SAEMIX, an R version of the SAEM algorithm

Emmanuelle Comets [1]★, Audrey Lavenu [2] and Marc Lavielle [3]

[1] *INSERM UMR738, Paris, France; Université Paris Diderot, Paris, France* [2] *University Rennes-I, Rennes, France; INSERM CIC 0203, Rennes, France* [3] *INRIA, Saclay, France*

Objective: to implement the SAEM algorithm [1] in the statistical software R [2].

Introduction

- Stochastic Approximation Expectation Maximization (SAEM) algorithm
  - stochastic approximation version of the EM algorithm
  - quick and efficient convergence to the maximum likelihood estimators (MLE) [3]
  - increasingly widespread use over the last few years
- Implementation in MONOLIX, NONMEM 7 and Matlab (statistical toolbox)
- R [2], a general statistical software
  - many packages for statistical analyses, including parameter estimation in linear and nonlinear mixed models (nlme, lmer, ...)
  - used by many modellers to handle data, prepare runs and evaluate results
  - flexible programming language including object-oriented concepts

Methods

Statistical models

Model for observation y_{ij}

$$y_{ij} = f(\psi_i, x_{ij}) + g(\psi_i, \gamma, x_{ij})\varepsilon_{ij}$$

- subject i ($i = 1,...N$), with n_i observations $\mathbf{y}_i = \{y_{i1},...,y_{in_i}\}$ at times t_{ij}, and covariates $\mathbf{z}_i$
- f: structural model (analytical expression, see example)
- $g = g(a,b,c)$: residual error model (one of: constant, proportional, combined, exponential) with parameters $\gamma = \{a,b,c\}$
- individual parameters ψ_i
  - modelled parametrically as a function $\psi_i = h(\phi_i, \mathbf{z}_i) = h(\mu(\mathbf{z}_i), \eta_i)$ of fixed effects μ and random effects η_i ($\eta \sim \mathcal{N}(0,\Omega)$)
  - in SAEMIX, h can be the identity function (normal distribution for ψ), the exponential function (log-normal distribution for ψ), or the logit or probit transformations

The SAEM algorithm

The SAEM algorithm computes the MLE of the unknown set of parameters $\theta = (\mu, \Omega, \gamma)$, by maximizing the likelihood of the observations $\ell(y;\theta)$. Given an initial estimate θ_0, at iteration k:

- *Simulation-step*: draw $\psi^{(k)}$ from the conditional distribution $p(\cdot|y;\theta_k)$
- *Stochastic approximation*: update the conditional expectation $Q_k(\theta)$ of the complete likelihood $p(y,\psi^{(k)};\theta)$ according to

$$Q_k(\theta) = Q_{k-1}(\theta) + \gamma_k(\log p(y,\psi^{(k)};\theta) - Q_{k-1}(\theta)) \quad (1)$$

where (γ_k) is a decreasing sequence of positive numbers with $\gamma_1 = 1$.

- *Maximization-step*: update θ_k according to

$$\theta_{k+1} = \underset{\theta}{\text{Arg max}}\, Q_k(\theta).$$

Using SAEMIX - a PK example

Input - Data

Dataset from 12 subjects given a single oral dose of theophylline used as an illustration:

- 11 blood samples over a period of 25 hours (data at t=0 was omitted from the dataset for all patients): nominal times 15 and 30 min, 1, 2, 4, 5, 7, 9, 12, 24 h
- one-compartment model with first-order absorption, parameterised as k_a, V, CL
- variability models: IIV modelled using an exponential model with diagonal variance-covariance matrix, residual variability modelled with a combined error model
- covariate model: weight on CL

Dataset available in the library (theo.saemix), used to create SAEMIX data object through the saemixData function:

```
data(theo.saemix)
data(theo.saemix)
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")
```

Input - Model

Define the model function and create SAEMIX model object through the saemixModel function:

```
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)
}
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"))),
covariate.model=matrix(c(0,0,1,0,0,0),ncol=3,byrow=TRUE))
```

Parameter estimation

```
saemix.options<-list((seed=632545,nb.chains=5,nbiter.saemix=c(300,150))
saemix.fit<-saemix(saemix.model,saemix.data,saemix.options)
```

Output - Fit results

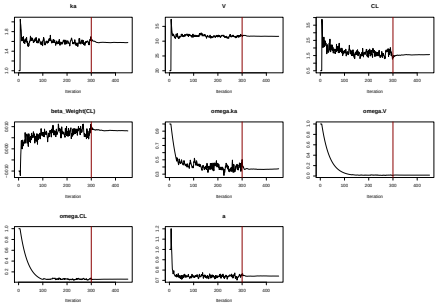

Figure 1: *Convergence plots.*

Parameter	Population estimate (SE%)	IIV Variance (SE%)
k_a (hr^{-1})	1.57 (19%)	0.39 (45%)
CL ($L.hr^{-1}$)	1.58 (64%)	0.07 (49%)
$\beta_{BW,CL}$ (-)	0.008 (110%)	-
V (L)	31.5 (4%)	0.02 (59%)
a ($mg.L^{-1}$)	0.74 (6%)	-

Table 1: *Pharmacokinetic parameters estimated by* SAEMIX *for the theophylline data.*

Output - Graphs

- Data
- Basic diagnostic plots
  - observations versus predictions
  - distribution of random effects
  - random effects and/or parameters versus covariates
  - individual fits

