`Saemix`: Open Source R package for mixed effects modeling

Marc Lavielle, Emmanuelle Comets, Audrey Lavenu and Belhal Karimi

2020-01-31

Contents

Welcome to mixed effects modeling in R

The saemix project is an R package available in CRAN that implements the Stochastic Approximation of the EM (SAEM) algorithm introduced in (Kuhn and Lavielle, 2004). This algorithm is state-of-the-art method for fitting, possibly non linear, models in agronomy, animal breeding or Pharmacokinetics-Pharmacodynamics (PKPD) analysis.

Thus far, the main area using the package thus far is Pharmacology, especially to understand how drugs, under development, behave in the body or how the body reacts to a drug during clinical trials but we ought to aim at a more general audience of biostatisticians dealing with nonlinear mixed effects modeling.


Saemix

Chapter 1

Introduction

Longitudinal data arise in many fields, such as agronomy, spatial analysis, imagery, clinical trials, and have been particularly prominent in the field of pharmacokinetics (PK) and pharmacodynamics (PD), where increasingly complex models involving mechanistic and empirical processes have been developed to describe the time course of and responses to drugs. Nonlinear models pose unique challenges in terms of estimation methods, and have driven the research to provide better estimation of parameters as well as the associated uncertainty, diagnostics of model misspecification and more informative designs. The SAEM algorithm, based on two highly cited publications by one of our project members Marc Lavielle, see (Delyon et al., 1999) and (Kuhn and Lavielle, 2004), was implemented in R in 2011 in the `saemix` R package~(Comets et al., 2017). Several applications of SAEM in agronomy, animal breeding and PKPD analysis have been published using `saemix`.

PK/PD analyses are now a fundamental element of the registration file submitted to health authority for the approval of new drugs, but NLMEM are also increasingly applied to other areas. In clinical trials, they complement the point analyses by offering a unique understanding of the evolution of disease or treatment action. In cohort studies, they allow to model trajectories such as growth or cognitive decline. Joint models are now routinely used to link the evolution of markers with the occurrence of an event. Making use of S4 classes and methods to provide user-friendly interaction, `saemix` provides a new maximum likelihood estimation tool with a powerful exact algorithm to the R community.

Chapter 2

Installation

`saemix` can be installed and used on several platforms. Installation can range from easy to challenging, depending on the platform. We are in the process of streamlining this process, and any help or suggestions are greatly appreciated!

2.1 `saemix`

Information on how to install 'saemix' and its dependencies on different platforms can be found on the `saemix` pkgdown site. Separate information can be found on `RxODE` pkgdown site.

2.1.1 Installation via GitHub

To Complete

2.2 `shinyMixR`: project management tool

A user-friendly tool was developed for `saemix` based on `Shiny`

Chapter 3

Materials

3.1 User's Guide

- Saemix User's Guide: PDF

3.2 Posters and Presentations

- PAGE 2011, Athens, Greece: PosterPAGE

Various other publications can be found here.

Chapter 4

Case Studies

Some basic Case Studies are demonstrated in this chapter; the vignettes will be discussing the application in more depth.

