bbr.bayes: An Open-Source Tool to Facilitate an Efficient, Reproducible Bayesian Workflow Using NONMEM

Tim Waterhouse[1], Kyle Meyer[1], Seth Green[1], Curtis Johnston[1], Bill Gillespie[1], Kyle Baron[1], Jonathan French[1,2], Katherine Kay[1], Tim Davis[1], Brian Davis[1], Matthew Riggs[1]

[1]*Metrum Research Group, Tariffville, CT, USA*, [2]*Johnson & Johnson Innovative Medicine, USA*

Abstract

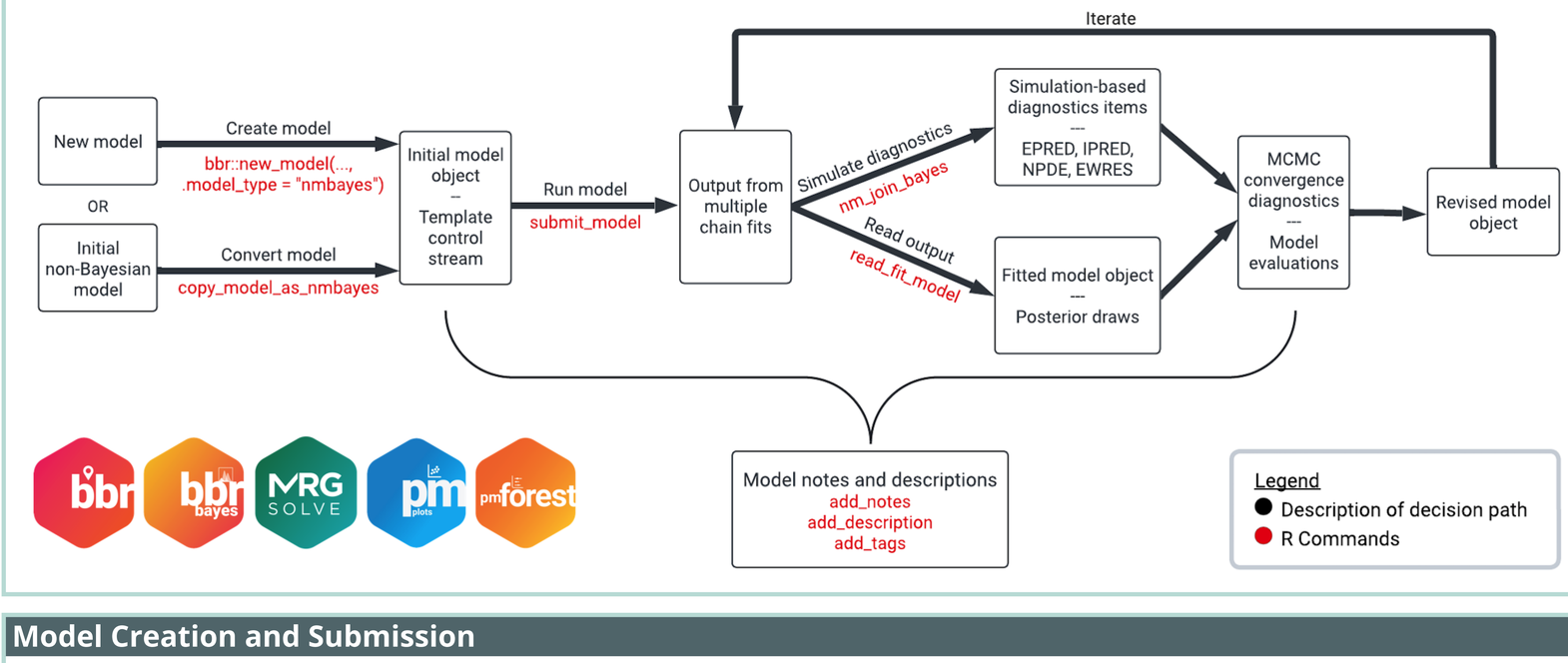
Introduction/Objectives: With the introduction of Monte Carlo Bayesian methods in NONMEM® 7, Bayesian approaches to modeling have become more accessible to those in the pharmacometrics community who are already familiar with NONMEM®. In addition to accessibility, NONMEM® provides a higher degree of optimization than other Bayesian tools for the types of datasets and model structures often encountered in pharmacometrics settings. These reasons make NONMEM® an attractive option for the pharmacometrician wishing to perform Bayesian analyses. However, constructing and managing control streams for multiple Markov chain Monte Carlo (MCMC) chains and then appropriately processing the full posterior distribution from model output for diagnosing, summarizing, and applying model fits [1] can be challenging. The objective of this work was to support good practice approaches and provide the pharmacometrics community with the R package **bbr.bayes**, which works in concert with other open-source tools to enable an efficient and reproducible Bayesian workflow with NONMEM® along with a set of illustrative examples to guide its appropriate use.

