nlmixr: an open-source package for pharmacometric modelling in R

Uppsala University Presentation

Rik Schoemaker, PhD
8 December 2016

The nlmixr development team:
Wenping Wang, Yuan Xiong, Justin Wilkins and Rik Schoemaker
nlmixr is an open-source R package

• Written by Wenping Wang and available on GitHub:
  • builds on RxODE, an R package for simulation of nonlinear mixed effect models using ODEs
  • combined with nlme, an R package for parameter estimation in nonlinear mixed effect models
  • but also gnlmm and SAEM estimation routines...
• nlmixr provides an efficient and versatile way to specify pharmacometric models (closed-form and ODEs) and dosing scenarios, with rapid execution due to compilation in C
• NONMEM® with first-order conditional estimation with interaction was used as a comparator to test nlmixr
Example syntax

```r
library(nlmixr)
datr<-read.csv("BOLUS_1CPT.csv", header=TRUE)
datr$EVID<-ifelse(datr$EVID==1, 101, datr$EVID)
specs<-list(fixed=lCL+lV~1, random=pdDiag(form=lCL+lV~1), start=c(lCL=1.6, lV=4.5))

#Closed-form:
fit<-nlme_lin_cmpt(datr, par_model=specs, ncmt=1, oral=FALSE, weight=varPower(fixed=c(1)))

#ODE:
ode <- "d/dt(centr) = -(CL/V)*centr;"
mypar <- function(lCL, lV )
  {CL = exp(lCL)
   V = exp(lV)}
fitODE<-nlme_ode(datr, model=ode, par_model=specs, par_trans=mypar, response="centr",
  response.scaler="V", weight=varPower(fixed=c(1)),
  control=nlmeControl(pnlsTol=.1))
```
Rich data sets

• 4 different dose levels (10, 30, 60 and 120 mg) of 30 subjects each as
  • single dose (over 72h)
  • multiple dose (4 daily doses)
  • single and multiple dose combined
  • and steady state dosing

• Range of test models:
  • 1- and 2-compartment disposition
  • with and without 1st order absorption
  • linear or Michaelis-Menten (MM) clearance

• A total of 42 test cases
  • all IIVs were set at 30%, residual error at 20%
  • overlapping PK parameters were the same for all models
Example full profiles (linear elimination)

Individual concentration profiles

<table>
<thead>
<tr>
<th>ID</th>
<th>Dose</th>
<th>Time (h)</th>
<th>Concentration (mg/L)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>60 mg</td>
<td>0.01</td>
<td>1000</td>
</tr>
<tr>
<td>2</td>
<td>10 mg</td>
<td>0.1</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>10 mg</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>120 mg</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>60 mg</td>
<td>100</td>
<td>0.1</td>
</tr>
<tr>
<td>6</td>
<td>30 mg</td>
<td>1000</td>
<td>0.01</td>
</tr>
<tr>
<td>7</td>
<td>120 mg</td>
<td>1000</td>
<td>0.01</td>
</tr>
<tr>
<td>8</td>
<td>30 mg</td>
<td>1000</td>
<td>0.01</td>
</tr>
<tr>
<td>9</td>
<td>120 mg</td>
<td>1000</td>
<td>0.01</td>
</tr>
<tr>
<td>10</td>
<td>30 mg</td>
<td>1000</td>
<td>0.01</td>
</tr>
</tbody>
</table>
Example full profiles (MM elimination)
Vc is available in all models: Theta estimates using NONMEM

Horizontal black line: value used for simulation
Grey line: nlmixr/nlme estimates using ODEs
Non MM models also implemented using closed-form solutions
Blue line: nlmixr/nlme estimates using closed-form solutions
SE of theta estimates for Vc are very comparable
Residual error is well-estimated
Horizontal black line: value used for simulation
Run times are perfectly acceptable, and often lower than NONMEM.
For Vc, Omega (IIV) estimates are also very comparable

Horizontal black line: value used for simulation
But if we examine $V_p$...
...or Ka...
...or Q...

often the IIVs are estimated close to zero
But what about sparse data?

