----- Variance of random effects -----
## -----
##      Parameter      Estimate SE      CV(%)
## Ymax  omega2.Ymax  1.3e+00  3.1e-01 24
## Xmax  omega2.Xmax  1.0e+03  2.9e+02 28
## slope omega2.slope 3.2e-05  1.2e-05 37
## -----
## ----- 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= 388.7811
##      AIC = 404.7811
##      BIC = 417.6684
##
## Likelihood computed by importance sampling
##      -2LL= 380.854
##      AIC = 396.854
##      BIC = 409.7413

```

```

## -----
# 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= 641.7342
## Model with covariate, BIC= 409.7413
##      LRT: p= 0

# Diagnostics
ynpde<-npdeSaemix(saemix.fit2)

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

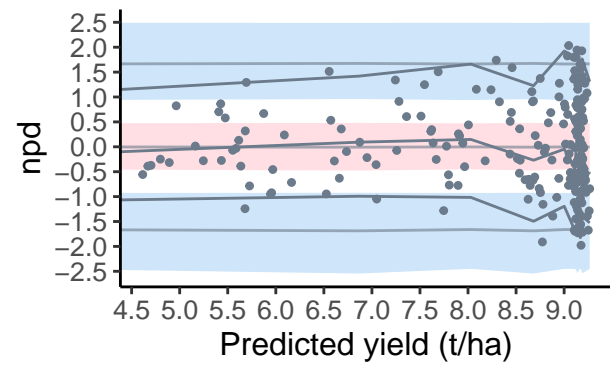
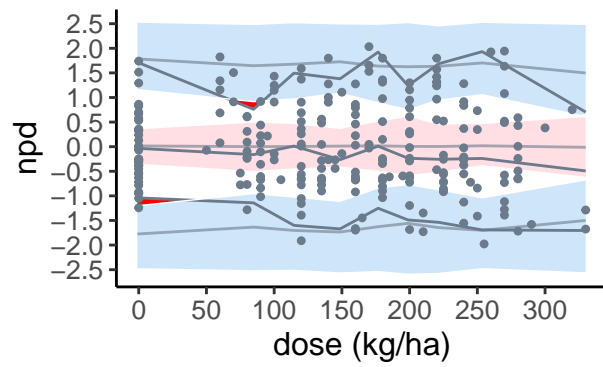
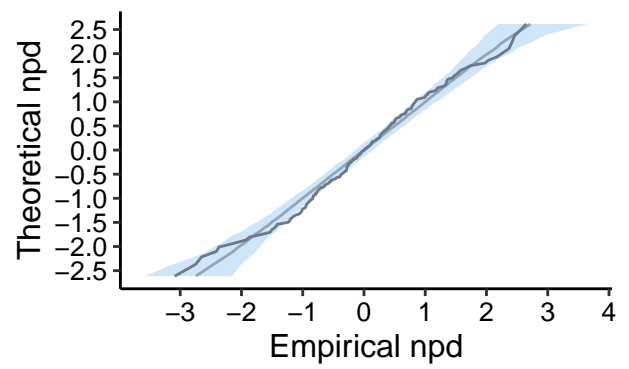
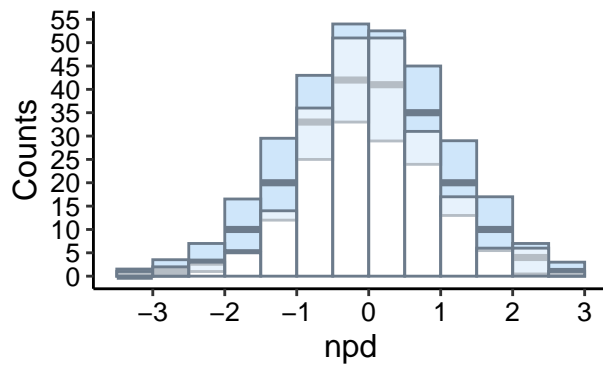
## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

## Warning in read(x, dat, detect = detect, verbose = verbose): NAs introduits
## lors de la conversion automatique

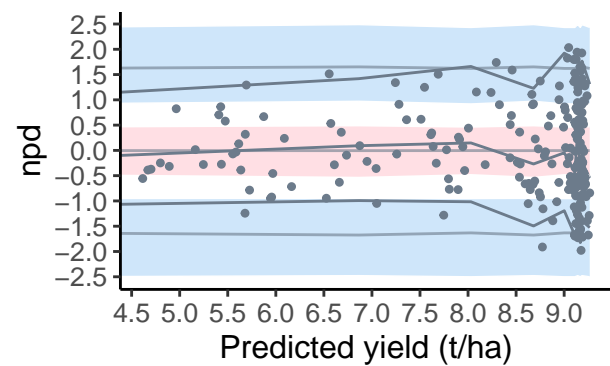
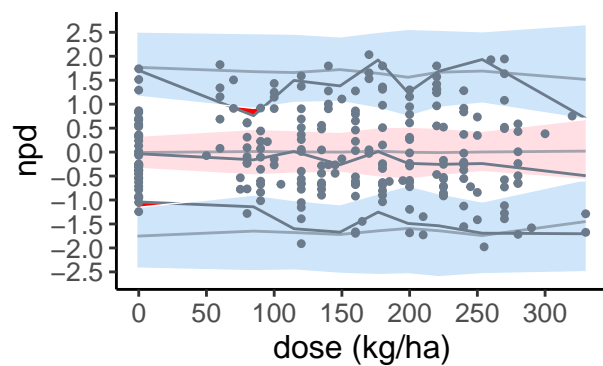
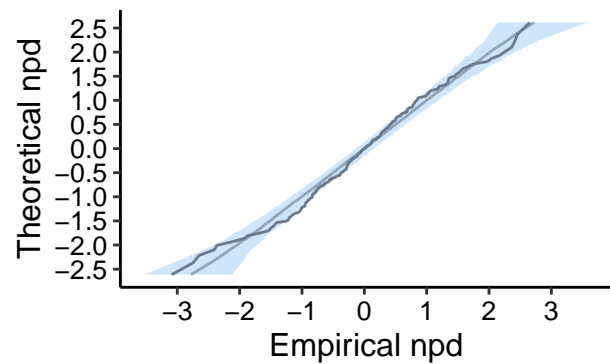
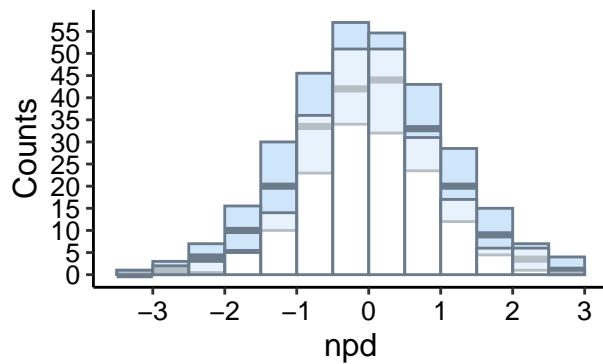
## Warning in which(!is.na(as.integer(object@name.covariates))): NAs introduits
## lors de la conversion automatique