Methods: Metrum Research Group (MetrumRG) previously developed **bbr** [2], an R package for managing modeling and simulation projects, and has extended this with a new package, **bbr.bayes** [3], for accommodating Bayesian analyses. The **bbr.bayes** package facilitates traceable and reproducible Bayesian workflows in NONMEM® (and Stan [4]) by automating creation and submission of multiple MCMC chains as well as integrating harmoniously with (i) existing MeRGE [5] packages for data handling and reporting, (ii) mrgsolve [6] for generating simulation-based diagnostics, and (iii) packages from the Bayesian modeling community such as posterior [7] and bayesplot [8] for efficient handling of outputs like posterior draws and generating MCMC diagnostic plots. We have assembled example code and accompanying documentation for typical tasks in a NONMEM® Bayesian workflow to illustrate the functionality of **bbr** and **bbr.bayes** working in concert with these other packages. While these tasks overlap with many of those considered for a typical analysis using maximum likelihood estimation [9], these Bayesian-specific examples focus on the use of the full Bayesian posterior in downstream activities such as construction of model diagnostics, MCMC diagnostics, parameter tables, and forest plots.

Results: The **bbr.bayes** package reduces much of the friction associated with a Bayesian pharmacometrics analysis in NONMEM® and promotes good practice applications. In addition to managing the multiple MCMC chains required for such an analysis in a traceable and reproducible manner, the package provides functionality for generating simulation-based diagnostic items using the full posterior including individual and population predictions, normalized prediction distribution errors, and expected weighted residuals. A complete, reproducible example of a NONMEM® Bayes workflow is hosted in a publicly-available, version-controlled repository on GitHub encompassing multiple states and stages of a modeling and simulation project. Similarly, source code for **bbr.bayes** is hosted in a public GitHub repository. In addition to the scripted example, vignettes and user guides provide step-by-step directions detailing how **bbr.bayes** and other R packages facilitate key parts of the modeling and simulation analysis workflow by utilizing the full Bayesian posterior [10].

Conclusions: MetrumRG developed the open-source R package **bbr.bayes** to support traceable, reproducible, and scalable Bayesian pharmacometrics analyses in NONMEM®. Examples of how to use these tools in conjunction with best practice recommendations are provided to the pharmacometrics community.

Workflow



Model Creation and Submission

A new model object may be created for an existing Bayesian NONMEM® template control stream:

```
mod100 <- new_model(file.path(MODEL_DIR, "100"),
  .model_type = "nmbayes")
```

Alternatively, **bbr.bayes** can generate a new template control stream from a previous non-Bayesian model (e.g., FOCE):

```
mod99 <- read_model(file.path(MODEL_DIR, 99))
mod100 <- copy_model_as_nmbayes(mod99, file.path(MODEL_DIR, 100))
```

This adds some placeholder code to define the Bayesian model. The user then adapts this as necessary and adds appropriate priors.

```
$EST METHOD=CHAIN FILE=model_id.chn NSAMPLE=4 ISAMPLE=0 SEED=1
 CTYPE=0 IACCEPT=0.3 DF=10 DFS=0
$EST METHOD=NUTS SEED=1 NBURN=250 NITER=NNNN
 AUTO=2 CTYPE=0 OLKJDF=2 OVARF=1
 NUTS_DELTA=0.95 PRINT=10 MSFO=model_id.msf RANMETHOD=P PARAFPRINT=10000
 BAYES_PHI_STORE=1
```