4.1 A two-compartment PK model

```
library(saemix)
?saemix
```

Read the Data

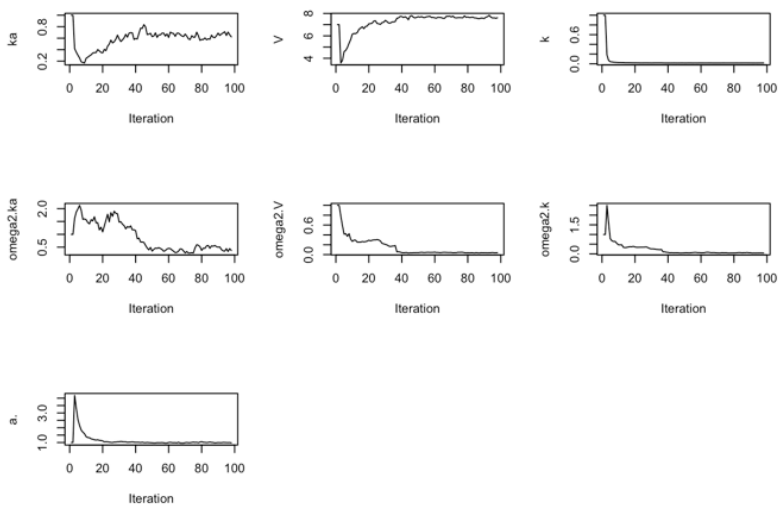
```
warfa_data <- read.table("data/warfarin_data.txt", header=T)
saemix.data<-saemixData(name.data=warfa_data,header=TRUE,sep=" ",
  na=NA, name.group=c("id"),name.predictors=c("amount","time"),
  name.response=c("y1"), name.X="time")
```

Create the Model

saemix models are contained in a R function with one blocks:

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

```
## Running main SAEM algorithm
## [1] "Fri Jan 31 10:52:18 2020"
## .
```




```
saemix.model<-saemixModel(model=model1cpt,description="warfarin",
  type="structural",psi0=matrix(c(1,7,1,0,0,0),ncol=3,byrow=TRUE,
  dimnames=list(NULL, c("ka","V","k"))),transform.par=c(1,1,1),
  omega.init=matrix(c(1,0,0,0,1,0,0,0,1),ncol=3,byrow=TRUE),
  covariance.model=matrix(c(1,0,0,0,1,0,0,0,1),ncol=3,
  byrow=TRUE))
```

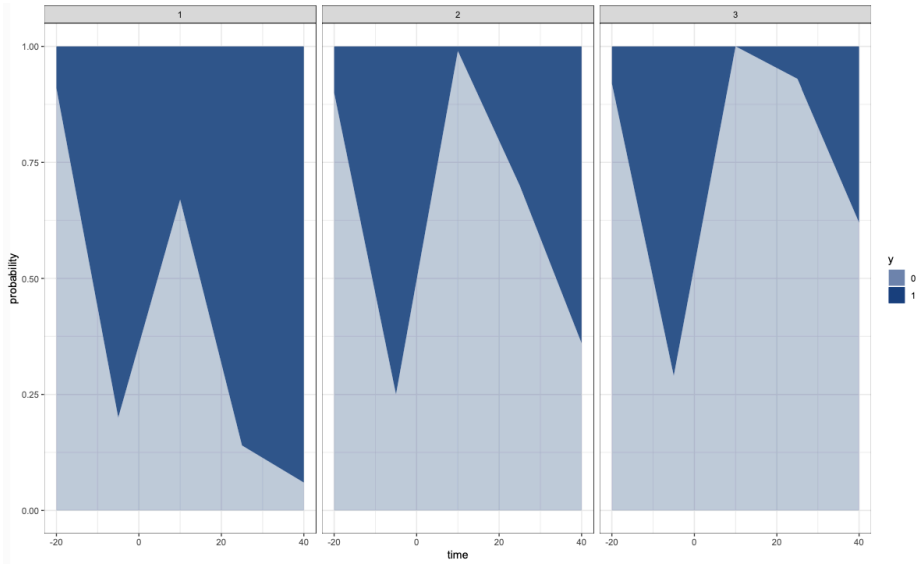
Run the SAEM algorithm

```
K1 = 200
K2 = 100


#Run SAEM
options<-list(seed=39546,map=F,fim=F,ll.is=F,
  nbiter.mcmc = c(2,2,2), nbiter.saemix = c(K1,K2),nbiter.sa=0,
  displayProgress=TRUE,save.graphs=FALSE,nbiter.burn =0)
fit<-saemix(saemix.model,saemix.data,options)
```

