Package 'ggPMX'

June 25, 2020

Title 'ggplot2' Based Tool to Facilitate Diagnostic Plots for NLME Models

Description At Novartis, we aimed at standardizing the set of diagnostic plots used for modeling activities in order to reduce the overall effort required for generating such plots.
For this, we developed a guidance that proposes an adequate set of diagnostics and a toolbox, called 'ggPMX' to execute them. 'ggPMX' is a toolbox that can generate all diagnostic plots at a quality sufficient
for publication and submissions using few lines of code.

Version 1.1.2

Maintainer Matthew Fidler <matthew.fidler@gmail.com>

URL https://github.com/ggPMXdevelopment/ggPMX

BugReports https://github.com/ggPMXdevelopment/ggPMX/issues

Depends R (>= 3.5)

Imports data.table, yaml, R6, gtable, ggplot2 (>= 2.2.0), magrittr, stringr, assertthat, GGally, zoo, knitr, rmarkdown

License GPL-2

LazyData true

Suggests testthat, xtable, nlmixr

VignetteBuilder knitr

NeedsCompilation no

RoxygenNote 7.1.0.9000

Author Amine Gassem [aut],
Bruno Bieth [aut],
Irina Baltcheva [aut],
Thomas Dumortier [aut],
Christian Bartels [aut],
Souvik Bhattacharya [aut],
Inga Ludwig [aut],
Ines Paule [aut],
Didier Renard [aut],
Matthew Fidler [ctb, cre] (<https://orcid.org/0000-0001-8538-6691>),

Qing Xi Ooi [ctr],
Novartis Pharma AG [cph]

Repository CRAN

Date/Publication 2020-06-25 15:50:09 UTC

R topics documented:

`abbrev` *Give the whole abbreviation definition*

Description

Give the whole abbreviation definition

Usage

```
abbrev(param)
```

Arguments

`param`	abbreviation term

Value

character abbreviation definition

Examples

```
abbrev("VPC")
```

`add_draft` *Add draft layer annotation*

Description

This function adds the word draft to certain graphics.

Usage

```
add_draft(
  label = "DRAFT",
  size = 10,
  colour = "grey50",
  x = Inf,
  y = -Inf,
  ...
)
```

Arguments

label	draft layer default to DRAFT
size	size of the annotation
colour	color of the annotation default to grey50
x	numeric x coordinate of the draft label
y	numeric y coordinate of the draft label
...	extra parameters to geom text used to annotate the draft

Value

ggplot2 annotation

---

distrib *creates a graphic distribution object*

Description

creates a graphic distribution object

Usage

```
distrib(
  labels,
  is.shrink,
  type = c("box", "hist"),
  is.jitter = FALSE,
  jitter = NULL,
  facets = NULL,
  histogram = NULL,
  shrink = NULL,
  dname = NULL,
  ...
)
```

Arguments

labels	list of texts/titles used within the plot
is.shrink	logical if TRUE add shrinkage layer
type	box for boxplot or histogram
is.jitter	logical if TRUE add jitter operator for points
jitter	list set jitter parameter
facets	list set the facet setting in case of histogram plot
histogram	list histogram graphical parameters
shrink	list list of parameters to tune the shrinkage
dname	name of dataset to be used
...	others graphics arguments passed to pmx_gpar internal object.

Details

labels is a list that contains:

- **title:** plot title default "EBE distribution"
- **subtitle:** plot subtitle default empty
- **x:** x axis label default to "Etas"
- **y:** y axis label default to empty
- **legend:** legend title default to "random Effect"

shrink is a list that contains:

- **fun:** shrinkage function can be `sd` or `var`
- **size:** shrinkage text size
- **color:** shrinkage text color
- **vjust:** shrinkage position vertical adjustment

Value

distrib object

See Also

Other plot_pmx: `eta_cov()`, `eta_pairs()`, `individual()`, `plot_pmx.distrib()`, `plot_pmx.eta_cov()`, `plot_pmx.eta_pairs()`, `plot_pmx.individual()`, `plot_pmx.pmx_dens()`, `plot_pmx.pmx_gpar()`, `plot_pmx.pmx_qq()`, `plot_pmx.residual()`, `plot_pmx()`

---

eta_cov *This creates an ETA covariance matrix which can be used to define the co-relation between the parameters and its shrinkage..*

Description

This creates an ETA covariance matrix which can be used to define the co-relation between the parameters and its shrinkage..