Now we have a single template control stream associated with the model object `mod100`. **bbr.bayes** makes use of NONMEM®'s `METHOD=CHAIN` to generate initial estimates for separate runs (chains) of the same model.

bbr.bayes requires that the control stream includes a line with

```
$EST METHOD=CHAIN FILE=xyz.chn NSAMPLE=4 ISAMPLE=0 ...
```

where `xyz.chn` is the file with initial estimates for model number xyz, and `NSAMPLE` is the number of chains. `ISAMPLE` is set accordingly when running each chain. But this initial value of 0 is set to generate the initial estimates rather than read them in.

Submit a model with a single function call:

```
submit_model(mod100, .bbi_args = list(threads = 2))
```

This generates and runs several control streams:

1. A control stream to generate the sets of initial estimates.
2. A control stream for each chain.

The individual chain runs can be monitored using some **bbr** helpers, e.g.:

```
tail_output(mod100)
```

```
— Chain 1 ————————————————————————————————————————
License Registered to: Metrum Research Group
Expiration Date:    14 JUL 2024
Current Date:        2 JUN 2024
...
 iteration         -110 MCMCOBJ=    28437.707064556558
```

MCMC Diagnostics

With **bbr.bayes** we can easily read the output from all NONMEM® chains and store the result in a single **draws_array** object from the **posterior** [7] package.

```
draws <- read_fit_model(file.path(MODEL_DIR, 100))
```

For illustration, we'll restrict this example to only the `THETA` parameters.

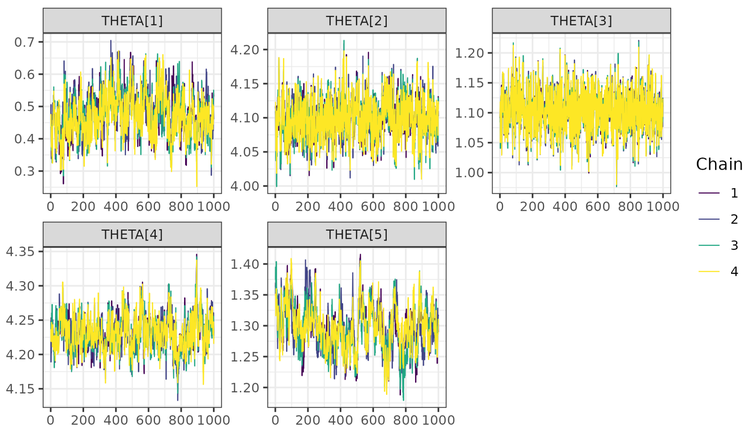
```
draws_theta <- posterior::subset_draws(draws, variable = "THETA")
```

Diagnostics of the MCMC draws can be generated with functions from the **posterior** and **bayesplot** [8] packages.

```
posterior::summarize_draws(draws_theta)
```

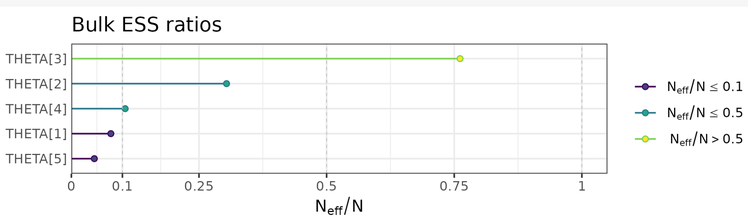
```
# A tibble: 5 × 10
  variable  mean median     sd    mad    q5   q95  rhat ess_bulk ess_tail
  <chr>    <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl> <dbl>    <dbl>    <dbl>
1 THETA[1] 0.475  0.475 0.0646 0.0625 0.369 0.581 1.00      309.     841.
2 THETA[2] 4.10   4.10  0.0290 0.0285 4.05  4.15  1.00     1216.    2330.
3 THETA[3] 1.10   1.10  0.0337 0.0319 1.05  1.16  0.999    3046.    3457.
4 THETA[4] 4.23   4.23  0.0249 0.0246 4.19  4.27  1.00      422.     712.
5 THETA[5] 1.30   1.30  0.0368 0.0375 1.24  1.36  1.00      180.     459.
```

```
bayesplot::mcmc_trace(draws_theta)
```