4.2 A categorical data model with regression variables

4.2.1 mlxR: simulate synthetic data

```
library("mlxR")
catModel <- inlineModel(
"[LONGITUDINAL]
input =  {beta0,gamma0,delta0, dose}
dose = {use=regressor}
EQUATION:
lm0 = beta0+gamma0*t + delta0*dose
D = exp(lm0)+1
p0 = exp(lm0)/D
p1 = 1/D

DEFINITION:
y = {type=categorical, categories={0, 1},
     P(y=0)=p0,
     P(y=1)=p1}
[INDIVIDUAL]
input={beta0_pop, o_beta0,
      gamma0_pop, o_gamma0,
      delta0_pop, o_delta0}
DEFINITION:
beta0  ={distribution=normal, prediction=beta0_pop,  sd=o_beta0}
gamma0  ={distribution=normal, prediction=gamma0_pop,  sd=o_gamma0}
delta0  ={distribution=normal, prediction=delta0_pop,  sd=o_delta0} ")

nobs = 15
tobs<- seq(-20, 50, by=nobs)
reg1 <- list(name='dose',
             time=tobs,
             value=10*(tobs>0))

reg2 <- list(name='dose',
             time=tobs,
             value=20*(tobs>0))

reg3 <- list(name='dose',
             time=tobs,
             value=30*(tobs>0))

out  <- list(name='y', time=tobs)
```




```
N  <- 100
p <- c(beta0_pop=-4, o_beta0=0.3,
       gamma0_pop= -0.5, o_gamma0=0.2,
       delta0_pop=1, o_delta0=0.2)

g1 <- list(size=N,regressor = reg1)
g2 <- list(size=N,regressor = reg2)
g3 <- list(size=N,regressor = reg3)
g <- list(g1,g2,g3)
res <- simulx(model=catModel,output=out, group=g,parameter=p)
plot1 <- catplotmlx(res$y)
```

4.2.2 saemix: fit the noncontinuous data model

Create the saemix.data object

```
saemix.data<-saemixData(name.data=res,header=TRUE,sep=" ",
  na=NA, name.group=c("id"),name.predictors=c("amount","time"),
  name.response=c("y1"), name.X="time")
```

Create the model

saemix models are contained in a R function with one blocks:

```
cat.model<-function(psi,id,xidep) {
level<-xidep[,1]
```

```
dose<-xidep[,2]
time<-xidep[,3]
th1 <- psi[id,1]
th2 <- psi[id,2]
delta0 <- psi[id,3]
lm0 <- th1+th2*time + delta0*dose
D <- exp(lm0)+1
P0 <- exp(lm0)/D
P1 <- 1/D

P.obs = (level==0)*P0+(level==1)*P1
return(P.obs) }

saemix.model<-saemixModel(model=cat.model,description="cat model",
  type="likelihood", psi0=matrix(c(2,1,2),ncol=3,byrow=TRUE,
  dimnames=list(NULL,c("th1","th2","th3"))),transform.par=c(0,1,1),
  covariance.model=matrix(c(1,0,0,0,1,0,0,0,1),ncol=3,byrow=TRUE),
  omega.init=matrix(c(2,0,0,0,1,0,0,0,1),ncol=3,byrow=TRUE),
  error.model="constant")
```

Run the SAEM algorithm

```
K1 = 500
K2 = 100

options<-list(seed=39546,map=F,fim=F,ll.is=F,
  nbiter.mcmc = c(2,2,2), nbiter.saemix = c(K1,K2),nbiter.sa=0,
  displayProgress=TRUE,save.graphs=FALSE,nbiter.burn =0)
saemix.fit<-saemix(saemix.model,saemix.data,options)
```