Usage

```
eta_cov(
  labels,
  type = c("cats", "conts"),
  dname = NULL,
  show.correl = TRUE,
  correl = NULL,
  facets = NULL,
  point = NULL,
  covariates = NULL,
  ...
)
```

Arguments

labels	list of texts/titles used within the plot
type	box for cats or conts
dname	name of dataset to be used
show.correl	logical if TRUE add correlation to the plot
correl	list correl geom text graphical parameter
facets	list facetting graphical parameter
point	list geom point graphical parameter
covariates	pmxCOVObject pmx_cov
...	others graphics arguments passed to pmx_gpar internal object.

Details

labels is a list that contains:

- **title:** plot title default "EBE vs. covariates"
- **x:** x axis label default to "Etas"
- **y:** y axis label default to empty

Value

eta_cov object

See Also

Other plot_pmx: distrib(), eta_pairs(), individual(), plot_pmx.distrib(), plot_pmx.eta_cov(), plot_pmx.eta_pairs(), plot_pmx.individual(), plot_pmx.pmx_dens(), plot_pmx.pmx_gpar(), plot_pmx.pmx_qq(), plot_pmx.residual(), plot_pmx()

---

eta_cov_plot *Eta Covariates plots*

Description

Eta Covariates plots

Relationships between (ETA) and categorical covariates

Relationships between (ETA) and continuous covariates

Usage

```
dummy(
  dname,
  show.correl,
  correl,
  point,
  facets,
  filter,
  strat.facet,
  strat.color,
  trans,
  pmxgpar,
  labels,
  axis.title,
  axis.text,
  ranges,
  is.smooth,
  smooth,
  is.band,
  band,
  is.draft,
  draft,
  is.identity_line,
  identity_line,
  scale_x_log10,
  scale_y_log10,
  color.scales
)

pmx_plot_eta_cats(ctr, ...)

pmx_plot_eta_conts(ctr, ...)
```

Arguments

dname	character name of dataset to be used
show.correl	logical if TRUE add correlation to the plot
correl	list correl geom text graphical parameter
point	list geom point graphical parameter
facets	list facetting graphical parameter
	pmx_update parameters
filter	expression filter which will be applied to plotting data.
strat.facet	formula optional stratification parameter by facetting. This split plot by strats(each strat in a facet)
strat.color	character optional stratification parameter by grouping. This will split the plot by group (color) of strat.

trans	character define the transformation to apply on x or y or both variables
pmxgpar	a object of class pmx_gpar possibly the output of the
	pmx_gpar: Shared basic graphics parameters
labels	list list containing plot and/or axis labels: title, subtitle, x , y
axis.title	list containing element_text attributes to customize the axis title. (similar to ggplot2 axis.title theme)
axis.text	list containing element_text attributes to customize the axis text (similar to ggplot2 axis.text theme)
ranges	list limits of x/y ranges
is.smooth	logical if set to TRUE add smooth layer
smooth	list geom_smooth graphical/smoothing fun parameters
is.band	logical if TRUE add horizontal band
band	list horizontal band parameters. geom_hline graphical parameters.
is.draft	logical if TRUE add draft layer
draft	list draft layer parameters. geom_text graphical parameters.
is.identity_line	logical if TRUE add an identity line
identity_line	listgeom_abline graphical parameters.
scale_x_log10	logical if TRUE use log10 scale for x axis.
scale_y_log10	logical if TRUE use log10 scale for y axis.
color.scales	list define scales parameter in case of strat.color pmx_settings
ctr	pmx controller
...	others graphics parameters passed :
	• pmx_gpar internal function to customize shared graphical parameters
	• eta_cov generic object for eta/covariates plots.
	• pmx_update function.
	eta_cov parameters

Value

ggplot2 object

Examples

```
# basic use  ---------------------------------------

ctr <- theophylline()
ctr %>% pmx_plot_eta_cats
ctr %>% pmx_plot_eta_conts
```

```
# update graphical parameter  ----------------------

## update labels
ctr %>% pmx_plot_eta_cats(
  labels = list(title = "New eta cats title")
)

## remove draft
ctr %>% pmx_plot_eta_cats(is.draft = FALSE)

## change text color line
ctr %>% pmx_plot_eta_conts(
  correl=list(colour="magenta")
  )

## set covariates custom labels

ctr %>% pmx_plot_eta_conts(
  covariates=pmx_cov(values=list("WT0","AGE0"),
                     labels=list("Weight","Age"))
)

## set effects and covaraites custom labels


ctr <- theophylline( settings = pmx_settings(
  effects=list( levels=c("ka", "V", "Cl"),
                labels=c("Concentration","Volume","Clearance")
  )
)
)
ctr %>% pmx_plot_eta_conts(
  covariates=pmx_cov(values=list("WT0","AGE0"),
                     labels=list("Weight","Age"))
)
```