With a little more wrangling of the output, we can generate additional visualizations of MCMC diagnostics.

```
draws_sum <- posterior::summarize_draws(draws_theta)
ess_bulk <- draws_sum$ess_bulk
names(ess_bulk) <- draws_sum$variable
ess_bulk_ratios <- ess_bulk / (niterations(draws) * nchains(draws))
mcmc_neff(ess_bulk_ratios) + yaxis_text() + labels(title = "Bulk ESS ratios")
```



Model Diagnostics

To diagnose model goodness-of-fit, we use the same tools as in a typical non-Bayesian pharmacometric analysis, although some items need to be generated using the full Bayesian posterior:

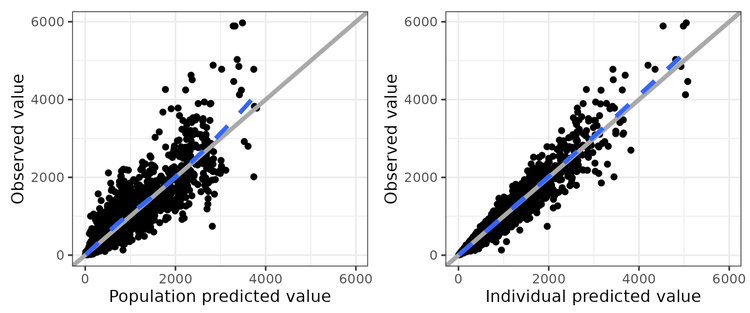
- Population predictions (EPRED)
- Individual predictions (IPRED)
- Normalized prediction distribution errors (NPDE)
- Expected weighted residuals (EWRES)

This is accomplished using the `nm_join_bayes()` function from **bbr.bayes**. This uses the **npde** package [11] under the hood and makes use of an **mrgsolve** [6] simulation model.

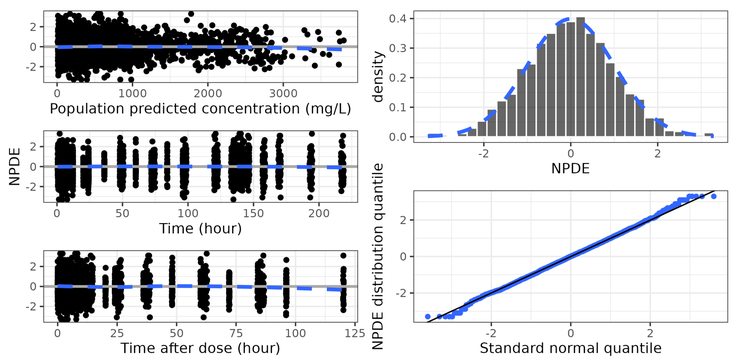
```
mod_ms <- mrgsolve::mread(here("script/model/100.mod"))
data <- nm_join_bayes(mod100, mod_ms, n_post = 1000)
```

We then use our standard tools such as **pmplots** [12] to generate diagnostic plots.

```
pmplots::dv_pred(data, x = "EPRED//Population predicted xname")
pmplots::dv_ipred(data)
```