4.3 A repeated time-to-event data model

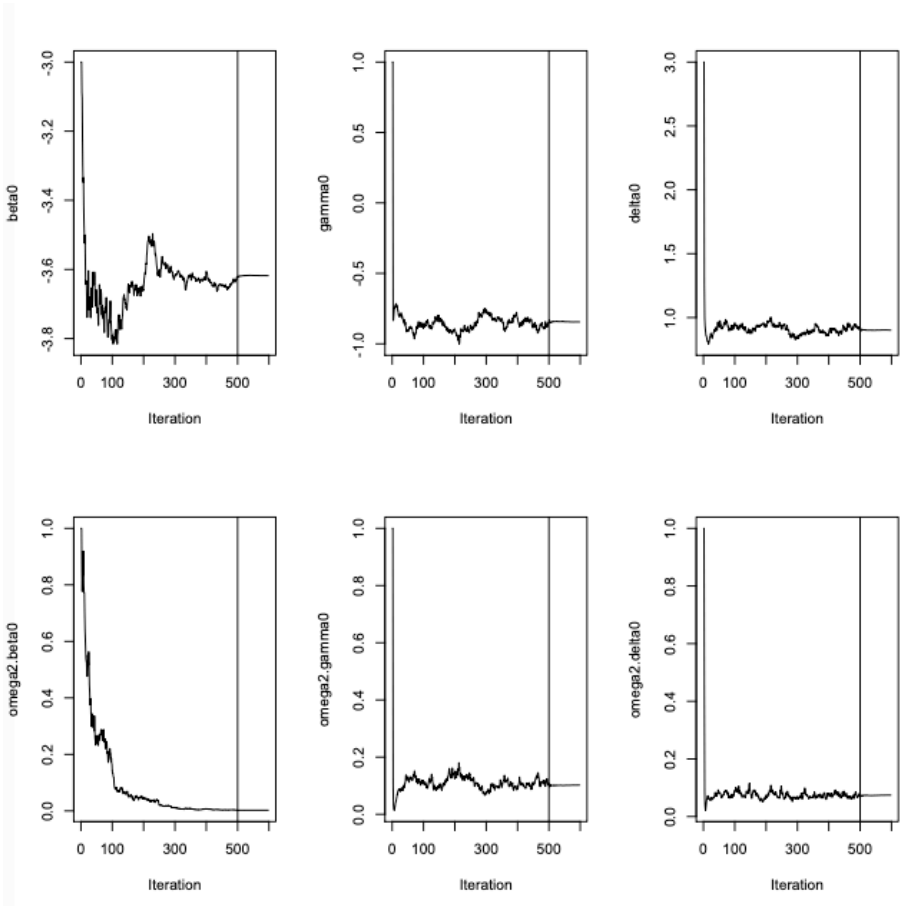
Read the Data

```
data(tte.saemix)
saemix.data<-saemixData(name.data=tte.saemix,header=TRUE,
  sep=" ",na=NA, name.group=c("id"),
  name.response=c("y"),name.predictors=c("time","y"),
  name.X=c("time"))
```

Create the Model

saemix models are contained in a R function with one blocks:

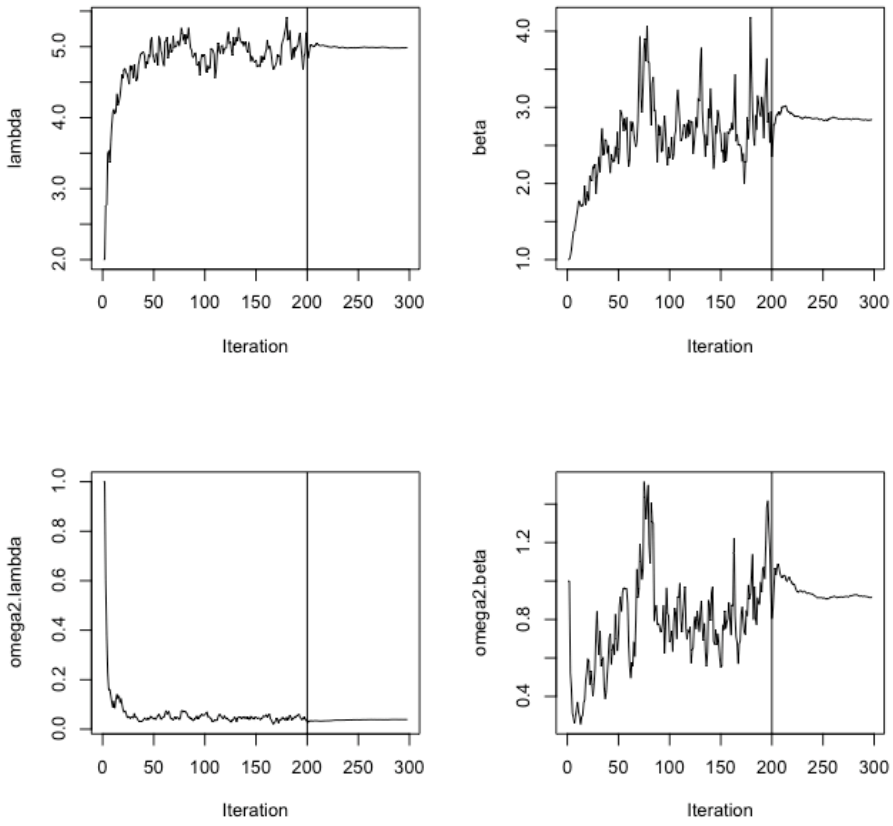
```
timetoevent.model<-function(psi,id,xidep) {
    T<-xidep[,1]
```


beta0
-3.0
-3.2
-3.4
-3.6
-3.8
0 100 300 500
Iteration
gamma0
1.0
0.5
0.0
-0.5
-1.0
0 100 300 500
Iteration
delta0
3.0
2.5
2.0
1.5
1.0
0 100 300 500
Iteration
omega2.beta0
1.0
0.8
0.6
0.4
0.2
0.0
0 100 300 500
Iteration
omega2.gamma0
1.0
0.8
0.6
0.4
0.2
0.0
0 100 300 500
Iteration
omega2.delta0
1.0
0.8
0.6
0.4
0.2
0.0
0 100 300 500
Iteration

```
    N <- nrow(psi)
    Nj <- length(T)
    censoringtime = 20
    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=timetoevent.model,description="time model",
  type="likelihood", psi0=matrix(c(2,1),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))
```

Run the SAEM algorithm

```
K1 = 200
K2 = 100

saemix.options<-list(map=F,fim=F,ll.is=F, nb.chains = 1,
  nbiter.saemix = c(K1,K2),displayProgress=TRUE,save.graphs=FALSE)
saemix.fit<-saemix(model,saemix.data,saemix.options)
```


lambda
2.0
3.0
4.0
5.0
0
50
100
150
200
250
300
Iteration
beta
1.0
2.0
3.0
4.0
0
50
100
150
200
250
300
Iteration
omega2.lambda
0.0
0.2
0.4
0.6
0.8
1.0
0
50
100
150
200
250
300
Iteration
omega2.beta
0.4
0.8
1.2
0
50
100
150
200
250
300
Iteration

Contacts

saemix is maintained by Emmanuelle Comets (emmanuelle.comets@inserm.fr) Inserm U738, Paris, France and CIC 0203, Rennes, France and Belhal Karimi (belhal.karimi@polytechnique.edu).

Please address any questions, bug notice or suggestions.

Bibliography

Comets, E., Lavenu, A., and Lavielle, M. (2017). Parameter estimation in nonlinear mixed effect models using saemix, an r implementation of the saem algorithm. *Journal of Statistical Software, Articles*, 80(3):1–41.

Delyon, B., Lavielle, M., and Moulines, E. (1999). Convergence of a stochastic approximation version of the EM algorithm. *Ann. Statist.*, 27(1):94–128.

Kuhn, E. and Lavielle, M. (2004). Coupling a stochastic approximation version of EM with an MCMC procedure. *ESAIM: Probability and Statistics*, 8:115–131.