`eta_distribution_plot` *Eta distribution plots*

Description

Eta distribution plots

Eta Distribution boxplot

Eta Distribution histogram plot

Usage

```
eta_distribution_plot(
  jitter,
  type,
  dname,
  is.shrink,
  shrink,
  is.jitter,
  histogram,
  filter,
  strat.facet,
  facets,
  strat.color,
  trans,
  pmxgpar,
  labels,
  axis.title,
  axis.text,
  ranges,
  is.smooth,
  smooth,
  is.band,
  band,
  is.draft,
  draft,
  is.identity_line,
  identity_line,
  scale_x_log10,
  scale_y_log10,
  color.scales,
  ...
)

pmx_plot_eta_box(ctr, ...)

pmx_plot_eta_hist(ctr, ...)
```

Arguments

jitter	list set jitter parameter
type	box for boxplot or histogram
dname	name of dataset to be used
is.shrink	logical if TRUE add shrinkage layer
shrink	list list of parameters to tune the shrinkage
is.jitter	logical if TRUE add jitter operator for points
histogram	list histogram graphical parameters
	pmx_update parameters

filter	expression filter which will be applied to plotting data.
strat.facet	formula optional stratification parameter by facetting. This split plot by strats(each strat in a facet)
facets	list facet_wrap parameters.
strat.color	character optional stratification parameter by grouping. This will split the plot by group (color) of strat.
trans	character define the transformation to apply on x or y or both variables
pmxgpar	a object of class pmx_gpar possibly the output of the
	pmx_gpar: Shared basic graphics parameters
labels	list list containing plot and/or axis labels: title, subtitle, x , y
axis.title	list containing element_text attributes to customize the axis title. (similar to ggplot2 axis.title theme)
axis.text	list containing element_text attributes to customize the axis text (similar to ggplot2 axis.text theme)
ranges	list limits of x/y ranges
is.smooth	logical if set to TRUE add smooth layer
smooth	list geom_smooth graphical/smoothing fun parameters
is.band	logical if TRUE add horizontal band
band	list horizontal band parameters. geom_hline graphical parameters.
is.draft	logical if TRUE add draft layer
draft	list draft layer parameters. geom_text graphical parameters.
is.identity_line	logical if TRUE add an identity line
identity_line	listgeom_abline graphical parameters.
scale_x_log10	logical if TRUE use log10 scale for x axis.
scale_y_log10	logical if TRUE use log10 scale for y axis.
color.scales	list define scales parameter in case of strat.color pmx_settings
...	others graphics parameters passed :
	• pmx_gpar internal function to customize shared graphical parameters
	• distrib generic object for distribution plots (histogram/boxplot).
	• pmx_update function.
	distrib parameters
ctr	pmx controller

Value

ggplot2 object

Examples

```
# *************** basic use ***************** ------

ctr <- theophylline()
## boxplot variation
p <- ctr %>% pmx_plot_eta_box
## histogram variation
p <- ctr %>% pmx_plot_eta_hist()

# update graphical parameter  ----------------------

## add jitter
ctr %>%
  pmx_plot_eta_hist(is.jitter = TRUE, jitter = list(alpha = 0.4, color = "red"))

## remove shrinkage
ctr %>%
  pmx_plot_eta_hist(is.shrink = FALSE)

## updat histogram graphical parameters
ctr %>%
  pmx_plot_eta_hist(
    histogram = list(
      color = NA,
      position = "fill",
      binwidth = 1 / 100)
    )



# stratification  ---------------------------------

## categorical stratification color parameter
ctr %>% pmx_plot_eta_hist(is.jitter=TRUE,strat.facet=~STUD,strat.color="SEX")
## categorical stratification facetting
ctr %>% pmx_plot_eta_hist(strat.facet = "SEX")
## using formula categorical stratification facetting
ctr %>% pmx_plot_eta_hist(strat.facet = STUD~SEX,
                          shrink=list(hjust=0.5))

# subsetting  -------------------------------------

## select a set of random effect
ctr %>% pmx_plot_eta_hist(filter = EFFECT %in% c("ka", "Cl"))
## filter and stratify by facets
ctr %>% pmx_plot_eta_hist(
  filter = EFFECT %in% c("ka", "Cl"), strat.facet = "SEX"
)
ctr %>% pmx_plot_eta_hist(
  filter = EFFECT %in% c("ka", "Cl"), strat.facet = "SEX"
)
```

