NMsim - Seamless NONMEM Simulation Platform in R

Philip Delff[1]

[1] Vertex Pharmaceuticals Incorporated

Introduction

While NONMEM offers great flexibility for estimation of PK and PK/PD models, many users find the simulation features in NONMEM insufficient and turn to alternative software for simulation. This leads to additional work of model reimplementation, with risk of the simulation model deviating from the estimated model, due to bugs in the reimplementation. For a wide range of model types, the limitation is not in NONMEM's ability to perform such simulations, but rather in the lack of a simple user-interface to obtain the simulations. `NMsim` (Delff 2024b) provides such an interface as an R package, allowing the modeler to simulate models as soon as an estimate is available.

Objectives

The goal for NMsim is to automate the NONMEM simulation workflow and provide a simple, flexible, and powerful R interface. With this automation, post-processing of model estimates can to great extends be automated.

	NONMEM	Third-party	NMsim
Implementation	None	Potentially error-prone	None
Execution	Tedious	Easy	Easy
Depends on NONMEM	Yes	No	Yes
Runtime	Fair	Fast	Fair

Methods

`NMsim` does not simulate, translate or otherwise interpret a NONMEM model. Instead, it automates the NONMEM simulation workflow (including execution of NONMEM) and wraps it all into one R function. Provided with a path to a NONMEM control stream and a data.frame to simulate, `NMsim` will do the following:

- Save the simulation input data in a csv file for NONMEM
- Create a simulation input control stream based on `file.mod` (`$INPUT` and `$DATA` matching the saved simulation data set; `$SIMULATE` instead of `$ESTIMATION` and `$COVARIANCE`)
- Update and fix initial values based on estimate (from `file.ext`)
- Run NONMEM on the generated simulation control stream
- Collect output data tables, combine them, and merge with the simulation input data
- Return the collected data in R

`NMsim` can call NONMEM directly or via `PSN`. If `NMsim` is run on a system where NONMEM cannot be executed, `NMsim` can still prepare the simulation control stream and datafile.

`NMsim` is in itself a relatively small R package. It makes extensive use of functionality to handle NONMEM data and control streams provided by the R package `NMdata` (Delff 2024a).

Results

When providing a simulation data set, the default `NMsim()` behavior is to sample a new subject (ETA's).

```
library(NMsim) ## Used version 0.1.4
file.mod <- system.file("examples/nonmem/xgxr021.mod",
                        package="NMsim")
data.sim                                                   <-
        read.csv(system.file("examples/derived/dat_sim1.csv",
                             package="NMsim"))
simres <- NMsim(file.mod=file.mod,data=data.sim)
```

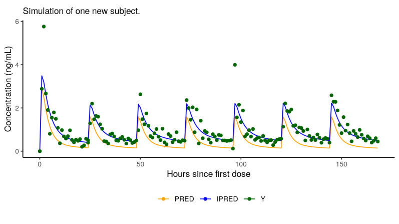

Figure 1: `PRED`, `IPRED`, and `Y` (if defined in control stream) are easily obtained with NMsim.

Notice that no information about the model is needed except for the control stream file path. The simulation is based on evaluation of `PRED`, `IPRED`, and optionally `Y`. Options exist for building more advanced simulation models. The models shown here are based on data available in the `xgxr` (Stein et al. 2021).

Generation of simulation data sets

The simulation input data set is a data.frame, and `NMsim()` returns a data.frame. The input data is a data.frame that

- Must contain at least the variables NONMEM will need to run the model (typically `ID`, `TIME`, `CMT`, `AMT`, etc. plus covariates)
- Can contain character variables (automatically carried to results)
- Column order does not matter

There are no requirements to how the data sets are created. NMsim provides convenient helper functions that can optionally be used. E.g., the data set used in these simulations can be created this way:

```
doses <- NMcreateDoses(TIME=c(0,24),AMT=c(300,150),
                       addl=list(ADDL=c(0,5),II=c(0,24)),CMT=1)
dat.sim <- addEVID2(doses,TIME=0:(24*7),CMT=2)
```

Typical subject simulation

- A typical subject is a subject with all ETAs = 0
- Covariates values are supplied using the simulation input data set
- `typical=TRUE`: replace all `$OMEGA` values with zeros

```
simres.typ <- NMsim(file.mod=file.mod,data=data.sim,
                    typical=TRUE)
```

Simulate multiple models

Multiple models can be simulated using the same data set in one function call by supplying more than one model in the `file.mod` argument. The models can be simulated on multiple data sets by submitting a list of data.frames in the `data` argument. NMsim will return one data.frame with all the results for easy post-processing.

```
file2.mod <- "models/xgxr114.mod"
simres.typ2 <- NMsim(file.mod=c("2 compartments"=file.mod,
                                "1 compartment"=file2.mod),
                     data=data.sim,
                     typical=TRUE)
## The "model" column is used to distinguish the two models
subset(simres.typ2,EVID==2) |>
    ggplot(aes(TIME,PRED,colour=model))+
    geom_line()
```

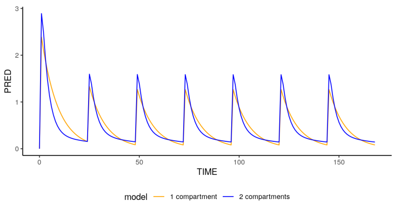

Figure 2: Simulation of multiple models and even multiple data sets is handled within one `NMsim()` call.

