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-1, 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 estimates
    (MLE) [3]
  – increasingly widespread use over the last few years
• Implementation in MONOLIX, NONMEM and Matlab (statistical software)
• R [2], a general statistical software
  – many packages for statistical analyses, including parameter estimation
    in linear and nonlinear mixed models (nlme, lme4, ...)
  – 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 \( y_i \):

\[
y_i = f(x_i, \theta) + g(W_i, \epsilon_i) \]

- subject \( i = 1, ..., N \), with \( n_i \) observations \( y_{i1}, ..., y_{in_i} \) at times \( t_{ik} \) and covariates \( x_i \)

• structural model (analytical expression, see example)

- residuals model: constant, proportional, combined, exponential with parameters \( \gamma = (a, b, c) \)
• individual parameters \( \psi_i \):
  - modelled parametrically as a function \( \psi_i = k\phi(\xi_i) = k(\mu(\xi_i), \eta_i) \) of fixed effects \( k \) and random effects \( \eta_i (\xi_i \sim N(0, \Omega)) \)
  - in SAEMIX, \( \chi \) can be the identity function (normal distribution for \( \chi \), the exponential function (log-normal distribution for \( \chi \), or the log or probit transformations)

The SAEM algorithm
The SAEM algorithm computes the MLE of the unknown set of parameters \( \theta = (\Omega, \Psi) \), by maximizing the likelihood of the observations \( L(\theta) \). Given an initial estimate \( \theta_0 \), at iteration \( j \):

- Simulation step: draw \( \psi_j^{(1)} \) from the conditional distribution \( p(\psi_j | y, \theta^{(j-1)}) \)
- Stochastic approximation: update the conditional expectation \( Q_j(\theta_j | y, \theta^{(j-1)}) \)

\[
Q_j(\theta | y) = Q_{j-1}(\theta | y) + \alpha_j \log p(y | \theta, \psi_j) - Q_{j-1}(\theta | y) \]

where \( \alpha_j \) is a decreasing sequence of positive numbers with \( \alpha_j = 1 \).

- Maximization step: update \( \theta_j \) according to

\[
\theta_{j+1} = \arg \max_{\theta_j} Q_j(\theta | y) \]

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

Simulation study

Methods

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

• Models
  – structural model: sigmoid function \( E_{\text{max}} \)
  – IV exponential model, no IV on \( \gamma \), correlation between \( E_{\text{max}} \) and \( \epsilon \)
  – residual error model: additive

Output - Fit results

• Parameter Population estimate IV Variance (SE%) [\%]
  – functional form: linear, log, ... [\%]
  – covariate effect

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

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
treatment effect on EC\(_{50}\) and \( E_{\text{max}} \)
  - random effects and/or parameters versus covariates
  - individual fits

Figure 4: Relative estimation errors for the fixed effects (left) and the variance components (right) for SAEMIX (top, \( K=100 \)) and \( \text{nlme} \) (bottom, \( R=43 \)).

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 \( \text{nlme} \)
  – low convergence rate: problems obtaining standard errors

Type I error for covariate inclusion

Test SAEMIX \( p \)-value

\[
\text{LRT: 0.10 (0.03-0.35) K=19}
\]

\[
\text{Wald: 0.08 (0.03-0.35) K=19}
\]

Table 3: Estimate (CEF) 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 \( \text{nlme} \), the numbers for each test is reported. Good performance for SAEMIX compared with \( \text{nlme} \):

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

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