`eta_pairs` *This creates an eta correlation which defines the relationship between parameters*

Description

This creates an eta correlation which defines the relationship between parameters

Usage

```
eta_pairs(
  title,
  dname = NULL,
  type.eta = c("mode", "mean"),
  text_color = "black",
  is.shrink = TRUE,
  is.smooth = TRUE,
  smooth = NULL,
  point = NULL,
  shrink = NULL,
  is.hline = FALSE,
  hline = NULL,
  ...
)
```

Arguments

`title`	character the plot title
`dname`	name of dataset to be used
`type.eta`	`character` type of eat can be 'mode' or 'mean'.'mode' by default
`text_color`	color of the correlation text in the upper matrix
`is.shrink`	`logical` if TRUE add shrinkage to the plot
`is.smooth`	`logical` if TRUE add smoothing to lower matrix plots
`smooth`	`list` geom_smooth graphical parameters
`point`	`list` geom_point graphical parameter
`shrink`	`list` shrinkage graphical parameter
`is.hline`	`logical` if TRUE add horizontal line to lower matrix plots
`hline`	`list` geom_hline graphical parameters
`...`	others graphics arguments passed to `pmx_gpar` internal object.

Value

ecorrel object

See Also

Other plot_pmx: distrib(), eta_cov(), individual(), plot_pmx.distrib(), plot_pmx.eta_cov(), plot_pmx.eta_pairs(), plot_pmx.individual(), plot_pmx.pmx_dens(), plot_pmx.pmx_gpar(), plot_pmx.pmx_qq(), plot_pmx.residual(), plot_pmx()

FacetWrapPaginate *Extend facet_wrap using ggproto*

Description

Extend facet_wrap using ggproto

Usage

```
FacetWrapPaginate
```

---

facet_wrap_paginate *Split facet_wrap over multiple plots*

Description

This extension to facet_wrap will allow user to split a facetted plot over multiple pages. User define the specific number of rows and columns per page as well as the page number to plot, and the function will automatically plot in the correct panels. This will be rendered in a loop to plot pages one by one.

Usage

```
facet_wrap_paginate(
  facets,
  nrow = NULL,
  ncol = NULL,
  scales = "fixed",
  shrink = TRUE,
  labeller = "label_value",
  as.table = TRUE,
  switch = NULL,
  drop = TRUE,
  dir = "h",
  strip.position = "top",
  page = 1
)
```

Arguments

facets	A set of variables or expressions quoted by vars() and defining faceting groups on the rows or columns dimension. The variables can be named (the names are passed to labeller). For compatibility with the classic interface, can also be a formula or character vector. Use either a one sided formula, ~a + b, or a character vector, c("a","b").
nrow	Number of rows and columns.
ncol	Number of rows and columns
scales	Should scales be fixed ("fixed", the default), free ("free"), or free in one dimension ("free_x", "free_y")?
shrink	If TRUE, will shrink scales to fit output of statistics, not raw data. If FALSE, will be range of raw data before statistical summary.
labeller	A function that takes one data frame of labels and returns a list or data frame of character vectors. Each input column corresponds to one factor. Thus there will be more than one with vars(cyl,am). Each output column gets displayed as one separate line in the strip label. This function should inherit from the "labeller" S3 class for compatibility with labeller(). You can use different labeling functions for different kind of labels, for example use label_parsed() for formatting facet labels. label_value() is used by default, check it for more details and pointers to other options.
as.table	If TRUE, the default, the facets are laid out like a table with highest values at the bottom-right. If FALSE, the facets are laid out like a plot with the highest value at the top-right.
switch	By default, the labels are displayed on the top and right of the plot. If "x", the top labels will be displayed to the bottom. If "y", the right-hand side labels will be displayed to the left. Can also be set to "both".
drop	If TRUE, the default, all factor levels not used in the data will automatically be dropped. If FALSE, all factor levels will be shown, regardless of whether or not they appear in the data.
dir	Direction: either "h" for horizontal, the default, or "v", for vertical.
strip.position	By default, the labels are displayed on the top of the plot. Using strip.position it is possible to place the labels on either of the four sides by setting strip.position = c("top","bottom","left","right")
page	The page to draw