- first-order absorption, one-compartment distribution, linear elimination model
- 4 doses, 150 subjects per dose
- 4 random time point samples in 24 hours after the 7\(^{th}\) dose
- 500 datasets
CL estimates using nlmixr (top) seem to demonstrate some bias compared to NONMEM (bottom)
And V estimates demonstrate even larger bias...
...but NONMEM and nlmixr estimates are highly correlated
This is also the case for Ka theta estimates… (right)
...but IIV estimates for Vc and especially for Ka clearly demonstrate a large fraction of runs with IIV=0 for nlmixr (91.1% vs. 2.2% for NONMEM)
Disappointing results?

- Findings are in line with earlier experience with nlme
- Bob Bauer claims nlme is somewhere between ITS and FOCE

- However, nlme in nlmixr provides a gateway into nonlinear mixed effect modelling for statisticians...

- With the machinery in place, the groundwork is laid for other/better estimation routines, like SAEM or FOCE-I...

- SAEM currently also available in nlmixr
Example nlmixr/SAEM syntax

```r
library(nlmixr)
datr<-read.csv("BOLUS_1CPT.csv", header=TRUE)
datr$EVID<-ifelse(datr$EVID==1,101,datr$EVID)
#temporary work-around for specifying covariates
datr$WT<-1

#Closed-form:
saem_fit <- gen_saem_user_fn(model=lincmt(ncmt=1, oral=FALSE))

#ODE:
od<"d/dt(centr) = -(CL/V)*centr;"
m3 = RxODE(ode, modName="m3")
PRED = function() centr / V
mympar <- function(lCL, lV )
{CL = exp(lCL)
 V = exp(lV)}
saem_fit <- gen_saem_user_fn(model=m3, PKpars=mypar, pred=PRED)

#run SAEM:
model = list(saem_mod=saem_fit, res.mod=2,covars="WT")
inits = list(theta=c(5,90),omega=c(0.1,0.1),bres=0.2)
cfg   = configsaem(model, datr, inits)
cfg$print = 50
fit = saem_fit(cfg)
```

nlmixr development team
IIVs for nlmixr/SAEM for Vp show none of the close to zero behaviour
IIVs for nlmixr/SAEM for Vp show none of the close to zero behaviour.
And some estimates seem better behaved...

Model:
- NONMEM
- nlmixr/SAEM ODE
- nlmixr/SAEM closed-form
…and could this also be the case for SEs for Vp…?
...and Vc...?
...and Vc...?
nlmixr/SAEM is slower than nlmixr/nlme but still workable
Very close correspondence for sparse sample theta estimates…
...and no close to zero IIVs for nlmixr/SAEM
...and no close to zero IIVs for nlmixr/SAEM and even better behaved than NONMEM
Is there an error in the algorithm in view of the systematic bias? Again, that pronounced shift to the left for nlmixr...
When samples are taken after the 1\textsuperscript{st} dose instead of the 7\textsuperscript{th}...
When samples are taken after the 1\textsuperscript{st} dose instead of the 7\textsuperscript{th}...
The bias is in the NONMEM estimates and nlmixr is spot on.
More good news?

- nlmixr is available on GitHub at https://github.com/nlmixrdevelopment/nlmixr
- nlmixr also has an adaptive Gaussian quadrature algorithm (like NONMEM’s Laplace and higher) allowing fancy models
- nlmixr also has single subject dynamic models e.g. for complex system simulation and estimation (mcmc algorithm)
- Elementary implementation of VPC and bootstrap functionality
What’s next?

• We need you!

