

User Guide Overview

Contents

1	Repository Overview	2
1.1	data	2
1.1.1	derived	2
1.1.2	spec	2
1.2	deliv	3
1.2.1	figure	3
1.2.2	table	3
1.3	model	4
1.3.1	mrgsolve	4
1.3.2	nonmem	4
1.3.2.1	1000.ctl; 1000/	4
1.3.2.2	1000-FOCE.ctl; 1000-FOCE/	4
1.3.2.3	1000-7.4.ctl	4
1.3.2.4	2000.ctl; 2000/	4
1.3.2.5	2000pps.ctl; 2000pps/	4
1.3.2.6	2001.ctl; 2001/	4
1.3.2.7	2002.ctl; 2002/	4
1.3.2.8	2003.ctl; 2003/	4
1.3.2.9	2004.ctl; 2004/	4
1.3.2.10	2005.ctl; 2005/	5
1.4	script	5
1.4.1	applied-sims.R	5
1.4.2	demo-model-table.R	5
1.4.3	dfeval.R	5
1.4.4	diagnostic-templates/	5
1.4.5	functions-diagnostics-npde.R	5
1.4.6	functions-diagnostics-rhat-ess.R	5
1.4.7	functions-diagnostics.R	5
1.4.8	functions-model.R	5
1.4.9	functions-table.R	5
1.4.10	model-management.R	5
1.4.11	pediatricpps.R	5
1.4.12	pk-model-diagnostics-report.R	5
1.4.13	pksens.R	5
1.4.14	run-sims-npde.R	6
1.4.15	simout/	6
1.4.16	tags.yaml	6
2	Workflow overview	7
2.1	Model execution	7
2.2	Model diagnostics	8
2.3	Model parameter table	8
2.4	Sensitivity analysis	8
2.5	Applied simulations	8

1 Repository Overview

This repository consists of example code for running Bayesian estimation with NONMEM. In the sections below we provide a general overview for how to navigate these scripted tools.

Packages have been specified in `/renv` and should be accessible by first running in your RStudio Console window:

```
install.packages("renv")  
library(renv)  
renv::restore()
```

For further information on using renv, please see: <https://rstudio.github.io/renv/articles/renv.html#reproducibility>

1.1 data

The data directory holds two folders, `derived` and `spec`.

1.1.1 derived

The `derived` directory holds the example analysis datasets along with their associated specification documents.

1.1.2 spec

The `spec` directory holds the yaml files used to generate the specification pdfs located in the `derived` directory.

1.2 deliv

1.2.1 figure

The figure directory contains several directories with diagnostic and applied simulation outputs. Note, that the generating script is listed as footnote tag on the bottom left hand side of each figure. Additionally, any supplementary figures have been renamed to match their name in the manuscript submission.

1.2.2 table

The table directory contains example parameter tables for the models described in the tutorial. Note, that the generating script is listed as footnote tag on the bottom left hand side of each table. Additionally, any supplementary figures have been renamed to match their name in the tutorial.

1.3 model

The model directory contains folders for both the NONMEM and mrgsolve model files described in the tutorial.

1.3.1 mrgsolve

The names of the mrgsolve simulation model files correspond to the NONMEM models located in model/pk.

1.3.2 nonmem

1.3.2.1 1000.ct1; 1000/ This is a standard example control stream for a Bayesian population PK analysis. The folder named 1000 contains the outputs for the four chains associated with the Bayesian run along with an html diagnostic file. These four chains are generated as part of the `run_chains` R function which is described in further detail below in Section 2.1.

1.3.2.2 1000-FOCE.ct1; 1000-FOCE/ This is a standard example control stream for an FOCE population PK analysis. The folder named 1000-FOCE contains the outputs for the model run along with an html diagnostic file.

1.3.2.3 1000-7.4.ct1 This is a example control stream for writing out individual posterior samples of ETA values using FORTRAN write statements in NONMEM version 7.4.3.

1.3.2.4 2000.ct1; 2000/ This is the pediatric example control stream for a Bayesian population PK analysis. The folder named 2000 contains the outputs for the four chains associated with the Bayesian run along with an html diagnostic file. These four chains are generated as part of the `run_chains` which is described in further detail below in Section 2.1.

1.3.2.5 2000pps.ct1; 2000pps/ This is the pediatric example control stream for a prior predictive check. The application and assessment of the prior predictive check is shown `pediatricpps.R`

1.3.2.6 2001.ct1; 2001/ This is a sensitivity analysis for the pediatric example where Ka was fixed. The folder named 2001 contains the outputs for the four chains associated with the Bayesian run along with an html diagnostic file. These four chains are generated as part of the `run_chains` which is described in further detail below in Section 2.1.

