A Suite of Open-Source Tools to Guide Efficient Pharmacometric Analyses

Katherine Kay, Ph.D.*[1], Kyle Baron, Pharm.D. Ph.D.*[1], Seth Green M.S.[1], Samuel Callisto, Ph.D.[1], Curtis Johnston, Pharm.D.[1], Kyle Barrett, M.S.[1], Devin Pastoor, Ph.D.[1,2], Jim Rogers, Ph.D.[1], Ana Ruiz-Garcia, Pharm.D. Ph.D.[1,3], Tim Waterhouse, Ph.D.[1], Matthew Wiens, M.A.[1], Matthew Riggs, Ph.D.[1]

*These authors contributed equally to this work
[1]*Metrum Research Group, Tariffville, CT, USA*, [2]*Posit (formerly RStudio), Boston, MA, USA (current)*, [3]*Gilead, City, ST, USA (current)*

METRUM RESEARCH GROUP

Abstract

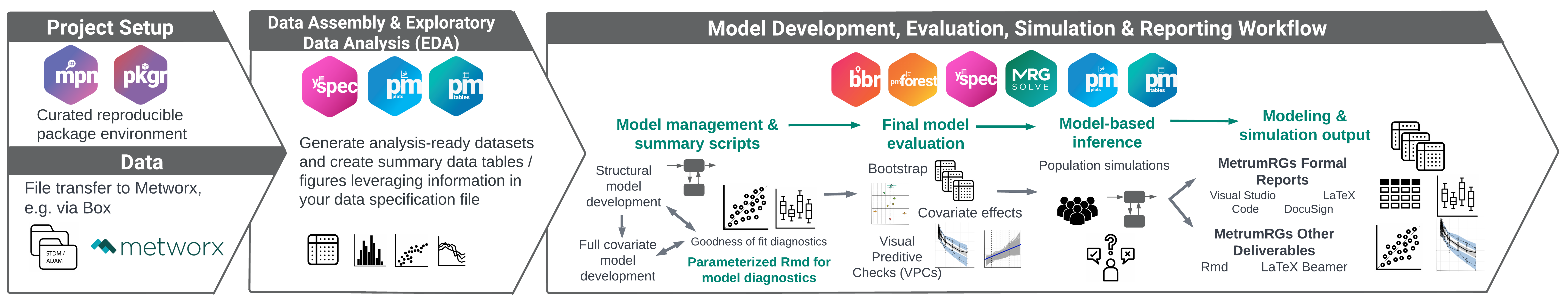
BACKGROUND Quantitative scientists are tasked with building analytical models to support decisions about complex systems. Such modeling often involves an array of processes and tools that present a challenge to collaboration and ensuring quality compliance. The objective of this work was to provide the pharmacometrics community with a suite of standardized, freely available, open-source tools to efficiently conduct traceable, reproducible, and scalable pharmacometric analyses in a controlled R environment [1].

METHODS Metrum Research Group has developed a suite of open-source tools that can be used independently, or seamlessly integrated into a larger R-based ecosystem, for pharmacometric analyses. We have compiled example code, and accompanying documentation, for each of the tasks typically required in a population pharmacokinetic (PK) analysis workflow to showcase the wide-ranging functionality of these tools. We considered R package management, data specifications, and common modeling and simulation activities such as, exploratory data analysis, executing and managing NONMEM® models and output, outcomes simulation, generating diagnostic plots and parameter tables, and analysis reporting. While we framed this example in the context of a simple population PK analysis, these packages can also be used in other types of pharmacometric analyses, e.g., pharmacokinetic-pharmacodynamic (PK-PD), quantitative systems pharmacology (QSP), and statistical models.
All our packages were developed and are maintained through an iterative open-source software development life cycle (SDLC) [2]. This transparent process incorporates robust planning, iterative development, validation, and release of each package.

RESULTS A complete, reproducible pharmacometric analysis workflow is hosted in a publicly available, version-controlled repository on GitHub [3] encompassing multiple states and stages of a modeling and simulation project. In addition to the scripted example, vignettes and user guides provide step-by-step directions detailing how the R packages facilitate an entire modeling and simulation analysis workflow. This package ecosystem includes: bbi/bbr, lastdose, Metrum Package Network (MPN) and pkgr, mrggsave, mrgsolve, pmforest, pmplots, pmtables and yspec.

CONCLUSIONS Metrum Research Group developed a suite of open-source R packages to support traceable, reproducible, and scalable pharmacometric analyses and now provides the pharmacometric community with an example of how to use these tools.