Note

If either ncol or nrow is NULL this function will fall back to the standard facet_wrap functionality.

getPmxOption *Get ggPMX Option*

Description

Get ggPMX Option

Usage

```
getPmxOption(name, default = NULL)
```

Arguments

name	Name of an option to get.
default	Value to be returned if the option is not currently set.

Examples

```
## Not run:
pmxOptions(myOption = 10)
getPmxOption("myOption")

## End(Not run)
```

get_abbrev *Get abbreviation definition by key*

Description

Get abbreviation definition by key

Usage

```
get_abbrev(ctr, param)
```

Arguments

ctr	pmxClass controller
param	abbreviation term

Value

character abbreviation definition

get_cats *Get category covariates*

Description

Get category covariates

Usage

```
get_cats(ctr)
```

Arguments

ctr	the controller object

Value

a charcater vector

See Also

Other pmxclass: get_conts(), get_covariates(), get_data(), get_occ(), get_plot_config(), get_plot(), get_strats(), plot_names(), plots(), pmx_update(), set_data(), set_plot()

get_conts *Get continuous covariates*

Description

Get continuous covariates

Usage

```
get_conts(ctr)
```

Arguments

ctr	the controller object

Value

a charcater vector

See Also

Other pmxclass: get_cats(), get_covariates(), get_data(), get_occ(), get_plot_config(), get_plot(), get_strats(), plot_names(), plots(), pmx_update(), set_data(), set_plot()

get_covariates *Get covariates variables*

Description

Get covariates variables

Usage

```
get_covariates(ctr)
```

Arguments

ctr	the controller object

Value

a charcater vector

See Also

Other pmxclass: get_cats(), get_conts(), get_data(), get_occ(), get_plot_config(), get_plot(), get_strats(), plot_names(), plots(), pmx_update(), set_data(), set_plot()

get_data *Get controller data set*

Description

Get controller data set

Usage

```
get_data(
  ctr,
  data_set = c("estimates", "predictions", "eta", "finegrid", "input", "sim",
    "individual")
)
```

Arguments

ctr	the controller object
data_set	the data set name

Value

a data.table of the named data set if available.

See Also

Other pmxclass: get_cats(), get_conts(), get_covariates(), get_occ(), get_plot_config(), get_plot(), get_strats(), plot_names(), plots(), pmx_update(), set_data(), set_plot()

`get_occ` *Get controller occasional covariates*

Description

Get controller occasional covariates

Usage

```
get_occ(ctr)
```

Arguments

ctr	the controller object

Value

a charcater vector

See Also

Other pmxclass: get_cats(), get_conts(), get_covariates(), get_data(), get_plot_config(), get_plot(), get_strats(), plot_names(), plots(), pmx_update(), set_data(), set_plot()

`get_plot` *Get plot object*

Description

Get plot object

Usage

```
get_plot(ctr, nplot, npage = NULL)
```

Arguments

ctr	pmxClass controller object
nplot	character the plot name
npage	integer or integer vector, set page number in case of multi pages plot

Value

ggplot object

See Also

Other pmxclass: get_cats(), get_conts(), get_covariates(), get_data(), get_occ(), get_plot_config(), get_strats(), plot_names(), plots(), pmx_update(), set_data(), set_plot()

Examples

```
library(ggPMX)
ctr <- theophylline()
p1 <- ctr %>% get_plot("iwres_ipred")
## get all pages or some pages
p2 <- ctr %>% get_plot("individual")
## returns one page of individual plot
p2 <- ctr %>% get_plot("individual", npage = 1)
p3 <- ctr %>% get_plot("individual", npage = c(1, 3))
## get distribution plot
pdistri <- ctr %>% get_plot("eta_hist")
```

---

get_plot_config *Get the plot config by name*

Description

Get the plot config by name

Usage

```
get_plot_config(ctr, pname)
```

Arguments

ctr	the controller object
pname	the plot name

Value

the config object

See Also

Other pmxclass: get_cats(), get_conts(), get_covariates(), get_data(), get_occ(), get_plot(), get_strats(), plot_names(), plots(), pmx_update(), set_data(), set_plot()