1.3.2.7 2002.ct1; 2002/ This is a sensitivity analysis for the pediatric example where variance was inflated 10 fold. The folder named 2002 contains the outputs for the four chains associated with the Bayesian run along with an html diagnostic file. These four chains are generated as part of the `run_chains` which is described in further detail below in Section 2.1.

1.3.2.8 2003.ct1; 2003/ This is a sensitivity analysis for the pediatric example where variance was inflated 100 fold. The folder named 2003 contains the outputs for the four chains associated with the Bayesian run along with an html diagnostic file. These four chains are generated as part of the `run_chains` which is described in further detail below in Section 2.1.

1.3.2.9 2004.ct1; 2004/ This is a sensitivity analysis for the pediatric example where the location parameter was increased 50%. The folder named 2004 contains the outputs for the four chains associated with the Bayesian run along with an html diagnostic file. These four chains are generated as part of the `run_chains` which is described in further detail below in Section 2.1.

1.3.2.10 2005.ctf; 2005/ This is a sensitivity analysis for the pediatric example where the location parameter was decreased 50%. The folder named 2005 contains the outputs for the four chains associated with the Bayesian run along with an html diagnostic file. These four chains are generated as part of the `run_chains` which is described in further detail below in Section 2.1.

1.4 script

1.4.1 applied-sims.R

This is an R script that performs the applied simulations for pediatric example.

1.4.2 demo-model-table.R

This is an R script for making the model parameter tables seen in the `table` directory.

1.4.3 dfeval.R

This is an R script that visualizes the properties of inverse Wishart distributions.

1.4.4 diagnostic-templates/

This folder contains diagnostic templates examples for generating the Bayesian-specific diagnostics.

1.4.5 functions-diagnostics-npde.R

This is an R script containing helper functions for post-hoc npde generation.

1.4.6 functions-diagnostics-rhat-ess.R

This is an R script containing helper functions for Bayesian-specific diagnostics.

1.4.7 functions-diagnostics.R

This is an R script containing helper functions for general diagnostics.

1.4.8 functions-model.R

This is an R script containing helper functions for Bayesian model management.

1.4.9 functions-table.R

This is an R script containing helper functions for table development with Bayesian-specific content.

1.4.10 model-management.R

This is an R script for model management (model creation and submission).

1.4.11 pediatricpps.R

This is an R script for performing prior predictive simulations for the pediatric example.

1.4.12 pk-model-diagnostics-report.R

This is an R script for generating model diagnostics.

1.4.13 pksens.R

This is an R script for summarizing the results of the sensitivity analysis for the pediatric example.

1.4.14 run-sims-npde.R

This is another R script containing helper functions for post-hoc npde generation.

1.4.15 simout/

This directory holds the simulation output for post-hoc diagnostics.

1.4.16 tags.yaml

This yaml file contains model tags for use with the bbr package.

2 Workflow overview

The process of conducting a Bayesian estimation using these tools is as follows:

2.1 Model execution

Start by looking into the `model-management.R` script. This script contains examples of how to submit NONMEM models for Bayesian estimation using the open source package `bbr`. The setup for this package is provided at the top of the script with links to pages for additional information on the package.

The general premise is that you have an initial control stream constructed as shown in `model/pk/1000.ct1` whereby the structural model is specified along with the desired prior information. This will be referred to as a chain stub file. At the end of the control stream you will note that there are two EST lines, one with `METHOD = CHAIN` and one with `METHOD = BAYES` with the second line commented out. The intention here is to first run this script and use `METHOD=CHAIN` to generate starting values for the four chains (`NSAMPLE = 4`) that will be run. Please refer to the tutorial for information on how to set up the options for `METHOD=CHAIN`. Note, the second EST line will be the estimation method applied to all subsequent chains run for this model and can be set up with whatever options the user may be interested in applying.

In `model-management.R`, on line 63, a model object is created using `bbr`, read in on line 77, and then submitted on line 81. The outcome of this submission for `1000.ct1` is the generation of a file named `1000.chn` (found in `1000/`) that contains the four starting values for the subsequent chains to be run (see line 88 for a assessment of this file). Next, using the “`run_chains`” function, four chains will be written out and submitted.

This function (`run_chains`) takes as arguments the model directory and the name or number of the model you are looking to submit for Bayesian estimation. There are also other options that are passed to `bbr`, refer to the the arguments list and `bbr` documentation for further information. What the “`run_chains`” function does is take the chain stub control stream, in this case `1000.ct1`, and uses it as a template to write out four additional control streams (appended with a numeral to denote the chain number) where each control stream pulls from the chain file (`1000.chn`) to use the values as starting points for the sampling.

