ggPMX Hands-On Exercises

## R version 4.1.0 (2021-05-18)  
## Platform: x86\_64-pc-linux-gnu (64-bit)  
## Running under: Red Hat Enterprise Linux  
##   
## Matrix products: default  
## BLAS/LAPACK: /CHBS/apps/EB/software/imkl/2019.1.144-gompi-2019a/compilers\_and\_libraries\_2019.1.144/linux/mkl/lib/intel64\_lin/libmkl\_gf\_lp64.so  
##   
## locale:  
## [1] LC\_CTYPE=en\_US.UTF-8 LC\_NUMERIC=C   
## [3] LC\_TIME=en\_US.UTF-8 LC\_COLLATE=en\_US.UTF-8   
## [5] LC\_MONETARY=en\_US.UTF-8 LC\_MESSAGES=en\_US.UTF-8   
## [7] LC\_PAPER=en\_US.UTF-8 LC\_NAME=C   
## [9] LC\_ADDRESS=C LC\_TELEPHONE=C   
## [11] LC\_MEASUREMENT=en\_US.UTF-8 LC\_IDENTIFICATION=C   
##   
## attached base packages:  
## [1] stats graphics grDevices utils datasets methods base   
##   
## other attached packages:  
## [1] ggplot2\_3.3.5 dplyr\_1.0.8 ggPMX\_1.2.8 nlmixr2\_2.0.6   
## [5] nlmixr2plot\_2.0.6 nlmixr2extra\_2.0.6 nlmixr2est\_2.0.8 nlmixr2data\_2.0.7   
## [9] rxode2\_2.0.8 rlang\_1.0.1   
##   
## loaded via a namespace (and not attached):  
## [1] Rcpp\_1.0.7 lattice\_0.20-44 zoo\_1.8-9   
## [4] vpc\_1.2.2 assertthat\_0.2.1 digest\_0.6.29   
## [7] utf8\_1.2.2 plyr\_1.8.6 R6\_2.5.1   
## [10] backports\_1.2.1 sys\_3.4 evaluate\_0.14   
## [13] pillar\_1.6.2 rstudioapi\_0.13 data.table\_1.14.0   
## [16] checkmate\_2.0.0 rmarkdown\_2.11 qs\_0.25.1   
## [19] stringr\_1.4.0 PreciseSums\_0.4 munsell\_0.5.0   
## [22] symengine\_0.1.5 compiler\_4.1.0 xfun\_0.25   
## [25] pkgconfig\_2.0.3 htmltools\_0.5.2 tidyselect\_1.1.1   
## [28] tibble\_3.1.3 reshape\_0.8.8 fansi\_0.5.0   
## [31] crayon\_1.4.2 withr\_2.4.3 grid\_4.1.0   
## [34] nlme\_3.1-152 GGally\_2.1.2 gtable\_0.3.0   
## [37] lifecycle\_1.0.1 DBI\_1.1.2 magrittr\_2.0.1   
## [40] scales\_1.1.1 RcppParallel\_5.1.4 cli\_3.1.1   
## [43] stringi\_1.7.3 cachem\_1.0.6 n1qn1\_6.0.1-10   
## [46] ellipsis\_0.3.2 generics\_0.1.0 vctrs\_0.3.8   
## [49] stringfish\_0.15.2 lotri\_0.4.2 RApiSerialize\_0.1.0  
## [52] RColorBrewer\_1.1-2 tools\_4.1.0 glue\_1.6.1   
## [55] purrr\_0.3.4 fastmap\_1.1.0 yaml\_2.2.1   
## [58] colorspace\_2.0-2 lbfgsb3c\_2020-3.2 memoise\_2.0.0   
## [61] knitr\_1.33

## Introduction

The purpose of this document is to to generate the diagnostic plots using ggPMX for the warfarin PKPD model using nlmixr.

**The material is designed to run on R 4.1.0 with nlmixr2 v2.0.6 and ggPMX v1.2.8.**

## Model Fitting

**Exercise 1:** Create a PKPD model using nlmixr2 (use the demo example below):

* Create the warfarin PKPD model using nlmixr2 (use demo code below)
* Fit the model to warfarin data (warfarin dataset is included by default in the nlmixr2 installation; it contains weight (wt), age and sex as covariates)
* Save the model fit as fit.rds in the working directory

Alternative: you can also use Monolix or NONMEM with any model if you wish. To best explore ggPMX capacities, it would be preferable if there are continuous and categorical covariates in your dataset.