## -----
## Distribution of npde :
##      nb of obs: 224
##      mean= 0.01499   (SE= 0.063 )
##      variance= 0.8981 (SE= 0.085 )
##      skewness= -0.0838
##      kurtosis= 0.8497
## -----
## Statistical tests (adjusted p-values):
##      t-test          : 1
##      Fisher variance test : 0.837
##      SW test of normality : 0.0973 .
##      Global test       : 0.0973 .
## ---
## Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
## -----

```

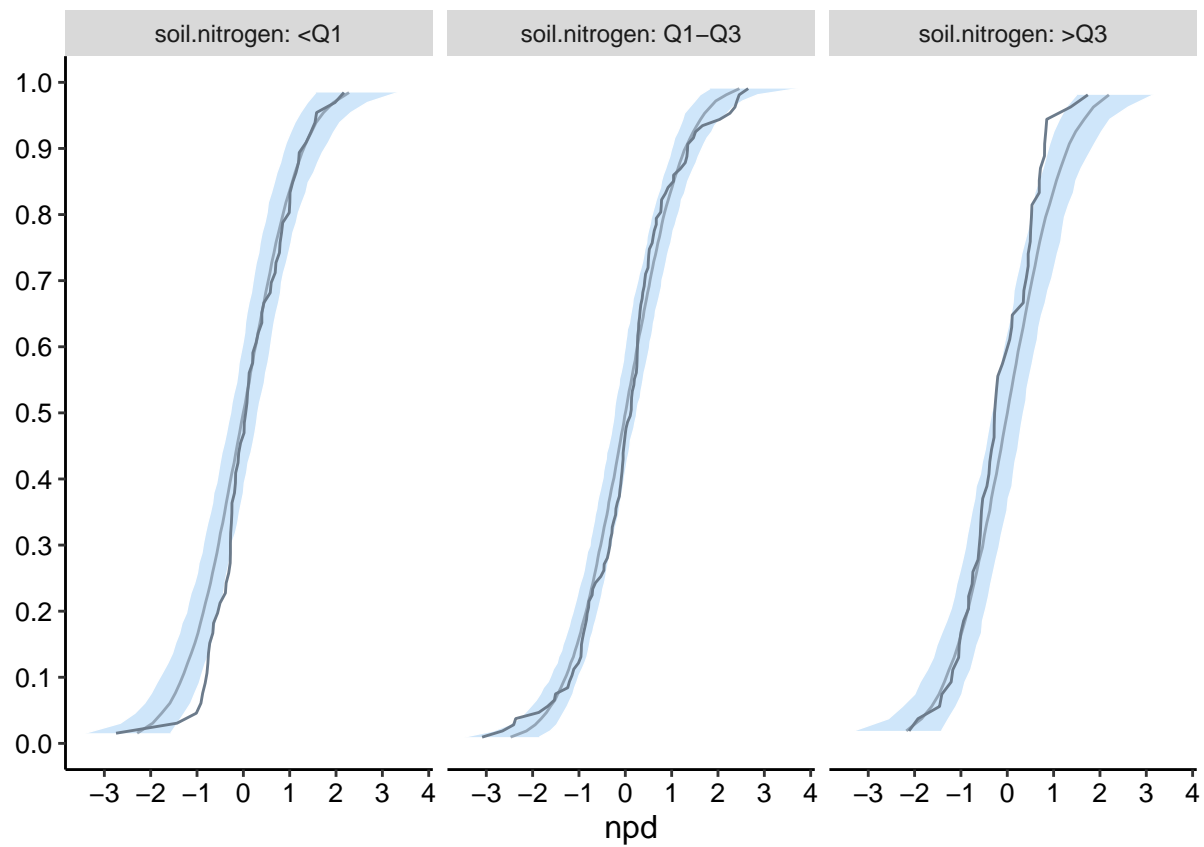


```
plot(ynpde)
```



```
# Splitting by covariates
```

```
plot(ynpde, plot.type="ecdf", which.cov="soil.nitrogen", covsplit=T)
```



Exiting

```
if(testMode) {  
  dev_mode()  
}
```