Workflow


Project Setup
mpn
pkgr
Curated reproducible package environment
Data
File transfer to Metworx, e.g. via Box
SDTM / ADAM
metworx
Data Assembly & Exploratory Data Analysis (EDA)
yspec
pm plots
pm tables
Generate analysis-ready datasets and create summary data tables / figures leveraging information in your data specification file
Model Development, Evaluation, Simulation & Reporting Workflow
bbr
pmforest
yspec
MRG SOLVE
pm plots
pm tables
Model management & summary scripts
Final model evaluation
Model-based inference
Modeling & simulation output
Structural model development
Full covariate model development
Goodness of fit diagnostics
Parameterized Rmd for model diagnostics
Bootstrap
Covariate effects
Visual Predictive Checks (VPCs)
Population simulations
MetrumRGs Formal Reports
Visual Studio Code
LaTeX
DocuSign
MetrumRGs Other Deliverables
Rmd
LaTeX Beamer

Package Showcase

yspec

The **yspec R package** is designed to help you document and manage your analysis datasets and then use this documentation throughout your project to:

- Guide and document the data assembly process
- Annotate figures and tables efficiently
- Manage decode values for numerically encoded discrete data items
- Make submission-ready define documents

yspec acts as a single, central location for maintaining the metadata for your analysis datasets to manage these activities.

1. Data specifications in yaml format

```
WT:
  short: weight
  unit: kg
  range: [40, 100]
SEX:
  short: sex
  values: [0, 1]
  decode: [male, female]
```

Continuous data item

Categorical data item

Analysis datasets are documented in yaml format and can be loaded into R as an object.

```
spec<-ys_load("spec/analysis3.yml")
```

Below we demonstrate how to use this spec object throughout all phases of project work.

2. Interactive query of the analysis dataset

Query categorical data item

```
spec$SEX
```

```
name   value
col    SEX
type   numeric
short  SEX
value  0 : male
         1 : female
```

Query continuous data item

```
spec$WT
```

```
name   value
col    WT
type   numeric
short  weight
unit   kg
range  .
```

3. Validating the dataset

Validate the contents of your dataset against the data specification (spec) file

```
ys_check(data, spec, error_on_fail=FALSE)
```

```
Messages:
 - spec has more items than cols in the data
 - names in spec but not in data:
    - AAG
#---------------------------------------
[1] FALSE
```

4. Annotating plots and tables for project deliverables

Extract the short label and units for use in figure axis column labels

```
contCovs <- axis_col_labs(specTex, c("AGE", "WT"), title_case = TRUE)
```

```
                AGE                 WT
"AGE//Age (years)"  "WT//Weight (kg)"
```

Extract the short label and units for use in tables

```
labs <- ys_get_short_unit(specTex, parens = TRUE, title_case=TRUE)
```

```
"Age (years)"  "Weight (kg)"
```

5. Generate define.pdf

Create a define.pdf suitable for regulatory submission

```
ys_document(spec, type = "regulatory", output_dir = "../data/derived",
    output_file = "define.pdf", author = "Kyle Baron")
```

1 Datasets

Description	Location
Example PopPK analysis data set	analysis3.xpt

1.1 Example PopPK analysis data set (`analysis3.xpt`)

VARIABLE	LABEL	TYPE	CODES
C	comment character	character	C = comment, . = non-comment
NUM	record number	numeric	
ID	subject identifier	numeric	
TIME	time after first dose (unit: hour)	numeric	

bbr

The **bbr R package** allows you to manage, track, and report modeling activities, through simple R objects, with a user-friendly interface between R and NONMEM®. bbr is used to submit models, process outputs and diagnostics, and iterate on models. It provides simple tagging and model inheritance trees to support replication and external review of your work.

1. Creating and submitting a model

Create your first model object from a NONMEM control stream

```
mod100 <- new_model(file.path(model_dir, 100))
```

Submit the NONMEM model to run

```
submit_model(mod100)
```

2. Iterative model development

Create a new model based on an existing model

```
mod101 <- copy_model_from(.parent_mod = mod100, .new_model = 101)
```

Compare a model ctl file to its parent model

```
model_diff(mod101)
```

3. Model evaluation and diagnostics

Parse NONMEM outputs into an R list object and use it for a quick look at the model specifics, and to check if any heuristic problems were detected

```
sum100 <- model_summary(mod100)
```