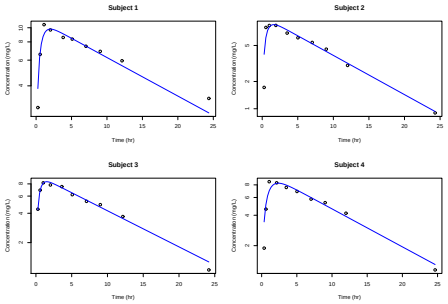

Figure 2: *Individual plots for 4 subjects, log-scale.*

- Simulation-based diagnostics
  - prediction discrepancies (pd), normalised prediction distribution errors (npde)
    - residuals versus time and predictions
    - histograms and QQ-plots of the distributions
  - VPC

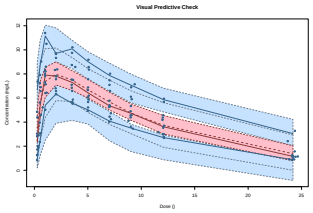

Figure 3: *VPC plot for the theophylline data.*

- A number of plots saved in output directory by default
  - additional options to customise plots (title, labels, colours, ...)
- example of code to obtain a VPC plot:

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

Simulation study

Methods

Same design as the simulation study performed by Plan et al. [4] to compare different estimation software:

- Models
  - structural model: sigmoid E_{max}
  - IIV: exponential model, no IIV on γ, correlation between E_{max} and EC_{50} ($\rho = 0.5$)
  - residual error model: additive

Parameter	Value	Parameter	Value
E_0 (-)	5	$\omega^2_{E_0}$	0.09
E_{max} (-)	30	$\omega^2_{E_{max}}$	0.49
EC_{50} (mg)	500	$\omega^2_{EC_{50}}$	0.49
γ (-)	2	$\text{cov}(E_0, E_{max})$	0.245
σ (-)	2		

Table 2: *Parameter values used in the simulation.*

- Design
  - 100 subjects simulated, 4 doses (0, 100, 300, 1000)
  - added a binary covariate (treatment) randomly assigned (50 subjects in each group)
- Evaluation
  - K=100 datasets simulated
  - bias and RMSE: compared to the parameters used for the simulation
  - type I error for a covariate model
    - simulation under H_0={no covariate effect}, estimation under H_0 or H_1={treatment effect on EC_{50}}
    - likelihood ratio test comparing the log-likelihood for both models
    - Wald test using the results from a model with the covariate
  - settings: 5 chains, number of iterations K_1=400 and K_2=500
- Comparison to the results obtained by nlme [5] (FOCE algorithm)

Results

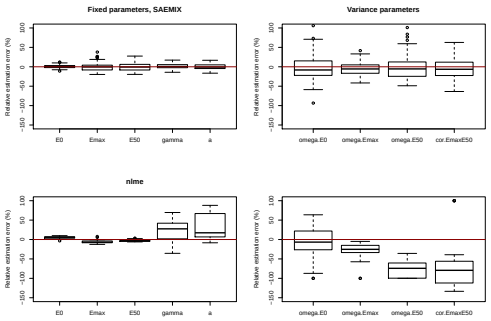

Figure 4: *Relative estimation errors for the fixed effects (left) and the variance components (right) for* SAEMIX *(top, K=100) and* nlme *(bottom, K=45).*

Performances similar to the results reported in [4]:

- good parameter estimates for SAEMIX
  - fixed effects: less than 1% bias and RMSE around 5-10%
  - variances: less than 5% bias and RMSE ranging from 20 to 40%
- estimation problems with nlme
  - low convergence rate, problems obtaining standard errors
  - performance of nlme remains poor when altering initial estimates

Type I error for covariate inclusion

Test	SAEMIX	nlme
LRT	0.07 [0.03-0.14]	0.21 [0.09-0.36] (K=39)
Wald	0.08 [0.03-0.15]	0.28 [0.15-0.44] (K=43)

Table 3: *Estimate ([CI]) of type I error rate under the null hypothesis. For* SAEMIX*, the LRT was performed using the estimate of LL obtained by importance sampling, and was computed over K=100 simulated datasets. For* nlme*, the number for each test is reported.*

Good performance for SAEMIX compared with nlme:

- adequate type I error for both tests
- for LRT, type I error based on linearised LL is p=0.22 (close to nlme)

Conclusion

The SAEMIX package for R implements the SAEM algorithm for parameter estimation in nonlinear mixed effect models. The algorithm has good performance .

It will be available on the CRAN shortly (installation as any other R package through the GUI or in command line).

It uses the S4 class system of R to provide a user-friendly input and output system.

REFERENCES

[1] E Kuhn and M Lavielle. Maximum likelihood estimation in nonlinear mixed effects models. *Computational Statistics and Data Analysis*, 49:1020–1038, 2005.

[2] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2006. ISBN 3-900051-07-0.

[3] B Delyon, M Lavielle, and E Moulines. Convergence of a stochastic approximation version of the EM algorithm. *Annals of Statistics*, 27:94–128, 1999.

[4] E Plan, A Maloney, F Mentré, MO Karlsson, and J Bertrand. Nonlinear mixed effects estimation algorithms: a performance comparison for continuous pharmacodynamic population models. *19th meeting of the Population Approach Group in Europe, Berlin, Germany*, 2010.

[5] Jose Pinheiro, Douglas Bates, Saikat DebRoy, Deepayan Sarkar, and the R Core team. *nlme: Linear and Nonlinear Mixed Effects Models*, 2009. R package version 3.1-96.

★ Presenting author

email: emmanuelle.comets@inserm.fr