Emperical Bayes' Estimates (known ETAs)

- By default, automatically re-uses estimated individual ETAs
- ID values in simulation data must match the ID values in the estimation that you want to simulate
- Other ETA sources can be specified
- Does not simulate residual variability - see `addResVar()` if needed
- Remember: Covariates may be needed in data set to fully reproduce the subjects' parameters

```
## Example using same simulated dosing+sampling for all subjects
library(NMdata)
res <- NMscanData(file.mod,quiet=T)
ids <- unique(res$ID)[1:5]
data.sim.ind <- merge(subset(data.sim,select=-ID),
                      data.frame(ID=ids))
setorder(data.sim.ind,ID,TIME,EVID)
simres.ebe <- NMsim(file.mod,
                    data=data.sim.ind,
                    method.sim=NMsim_EBE,
                    table.vars=c("CL","V2","IPRED","PRED")
)
```

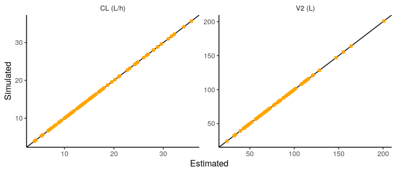

Figure 3: Individual parameters are confirmed to be identical in estimation results and simulation results

Prediction intervals

New subjects can be simulated in multiple ways with NMsim.

- If the input data set contains multiple subjects, these subjects will get separate random effects due to NONMEM `$SIMULATION`
- The `subproblems` argument translates to the `SUBPROBLEMS` NONMEM subroutine, replicating the simulation the specified number of times with new seeds
- The `simPopEtas()` function can generate a synthetic .phi file with a simulated population that can be reused in future `NMsim()` calls. This can be combined with simulation of covariates in R, allowing reuse of the same subjects across multiple simulations.

```
simres.subprob <- NMsim(file.mod=file.mod,
                        data=data.sim,
                        name.sim="Subproblems",
                        subproblems=1000)
## data.sim.nsubjs replicates data.sim for each subject,
## with sampled covariates
simPopEtas(file.mod=file.mod,N=1000,seed=1231,
           file.phi="simres/xgxr021_1000subjs.phi")
simres.datarep <- NMsim(file.mod=file.mod,
                        data=data.sim.nsubjs,
                        method.sim=NMsim_EBE,
                        file.phi="simres/xgxr021_1000subjs.phi",
                        name.sim="datarep")
simres.newsubjs <- rbind(as.data.table(simres.subprob),
                         as.data.table(simres.datarep),
                         fill=T)
```

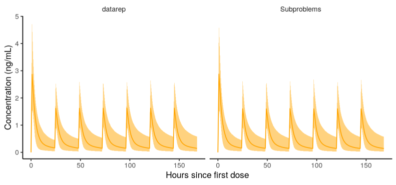

Figure 4: Prediction intervals. New subjects can be simulated in multiple ways with NMsim. A simulated population can be reused across simulations.

Configuration and Important `NMsim()` arguments

NMsim must be configured with the path to the NONMEM executable. This can be done for each `NMsim()` call using the `path.nonmem` argument, but more easily it can be configured globally the following way. Also including where NMsim will run NONMEM and store intermediate files (`dir.sims`) and where to store final results (`dir.res`).

```
library(NMdata)
NMdataConf(path.nonmem = "/opt/NONMEM/nm75/run/nmfe75")
## or on Windows, it could be
NMdataConf(path.nonmem = "c:/nm75g64/run/nmfe75.bat")
NMdataConf(dir.sims="simtmp", ## location of sim tmp files
           dir.res="simres")  ## location of sim results
```

`NMsim()` has many features which are explained and demonstrated in manuals and vignettes. A few often-used arguments are

- `table.vars`: Redefine the output table. This can dramatically speed up simulations. E.g., `table.vars=c("PRED","IPRED")`.
- `name.sim`: Assign a name to the simulation and the generated files. Keeps order and separates results files between simulations.
- `seed.R` and `seed.nm`: Define seed, either through R, or directly as the seed used in NONMEM simulation control stream.

See also

See the NMsim website for code, more publications, vignettes and news.

Related posters at ACoP 2024:

- Simulation of clinical trial predictions with model uncertainty using NMsim (T110)
- Building Automated Pharmacometrics Analysis Workflows in R with NMsim (T49)
- Simulate modified Nonmem models using NMsim (T19)
- A Model-Based Simulation Workflow Enables Automated and Accurate Generation of Clinical Pharmacology Summary Statistics (T103)

References

Delff, Philip. 2024a. *NMdata: Preparation, Checking and Post-Processing Data for PK/PD Modeling.* https://philipdelff.github.io/NMdata/.

———. 2024b. *NMsim: Seamless Nonmem Simulation Platform.* https://philipdelff.github.io/NMsim/.

Stein, Andrew, Alison Margolskee, Fariba Khanshan, and Konstantin Krismer. 2021. *Xgxr: Exploratory Graphics for Pharmacometrics.* https://opensource.nibr.com/xgx/.