Examples

```
ctr <- theophylline()
ctr %>% set_plot("IND", pname = "indiv1")
ctr %>% get_plot_config("distr1")
```

`get_strats` *Get extra stratification variables*

Description

Get extra stratification variables

Usage

```
get_strats(ctr)
```

Arguments

`ctr`	the controller object

Value

a charcater vector

See Also

Other pmxclass: `get_cats()`, `get_conts()`, `get_covariates()`, `get_data()`, `get_occ()`, `get_plot_config()`, `get_plot()`, `plot_names()`, `plots()`, `pmx_update()`, `set_data()`, `set_plot()`

`ggPMX` *ggPMX: A ggplot2 toolbox for Nonlinear Mixed-Effect Model graphical*

Description

This package aims to generate diagnostic plots in a standard way. The tool reads data from many sources (MONOLIX,NONMEM,others) and generates standard grahics that can be easily integrated in a single report.

Details

- Get data from different system and create a data source
- Plot many plots using the generic plot method `plot_pmx`.

For support, feedback or bug reports, please reach out to <ggPMX_ORG@dl.mgd.novartis.com>.

Version History

Jan 11 2017, 0.0.0 Init ggPMX from Novartis rtemplate.

Feb 06 2017, 0.3.0 Import version 0.3.0 of package.

`gtable_remove_grobs` *Remove named elements from gtable*

Description

Remove named elements from gtable

Usage

```
gtable_remove_grobs(table, names, ...)
```

Arguments

`table`	The table from which grobs should be removed
`names`	A character vector of the grob names (as listed in `table$layout`) that should be removed
`...`	Other parameters passed through to `gtable_filter`.

`individual` *This function can be used to obtain individual prediction and compare with observed data and population prediction for each individual separately*

Description

This function can be used to obtain individual prediction and compare with observed data and population prediction for each individual separately

Usage

```
individual(
  labels,
  facets = NULL,
  dname = NULL,
  ipred_line = NULL,
  pred_line = NULL,
  point = NULL,
  bloq = NULL,
  is.legend,
  use.finegrid,
  ...
)
```

Arguments

labels	plot texts. labels, axis,
facets	list facets settings nrow/ncol
dname	name of dataset to be used
ipred_line	list some pred line geom properties aesthetics
pred_line	list some ipred line geom properties aesthetics
point	list some point geom properties aesthetics
bloq	pmxBLOQ object created by pmx_bloq
is.legend	logical if TRUE add a legend
use.finegrid	logical if FALSE use predictions data set
...	others graphics arguments passed to pmx_gpar internal object.

Value

individual fit object

See Also

plot_pmx.individual

Other plot_pmx: distrib(), eta_cov(), eta_pairs(), plot_pmx.distrib(), plot_pmx.eta_cov(), plot_pmx.eta_pairs(), plot_pmx.individual(), plot_pmx.pmx_dens(), plot_pmx.pmx_gpar(), plot_pmx.pmx_qq(), plot_pmx.residual(), plot_pmx()

---

input_finegrid *Merge input and fingrid data sets*

Description

Merge input and fingrid data sets

Usage

```
input_finegrid(input, finegrid)
```

Arguments

input	data.table input data set
finegrid	data.table finegrid data set

Value

data.table

is.pmx_gpar *Check if an object is a pmx_gpar class*

Description

Check if an object is a pmx_gpar class

Usage

```
is.pmx_gpar(x)
```

Arguments

x	pmx_gpar object

Value

logical returns TRUE if it is a pmx_gpar object

load_config *Obtain the data source config*

Description

Obtain the data source config

Usage

```
load_config(x, sys = c("mlx", "nm", "mlx18"))
```

Arguments

x	the config name.
sys	can be mlx,nm,...

Value

a list :data configuration object

load_data_set *Load data set*

Description

Load data set

Usage

```
load_data_set(x, path, sys, ...)
```

Arguments

x	data set config
path	character path to the directory
sys	character mlx or nm
...	extra parameter passed to special readers

Value

data.table

load_source *Load all/or some source data set*

Description

Load all/or some source data set

Usage

```
load_source(sys, path, dconf, ...)
```

Arguments

sys	type cane mlx/nom
path	character directory path containing all sources.
dconf	configuration object
...	any extra parameters for readers

Value

list of data.table

l_left_join *Merge 2 lists*

Description

left join , the first list is updated by the seond one

Usage

```
l_left_join(base_list, overlay_list, recursive = TRUE)
```

Arguments

base_list	list to update
overlay_list	list used to update the first list
recursive	logical if TRUE do the merge in depth

Value

list

n_pages *Determine the number of pages in a paginated facet plot*

Description

This is a simple helper that returns the number of pages it takes to plot all panels when using facet_wrap_paginate . It partially builds the plot so depending on the complexity of your plot it might take some time to calculate...