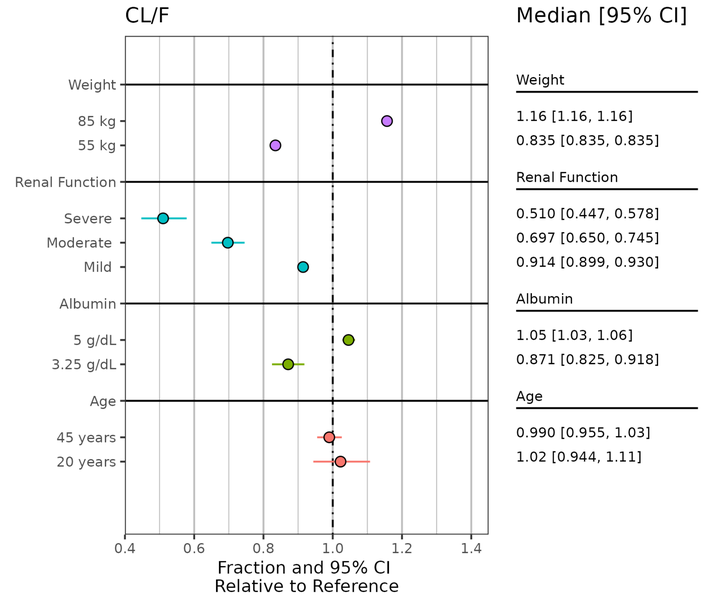

```
npde_pred(data, x = xPRED, y = "NPDE // ")
npde_time(data, x = xTIME)
npde_tad(data, x = xTAD, y = "NPDE // ")
npde_hist_q(data)
```



Posterior Applications

Simulations from a model involving parameter uncertainty involves simply sampling from the posterior draws supplied by the `read_fit_model()` function in **bbr.bayes**.

For example, model predictions generated by these posterior draws may be summarized and plotted using the **pmforest** [13] package.



External Packages

- The **npde R package** [11] is used to calculate the NPDE values from the full Bayesian posterior.

From the Stan ecosystem:

- The **posterior R package** [7] provides tools for working with output from Bayesian models.
- The **tidybayes R package** [14] integrates Bayesian modeling with the tidyverse and ggplot.
- The **bayesplot R package** [8] plots Bayesian models and diagnostics with ggplot.

Visit our Expo website

You'll find:

- Demonstration of various aspects of a Bayesian modeling workflow with NONMEM® using **bbr** and **bbr.bayes**.
- Access to example code in a Github repository.
- Information and vignettes on MetrumRG's suite of tools.

References

[1] Johnston, C.K., Waterhouse, T., Wiens, M., Mondick, J., French, J. and Gillespie, W.R. Bayesian estimation in NONMEM. *CPT Pharmacometrics Syst Pharmacol* **13** (2024):192–207.

[2] bbr package documentation. `https://metrumresearchgroup.github.io/bbr/`.

[3] bbr.bayes package documentation. `https://metrumresearchgroup.github.io/bbr.bayes/`.

[4] MeRGE Expo 2: Stan with bbr. `https://merge.metrumrg.com/expo/expo2-stan/`.

[5] MeRGE Expo. `https://www.metrumrg.com/merge-expo/`.

[6] mrgsolve documentation. `https://mrgsolve.org/`.

[7] posterior package documentation. `https://mc-stan.org/posterior/`.

[8] bayesplot package documentation. `https://mc-stan.org/bayesplot/`.

[9] Kay, K., Baron, K., Green, S., Callisto, S., Johnston, C., Barrett, K., Pastoor, D., Rogers, J., Ruiz-Garcia, A., Waterhouse, T., Wiens, M. and Riggs, M. A Suite of Open-Source Tools to Guide Efficient Pharmacometric Analyses. American Conference on Pharmacometrics (ACoP13) (2022).

[10] MeRGE Expo 3: NONMEM Bayesian estimation with bbr.bayes. `https://merge.metrumrg.com/expo/expo3-nonmem-bayes/`.

[11] Comets, E., Brendel, K. and Mentre, F. Computing normalised prediction distribution errors to evaluate nonlinear mixed-effect models: the npde add-on package for R. *Computer Methods and Programs in Biomedicine* **90** (2008):154–66.

[12] pmplots package documentation. `https://metrumresearchgroup.github.io/pmplots/`.

[13] pmforest package documentation. `https://metrumresearchgroup.github.io/pmforest/`.

[14] tidybayes package documentation. `https://mjskay.github.io/tidybayes/`.