• Improving computational efficiency of estimation algorithms (e.g. within-problem parallelisation)
• Implementation of FOCE-I
• Error-trapping
• Field-testing
• New features implementation
• Etc, etc...
Example nlmixr/gnlmm syntax:
PK-PD model with ODE of heavy-tail data: t-distribution

```
kin.m0 <- "
C2 = centr/V2;
C3 = peri/V3;
CONC = centr/V2*1000;
Stim = EMAX*(CONC^GAM)/(CONC^GAM+EC50^GAM);
d/dt(depot) = -KA*depot;
d/dt(centr) = KA*depot - CL*C2 - Q*C2 + Q*C3;
d/dt(peri) = Q*C2 - Q*C3;
d/dt(eff) = KIN*(1-Stim) - KOUT*eff;
"

sys1 = RxODE(kin.m0)
dt_ls <- function(x, df, mu, a, log=T) {
  if (log) {
    dt((x - mu)/a, df, log=T) - log(a)
  } else {
    1/a * dt((x - mu)/a, df)
  }
}
llik <- function() {
  pred = ifelse(eff>0.01, eff, 0.01)
  sdl = sqrt(sig2)*pred^.7
  #dnorm(DV, pred, sd=sdl, log=TRUE)
  dt_ls(DV, 4, pred, sd1, log=TRUE)
}
inits = list(THTA=c(-3, 0, 9, .7, -.4))
inits$OMGA = list(ETA[1]~.001, ETA[2]~1)
fit = gnlmm(llik, data, inits, pars, sys1,
  control=list(
    trace=TRUE,
    optim.inner = "Nelder-Mead",
    optim.outer = "nmsimplex",
    reltol.outer = 1.0e-3,
    mc.cores=4))
```
Example nlmixr/gnlmm syntax:
PK-PD model with ODE of bounded clinical endpoint: beta-distribution

```r
kin.m0 <- "
C2 = centr/V2;
C3 = peri/V3;
CONC = centr/V2*1000;
Stim= EMAX*(CONC^GAM)/(CONC^GAM+EC50^GAM);
d/dt(depot) = -KA*depot;
d/dt(centr) = KA*depot - CL*C2 - Q*C2 + Q*C3;
d/dt(peri) = Q*C2 - Q*C3;
d/dt(eff) = KIN*(1-Stim) - KOUT*eff;
"
.sys1 = RxODE(kin.m0)
llik <- function() {
  mn = ifelse(eff<.0001, .0001, eff2)
  odsp = odav/mn^pwod
  shp1 = mn*odsp
  shp2 = odsp - shp1
  dbeta(DV, shp1, shp2, log=TRUE)
}

inits = list(THTA=c(-3, 2, 7.5, .7, 2.1, -.4))
fit = gnlmm(llik, x, inits, pars, sys1,
            control=list(
               trace=TRUE,
               optim.outer = "nmsimplex",
               optim.inner = "Nelder-Mead",
               reltol.outer = 1.0e-2,
               mc.cores=4)
)
```
Example nlmixr/gnlmm syntax: PK-PD model with ODE of binary data with over-dispersion: beta-binomial distribution

od.bin1 <- "
C2 = centr/V2;
C3 = peri/V3;
CONC = centr/V2*1000;
\[\frac{d}{dt}(\text{depot}) = -KA*\text{depot};\]
\[\frac{d}{dt}(\text{centr}) = KA*\text{depot} - CL*C2 - Q*C2 + Q*C3;\]
\[\frac{d}{dt}(\text{peri}) = Q*C2 - Q*C3;\]
\[\frac{d}{dt}(\text{eff}) = kout*(1+emax*CONC^\text{gam}/(ec50^\text{gam}+CONC^\text{gam})) -kout*\text{eff};\]
sys5 = RxODE(od.bin1)

llik <- function() {
  lp = alpha+beta*eff
  pred = 1/(1+exp(-lp))
  if (do.betabinom)
    dbetabinom(DV, pred, 1, theta, log = TRUE)
  else dbinom(DV, 1, pred, log=TRUE)
}

do.betabinom = T
inits = list(
  THTA=c(2, -4, 0.3, -3, 0.2, -2, -3, 200),
  if(do.betabinom) 2 else NULL)
$OMGA = list(ETA[1]~.9, ETA[2]~.9)
fit1 = gnlmm(llik, mydat, inits, mypars, sys5,
control=list(
  trace=TRUE,
  mc.cores=4)
)