Usage

```
n_pages(plot)
```

Arguments

plot	A ggplot object using either facet_wrap_paginate or facet_grid_paginate

Value

If the plot uses using either facet_wrap_paginate or facet_grid_paginate it returns the total number of pages. Otherwise it returns NULL

parse_mlxtran *Parse MONOLIX mlxtran file*

Description

Parse MONOLIX mlxtran file

Usage

```
parse_mlxtran(file_name)
```

Arguments

file_name	absolute path to mlxtran file

Value

list key/values to initialize ggPMX controller

pk_occ *Creates pmx controller using monlix data having Occasional variable*

Description

Creates pmx controller using monlix data having Occasional variable

Usage

```
pk_occ()
```

Value

pmx controller

Examples

```
## Not run:
pk_occ()

## End(Not run)
```

pk_pd *Creates pkpd pmx controller using package internal data*

Description

Creates pkpd pmx controller using package internal data

Usage

```
pk_pd(code = "3")
```

Arguments

code	can be 3 or 4

plots *Get plots description*

Description

Get plots description

Usage

```
plots(ctr)
```

Arguments

ctr	pmxClass controller object

Value

data.frame of plots

See Also

Other pmxclass: get_cats(), get_conts(), get_covariates(), get_data(), get_occ(), get_plot_config(), get_plot(), get_strats(), plot_names(), pmx_update(), set_data(), set_plot()

plot_names *Get plot names*

Description

Get plot names

Usage

```
plot_names(ctr)
```

Arguments

ctr	pmxClass controller object

Value

list of plot names

See Also

Other pmxclass: get_cats(), get_conts(), get_covariates(), get_data(), get_occ(), get_plot_config(), get_plot(), get_strats(), plots(), pmx_update(), set_data(), set_plot()

plot_pmx *This is a generic plot method that produces all plots by default described in pmx model evaluation guidance.*

Description

This is a generic plot method that produces all plots by default described in pmx model evaluation guidance.

Usage

```
plot_pmx(x, dx, ...)
```

Arguments

x	object to plot
dx	data.table , plot source data
...	extra argument (not used)

See Also

pmx_gpar.

Other plot_pmx: distrib(), eta_cov(), eta_pairs(), individual(), plot_pmx.distrib(), plot_pmx.eta_cov(), plot_pmx.eta_pairs(), plot_pmx.individual(), plot_pmx.pmx_dens(), plot_pmx.pmx_gpar(), plot_pmx.pmx_qq(), plot_pmx.residual()

---

plot_pmx.distrib *Plot EBE distribution*

Description

Plot EBE distribution

Usage

```
## S3 method for class 'distrib'
plot_pmx(x, dx, ...)
```

Arguments

x	distribution object
dx	data set
...	not used for the moment

Value

ggplot2 plot

See Also

distrib

Other plot_pmx: distrib(), eta_cov(), eta_pairs(), individual(), plot_pmx.eta_cov(), plot_pmx.eta_pairs(), plot_pmx.individual(), plot_pmx.pmx_dens(), plot_pmx.pmx_gpar(), plot_pmx.pmx_qq(), plot_pmx.residual(), plot_pmx()

`plot_pmx.eta_cov` *This plots an ETA covariance matrix which can be used to define the co-relation between the parameters and its shrinkage*

Description

This plots an ETA covariance matrix which can be used to define the co-relation between the parameters and its shrinkage

Usage

```
## S3 method for class 'eta_cov'
plot_pmx(x, dx, ...)
```

Arguments

`x`	eta_cov object
`dx`	data set
`...`	not used for the moment

Value

ggplot2 plot

See Also

`eta_cov`

Other plot_pmx: `distrib()`, `eta_cov()`, `eta_pairs()`, `individual()`, `plot_pmx.distrib()`, `plot_pmx.eta_pairs()`, `plot_pmx.individual()`, `plot_pmx.pmx_dens()`, `plot_pmx.pmx_gpar()`, `plot_pmx.pmx_qq()`, `plot_pmx.residual()`, `plot_pmx()`