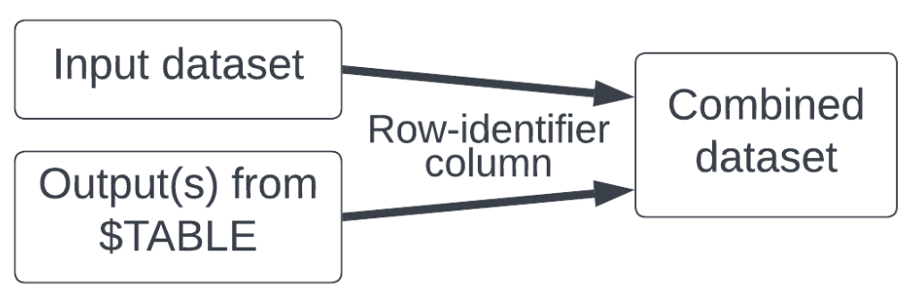
```
Dataset: ../../../data/derived/analysis3.csv
Records: 4292 Observations: 3142 Subjects: 160
Objective Function Value (final est. method): 33502.965
Estimation Method(s):
– First Order Conditional Estimation with Interaction
Heuristic Problem(s) Detected:
– parameter_near_boundary
```

Create a simple tibble with parameter estimates

```
sum100 %>% param_estimates()
```

```
parameter_names estimate  stderr random_effect_sd random_effect_sdse fixed diag  shrinkage
<chr>              <dbl>   <dbl>            <dbl>              <dbl> <lgl> <lgl>     <dbl>
THETA1            0.484   0.0636           NA                 NA      FALSE NA        NA
THETA2            4.14    0.0318           NA                 NA      FALSE NA        NA
THETA3            1.11    0.0350           NA                 NA      FALSE NA        NA
OMEGA(1,1)        0.225   0.0529            0.474              0.0557 FALSE TRUE      18.0
OMEGA(2,1)        0.0866  0.0258            0.466              0.103  FALSE FALSE     NA
OMEGA(2,2)        0.153   0.0172            0.392              0.0220 FALSE TRUE       3.17
SIGMA(1,1)        0.0411  0.00126           0.203              0.00311 FALSE TRUE      5.34
```

Combine the model output and input data, via a unique row-identifier column, to create a single combined data frame



```
data <- nm_join(mod100)
```

The yspec package (top left) can be used to decode the categorical covariates in your data.

```
data <- yspec_add_factors(data, spec)
```

```
   SEX  SEX_f
1:   1 female
2:   0   male
```

The pmplots box (top right) demonstrates how to make goodness of fit plots.

4. Model annotation

Add notes to your model

```
mod100 <- add_notes(mod100, "systematic bias, explore alternate structure")
```

Create highly customizable run logs to summarize model development

```
run_log(here("model/pk"), .recurse = FALSE) %>% add_summary() %>%
    select(run, based_on, ofv, param_count, notes)
```

run	based_on	ofv	param_count	notes
100	NULL	33502.96	10	systematic bias, explore alternate compartmental structure
101	100	31185.58	12	eta-V2 shows correlation with weight, consider adding allometric weight
102	101	30997.91	12	Allometric scaling with weight reduces eta-V2 correlation with weight. Will consider additional RUV structures, Proportional RUV performed best over additive (103) and combined (104)

pmplots

The **pmplots R package** allows you to make simple, standardized plots in a pharmacometric data analysis environment. The package provides a standard set of commonly used plots in pharmacometric analyses, as opposed to generating a new grammar of graphics.