# nlmixr2 model definition:  
  
# Warfarin example  
pk.turnover.emax3 <- function() {  
 ini({  
 tktr <- log(1)  
 tka <- log(1)  
 tcl <- log(0.1)  
 tv <- log(10)  
 ##  
 eta.ktr ~ 1  
 eta.ka ~ 1  
 eta.cl ~ 2  
 eta.v ~ 1  
 prop.err <- 0.1  
 pkadd.err <- 0.1  
 ##  
 temax <- logit(0.8)  
 tec50 <- log(0.5)  
 tkout <- log(0.05)  
 te0 <- log(100)  
 ##  
 eta.emax ~ .5  
 eta.ec50 ~ .5  
 eta.kout ~ .5  
 eta.e0 ~ .5  
 ##  
 pdadd.err <- 10  
 })  
 model({  
 ktr <- exp(tktr + eta.ktr)  
 ka <- exp(tka + eta.ka)  
 cl <- exp(tcl + eta.cl)  
 v <- exp(tv + eta.v)  
 emax = expit(temax+eta.emax)  
 ec50 = exp(tec50 + eta.ec50)  
 kout = exp(tkout + eta.kout)  
 e0 = exp(te0 + eta.e0)  
 ##  
 DCP = center/v  
 PD=1-emax\*DCP/(ec50+DCP)  
 ##  
 effect(0) = e0  
 kin = e0\*kout  
 ##  
 d/dt(depot) = -ktr \* depot  
 d/dt(gut) = ktr \* depot -ka \* gut  
 d/dt(center) = ka \* gut - cl / v \* center  
 d/dt(effect) = kin\*PD -kout\*effect  
 ##  
 cp = center / v  
 cp ~ prop(prop.err) + add(pkadd.err)  
 effect ~ add(pdadd.err) | pca  
 })  
}  
ui3 = nlmixr(pk.turnover.emax3)  
ui3  
  
# Dataset used to fit the model (imbedded in nlmixr2):  
summary(warfarin)  
  
# Fit the model to the data  
fit <- nlmixr(pk.turnover.emax3, warfarin, "saem", control=list(print=0),  
 table=list(cwres=TRUE, npde=TRUE))  
print(fit)  
  
saveRDS(fit,"fit.rds")

## ggPMX Controller

**Exercise 2:** ggPMX controller creation

* Create the controller **for PK only (DVID==“cp”)**
* Plot NPDE vs TIME
* Plot NPDE vs TIME for time > 50
* Plot NPDE vs PRED and change the labels of x and y axes
* Plot DV vs PRED on log-log scale
* Plot DV vs IPRED and stratify it by categorical covariates
* Plot EBE vs continuous covariates and EBE vs categorical covariates
* Stratify NPDE vs TIME by categorical covariates

Hints:

* To get help use ?pmx\_plot\_npde\_pred, ?pmx\_plot\_npde\_time, etc.

## ggPMX Report

**Exercise 3:** ggPMX report generation - **for PK only**

* Create a folder named GOF\_PK where will be stored the goodness-of-fit plots for the PK
* Generate Word and pdf ggPMX reports (named *ggPMX\_report\_pk.docx* and *ggPMX\_report\_pk.pdf*) in the GOF\_PK folder; use the option generating the diagnostic plots folder that contains each figure as a separate file.
* Generate an HTML report without footnote (the footnote indicates the source location of each figure) and without the folder with figures as separate files.

Hint: To get help on the various options, use ?pmx\_report.

**Exercise 4:** ggPMX Report Customization

* Edit your own template based on the file *ggPMX\_report\_pk.Rmd*.
* Set format to landscape in the new template.
* Change the color of the loess line to blue for the NPDE vs PRED plot and change the axis labels.
* Remove the loess line in the IWRES vs TIME plot.
* Filter TIME>30 in NPDE vs TIME plot.
* Set log-x and log-y axes for the DV vs PRED plot.
* Include all pages of individual plots (grid 4x4).
* Save the newly created Rmarkown file in another folder (name it *ggPMX\_report\_pk\_custom.Rmd*) and execute it using pmx\_report().

Hints:

* To get help use ?pmx\_plot\_npde\_pred, ?pmx\_plot\_npde\_time, ?pmx\_plot\_iwres\_time, ?pmx\_plot\_dv\_pred and ?pmx\_plot\_individual.
* Landscape format is set-up with classoption: landscape in the RMarkdown header

**Exercise 6:** ggPMX Report for PD

* Create a folder named GOF\_PD where will be stored the goodness-of-fit plots for the PD
* Generate a Word ggPMX report (named *ggPMX\_report\_pd.docx*) and the diagnostic plots folder for the PD endpoint (dvid==“pca”)
* Remove DRAFT label on all plots

Hints:

* First need to create the controller for PD endpoint (dvid==“pca”)
* Generate the report using the PD controller with the command pmx\_report(); output files should be stored in folder GOF\_PD
* Explore ?pmx\_settings