Looking into the `1000/` folder directory, we can open up one of the derived control streams to examine the changes. In `1000_1.ct1`, we can see that the overall structure of the model along with the priors is the same as specified in the original stub file. However, changes can be seen on both EST lines. The line with `METHOD=CHAIN` has been updated so that `FILE = ../1000.chn`, `NSAMPLE=0`, and `ISAMPLE=1`. What these changes correspond to are that this model should look into the file named `1000.chn`, one directory above the directory where the model is going to be run, and that it should use the first sample record from this file (`ISAMPLE=1`) as the initial estimates for the sampling chain. In the other control streams (`1000_2.ct1`, `1000_3.ct1`, etc.), you can see that the `ISAMPLE` value is updated to 2 and 3 accordingly, so that each chain will be run with different initial values corresponding to the sampled values in the `.chn` file. The second change in these derived files, is that the EST line with `METHOD=BAYES`, is now uncommented, so that when submitted, the runs will now execute and proceed with their sampling. Note that a user can specify any estimation options they want, they just need to specify them in the chain stub file, this is just an example. After being submitted, the output for each chain is contained in a sub directory with a name following the model number (`1000_1/`, `1000_2/`, etc.)

The pediatric example and the associated sensitivity analyses are shown further below in the `model-management.R` script.

An example prior predictive simulation is conducted for the pediatric example. See `2000pps.ct1` for an example on how to generate parameter vectors from the set prior distributions using NONMEM. The `$SIML` line is called with `TRUE = PRIOR` to allow for simulations from the prior distribution. In this case, we are only interested in the simulation of the parameters from their priors as we will subsequently pass these along to `mrgsolve` for simulation. Note that the estimation line is set to draw only 1 sample each for the warm-up and post-warm-up period. This is solely to get the LKJ prior to be applied correctly.

2.2 Model diagnostics

To generate diagnostics for a given model run, we use the `pk-model-diagnostics-report.R` script. This script relies upon several helper functions contained in the sourced scripts of `run-sims-npde.R`, `functions-diagnostics-npde.R`, and `functions-model.R`. The general premise is that working within `pk-model-diagnostics-report.R` you are able to specify which models you want to generate diagnostics for by passing in model specific information and rendering a template diagnostic file.

For example, for the demo model `1000.ct1`, the generation of diagnostics is shown starting on line 63. On line 69, the `run_sims_npde` function is run to generate simulations using the posterior distribution and `mrgsolve` for the purposes of generating diagnostics described in section titled “Model Assessment and Selection.” The required arguments for the `run_sims_npde` function are the `bbr` model object, the path to the corresponding `mrgsolve` model file, the path for the simulated outputs, and the number of posterior samples to be used for the simulations.

After running the simulations, model diagnostics are generated by rendering a model diagnostic template `.Rmd` file. This `.Rmd` template, `template-bayes-report.Rmd`, is found in the `diagnostic-templates` folder. The premise of this template is that model specific inputs are provide along with the render argument, and then model specific diagnostic files are generated. Note, the templated document can be modified to generate any desired outputs with respect to covariate subsets etc. Individual diagnostic files are generated and provided in a directory under `deliv/figure/report` within a directory named for the corresponding model names. Additionally, an `html` file is generated and output to the corresponding model directory under `model/pk/` for each model name. In the case of `1000.ct1`, this corresponding file can be found under `model/pk/1000/diagnostic-plots-1000.html`. These files can be opened in the user’s web browser by calling `browseURL` (see line 102).

2.3 Model parameter table

Example model parameter tables are generated using `demo-model-table.R`. Examples are provided for both the demo example (`1000.ct1`) and the pediatric example (`2000.ct1`). This script generates parameter tables using `pmtables` package with Bayesian-specific statistics (ESS and Rhat). The output is generated as a PDF and can be found in `deliv/table/pk` under a folder corresponding to the model `number/name`.

2.4 Sensitivity analysis

The assessment and evaluation of the sensitivity analyses are performed in the `pkstats.R` script. The premise of this script is that original estimated model and the subsequent sensitivity analyses are read in and contrasted. First, the posterior samples from the models are read in and formatted to identify which model they correspond to. Next, plots showing the impact of the prior information for `KA` on `CL/F` and `V2/F` are generated. Then using each model, the individual posterior samples are read in and formatted to make them accessible to the `loo` package. Lastly, for the original model and each sensitivity analysis run, the `loo` function of the `loo` package is called to generate the `ELPD` of each model. These values are then compared using the `loo_compare` function from the `loo` package and a table of these comparisons and is produced.

2.5 Applied simulations

For the applied simulation scenario described in the tutorial, population simulations are conducted using 1000 random samples from the posterior distribution. On line 42, the posterior distribution is read in and then 1000 random samples are selected from them. On line 64, a standard population of 1000 pediatric patients is generated by resampling from the analysis data. Next, four dose levels are evaluated using the resampled posterior samples to incorporate uncertainty. For each replicate, the percentage of patients with trough values below some hypothetical value is calculated and the resulting outputs are saved out. The resulting population percentage values are summarized for each dose level and a figure is generated to visual the results.