Observed versus population predictions

```
p1 <- dv_pred(data)
```

Observed versus individual predictions

```
p2 <- dv_ipred(data)
```

Combine for output: `p1 + p2`

Conditional weighted residuals versus time

```
p3 <- cwres_time(data)
```

Residuals versus population predictions

```
p4 <- res_pred(data)
```

NPDEs versus categorical covariate

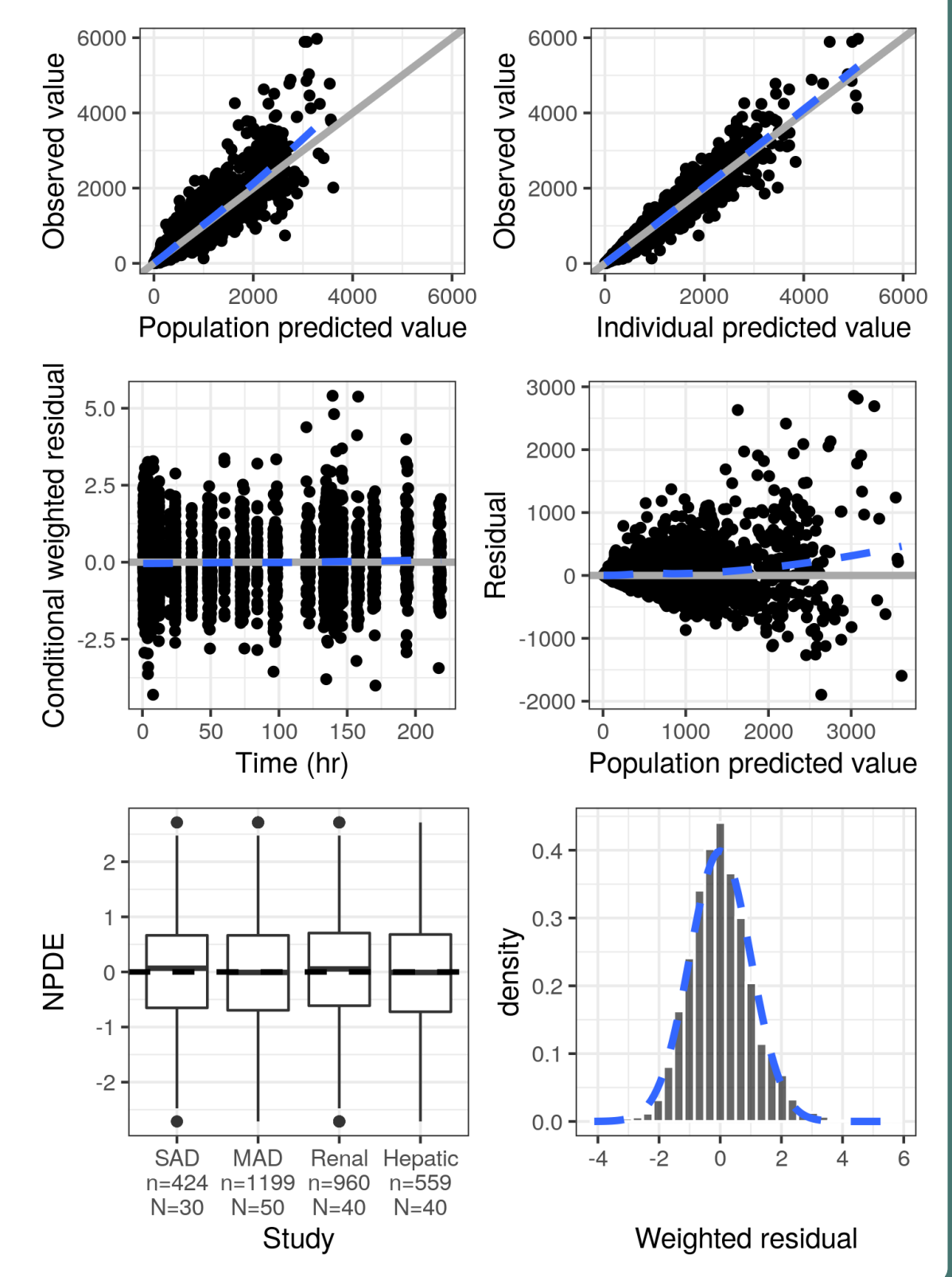
```
p5 <- npde_cat(data,
               x = "STUDYc//Study")
```

Histogram of weighted residuals

```
p6 <- wres_hist(data)
```

Combine for output

```
(p3 + p4) / (p5 + p6)
```



pmtables

The **pmtables R package** helps you create summary tables commonly used in pharmacometrics, such as data summaries and continuous or categorical covariate summaries. Further, it helps you turn any R table into a highly customized tex table suitable for reports generated with LaTeX or Markdown.

