Case study 1: developing a two-compartment PK model through RsNLME

Introduction

- R is one of the most widely used softwares among pharmacometricians to perform data manipulation/visualization and statistical analysis.
- RsNLME provides a R interface to the Phoenix NLME engine to enable users to
  - Define PK/PD models via R objects (package **RsNlme**).
  - Use the "Initial Estimates" shiny app to visually determine a set of reasonable initial values for fixed effects (package **RsNlme**).
  - Perform estimation and simulation in a R environment with the capability of parallelizing the runs using Multicore, MPI and Grids (SGE/Torque/LSF) in-house or hosted on AWS (package **Certara.NLME8**).
  - Access the xpose graphics library for PK/PD models by creating compatible database from NLME results (package **Xpose.Nlme**).

Objectives

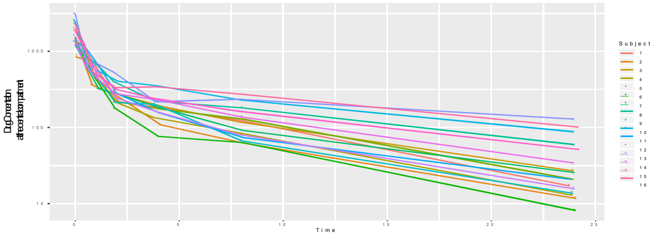
- Visually inspect the data and create the base model.
- Identify covariates through the stepwise covariate search.
- Bootstrapping analysis for the model selected by the covariate search procedure.

Note: R script and input dataset for this example can be found in C:\Program Files\R\R-n.n.n\library\RsNlme\

Construct the base model

Load the Input Dataset and Visually Inspect the Data

```
#loadtheinputdataset
dt_InputDataSet = fread("16subjects.csv")
```



Define the Base Model

```
#definethebasicPKmodel(atwo-compartmentmodelwithIVbolus)
model =pkmodel(numComp = 2, modelName = "TwCpt_IVBolus_FOCE-ELS")

#resetr e s i d u a lerrormodel(d e f a u l t:a d d i t i v emodelwithSD=1)
residualEffect(model,"C") = c(errorType= Multiplicative,SD= "0.16")
```

Map Model Variables to Input Dataset Columns

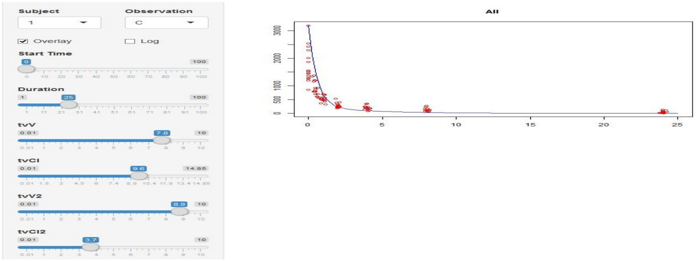
```
#initializemodelmappingandautomaticallymappingsomeofthemodel
 variablestothedatacolumns
initColMapping(model) = dt_InputDataSet

#manuallysetupthemappingfortherestofvariables
modelColumnMapping(model) = c(id = "Subject", CObs = "Conc", A1 = "Amount")
```

Use the Initial Estimates Shiny App, estimatesUI, to Visually Determine a Set of Reasonable Initial Values for Fixed Effects

```
#hostsetup:runlocallywithMPIenabled
host =NlmeParallelHost(sharedDirectory = Sys.getenv("NLME_ROOT_DIRECTORY")
                       , parallelMethod = NlmeParallelMethod("LOCAL_MPI")
                       , hostName = "MPI"
                       , numCores = 4)

#invoketheInitialEstimatesshinyapp
estimatesUI(model, unique(dt_InputDataSet$Subject), host)
```



....