Data summary table

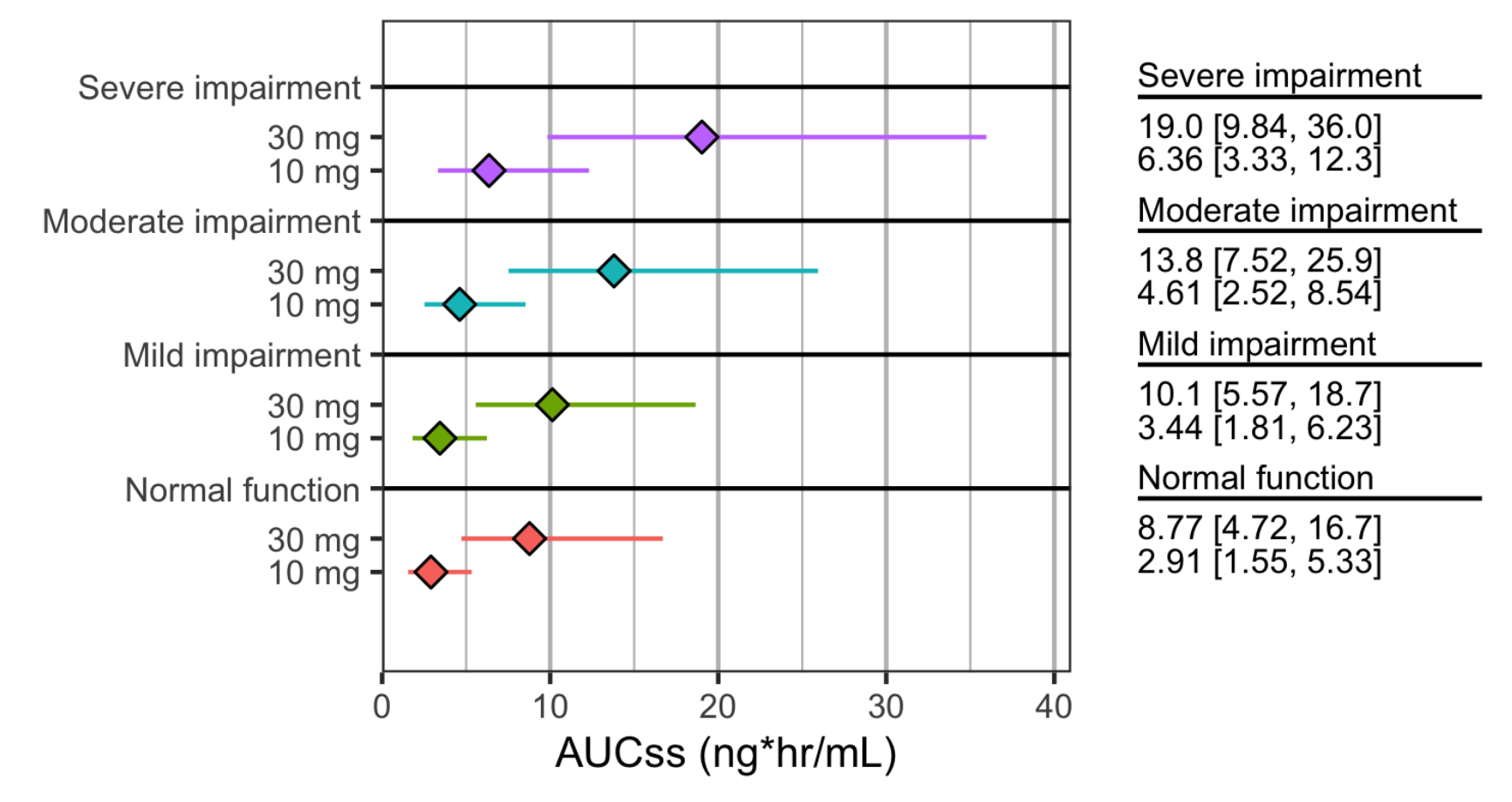
```
tab <- data %>%
  filter(EVID==0) %>%
  pt_data_inventory(by=c("Study"="STUDYc")
  stable()
```

	Number				Percent	
Study	SUBJ	MISS	OBS	BLQ	OBS	BLQ
SAD	30	0	424	0	13.5	0.0
MAD	50	0	1199	0	38.2	0.0
Renal	40	0	960	0	30.6	0.0
Hepatic	40	0	559	0	17.8	0.0
All data	160	0	3142	0	100.0	0.0

SUBJ: subjects
BLQ: below limit of quantification
MISS: missing observations (non-BLQ)
OBS: observations
Source code: poster-graphics.R
Source file: pk-data-sum.tex

pmforest

The **pmforest R package** helps you visualize the covariates of interest in your analysis by creating grouped displays of point estimates and variability ranges for any kind of continuous data.



Other Packages

- The **mrgsolve R package** allows you to simulate from ODE-based population PK/PD and QSP models.
- The **mrggsave R package** can be used to save images in bulk and annotate figures with the file names of the source code and the output image.
- The **lastdose R package** can be used to calculate time after dose.
- The **MPN (MetrumRG Package Network) and pkgr** tools allow you to create and manage curated, reproducible R package environments.

Visit our Expo website

You'll find:

- Our approach to project set-up, data assembly, modeling and simulation activities, and reporting.
- Access to example code in a Github repository.
- Information and vignettes on MetrumRG's suite of tools.

References

[1] Try Our Suite of Open-Source Tools. `https://metrumrg.com/try-open-source-tools/`. Accessed: 2022-9-27.

[2] Baron, K.T., Pastoor, D., Nevison, A., Kay, K. and Gastonguay, M.R. pmtables: TeX tables for pharmacometrics. In *Population Approach Group Europe Annual Meeting;* (2021).

[3] Metrum Research Group organization on GitHub. https://github.com/metrumresearchgroup (2022).