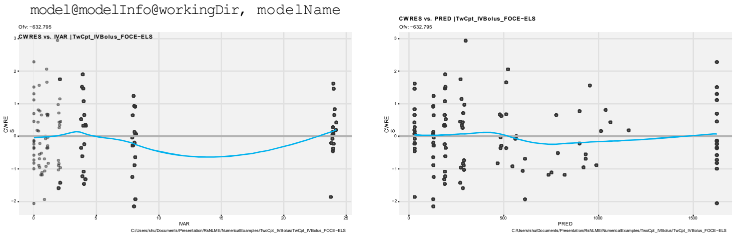
Fit the Base Model with Initial Estimates Picked from Shiny App

```
 #acceptinitialestimatespickedfromtheshinyapp  in
 itFixedEffects(model) =getInitialEstimates()

#enginesetup
engineParams =NlmeEngineExtraParams(PARAMS_METHOD = METHOD_FOCE_ELS,
                   PARAMS_NUM_ITERATIONS = 1000, PARAMS_SAND = "TRUE")

#fitthemodel
job =fitmodel(host, engineParams, model)

#importsresultsofanNLMErunintoxposedatabasetocreatecommonly
#useddiagnosticplots
 xp =xposeNlme(dir =                   = "TwCpt_IVBolus_FOCE-ELS")
 model@modelInfo@workingDir, modelName
```



Identify covariates through the covariate search

Add Covariates to the Base Model

```
#definecovariates,sex,weight,andage
sex =categoricalCovariate("sex",c(1,2), c("female","male"))
weight =NlmeCovariateParameter("weight", centerValue = "70",
                    continuousType = CovarNumber, direction = Forward)
age =NlmeCovariateParameter("age")

#automaticallyincorporatethecovariatesintothebasicmodel  c
ovarModel =addCovariates(covarModel, c(sex,weight,age),
                           c("V" = "weight,age", "Cl" = "sex,weight"))
```

Run the Stepwise Covariate Search

```
#setupforthestepwisecovariatesearch
stepwiseSearchSetup =NlmeStepwiseParams(0.01, 0.001, "-2LL")

#runstepwisecovariatesearch
job =stepwiseSearch(host, engineParams, c ovariateModel(covarModel), stepwis
               eSearchSetup, covarModel)
```

Load and View Results

```
#loadandviewthemodelselectedbythestepwisecovariatesearch
stepwiseLines = readLines("Stepwise.txt")
```

BootStrapping analysis for the selected model

Reset the Covariates to the List Suggested by the Covariate Search

```
#returnanewmodelwithallcovariateeffectscleared  selected
CovarModel =resetCovariateEffects(covarModel)

#enablethecovariatesselectedbythecovariatesearch  covariateEffect(s
electedCovarModel, "sex", "Cl") =                     EnableEffect
covariateEffect(selectedCovarModel, "age", "V") =   EnableEffect

#updatethePMLstatements
selectedCovarModel =generatePMLModel(selectedCovarModel)
```

Run the Bootstrap

```
#Copytheselectedmodelintoanewobject,andthencreateanewworking
#directoryandcopymodel,inputdataset,andcolumnmappingfilestoit
bootModel =copyModel(selectedCovarModel
      , modelName = "TwoCpt_IVBolus_SelectedCovarModel_Bootrapping"))

#Bootstrapsetup
bootSetup =NlmeBootstrapParams(numReplicates = 10, randomNumSeed = 1234
                               , stratifyColumns = "sex")

#Runthebootstrappingforthemodelselectedduringthecovariatesearch
job =bootstrap(host, engineParams, bootSetup, bootModel)
```

Load and View Results

```
#loadandviewtheestimationresultsforallboostrapruns
dt_out = fread("out.csv")
```

Conclusions

- RsNLME provides R commandline access to the Phoenix NLME engine allowing pharmacometricians with little or no knowledge of Phoenix NLME to format and visualize data, build and analyze models, and post-process results.
- RsNLME also provides greater flexibility for advanced Phoenix NLME users to work seamlessly with other R packages within the R environment.