`plot_pmx.eta_pairs` *Plot random effect correlation plot*

Description

Plot random effect correlation plot

Usage

```
## S3 method for class 'eta_pairs'
plot_pmx(x, dx, ...)
```

Arguments

x	distribution object
dx	data set
...	not used for the moment

Value

ggpairs plot

See Also

distrib

Other plot_pmx: distrib(), eta_cov(), eta_pairs(), individual(), plot_pmx.distrib(), plot_pmx.eta_cov(), plot_pmx.individual(), plot_pmx.pmx_dens(), plot_pmx.pmx_gpar(), plot_pmx.pmx_qq(), plot_pmx.residual(), plot_pmx()

---

plot_pmx.individual *This function can be used to plot individual prediction and compare with observed data and population prediction for each individual separately*

Description

This function can be used to plot individual prediction and compare with observed data and population prediction for each individual separately

Usage

```
## S3 method for class 'individual'
plot_pmx(x, dx, ...)
```

Arguments

x	individual object
dx	data set
...	not used for the moment

Value

a list of ggplot2

See Also

Other plot_pmx: distrib(), eta_cov(), eta_pairs(), individual(), plot_pmx.distrib(), plot_pmx.eta_cov(), plot_pmx.eta_pairs(), plot_pmx.pmx_dens(), plot_pmx.pmx_gpar(), plot_pmx.pmx_qq(), plot_pmx.residual(), plot_pmx()

plot_pmx.pmx_dens *This function plot EBE versus covariates using qq plots*

Description

This function plot EBE versus covariates using qq plots

Usage

```
## S3 method for class 'pmx_dens'
plot_pmx(x, dx, ...)
```

Arguments

x	eta_cov object
dx	data set
...	not used for the moment

Value

ggplot2 plot

See Also

eta_cov

Other plot_pmx: distrib(), eta_cov(), eta_pairs(), individual(), plot_pmx.distrib(), plot_pmx.eta_cov(), plot_pmx.eta_pairs(), plot_pmx.individual(), plot_pmx.pmx_gpar(), plot_pmx.pmx_qq(), plot_pmx.residual(), plot_pmx()

plot_pmx.pmx_gpar *The ggPMX base plot function*

Description

This function should be called internally by other plots to set general settings like , smoothing, add band, labelling, theming,...

Usage

```
## S3 method for class 'pmx_gpar'
plot_pmx(gpar, p)
```

Arguments

gpar	object of pmx_gpar type
p	plot

Value

ggplot2 object

See Also

Other plot_pmx: distrib(), eta_cov(), eta_pairs(), individual(), plot_pmx.distrib(), plot_pmx.eta_cov(), plot_pmx.eta_pairs(), plot_pmx.individual(), plot_pmx.pmx_dens(), plot_pmx.pmx_qq(), plot_pmx.residual(), plot_pmx()

---

plot_pmx.pmx_qq *This function plot EBE versus covariates using qq plots*

Description

This function plot EBE versus covariates using qq plots

Usage

```
## S3 method for class 'pmx_qq'
plot_pmx(x, dx, ...)
```

Arguments

x	pmx_qq object
dx	data set
...	not used for the moment

Value

ggplot2 plot

See Also

eta_cov

Other plot_pmx: distrib(), eta_cov(), eta_pairs(), individual(), plot_pmx.distrib(), plot_pmx.eta_cov(), plot_pmx.eta_pairs(), plot_pmx.individual(), plot_pmx.pmx_dens(), plot_pmx.pmx_gpar(), plot_pmx.residual(), plot_pmx()

`plot_pmx.residual` *This function plots residual for each observed value by finding the difference between observed and predicted points. It also fits a distribution to the residual value.*

Description

This function plots residual for each observed value by finding the difference between observed and predicted points. It also fits a distribution to the residual value.

Usage

```
## S3 method for class 'residual'
plot_pmx(x, dx, ...)
```

Arguments

`x`	residual object
`dx`	data set
`...`	not used for the moment

Value

ggplot2 object

See Also

`residual`

Other plot_pmx: `distrib()`, `eta_cov()`, `eta_pairs()`, `individual()`, `plot_pmx.distrib()`, `plot_pmx.eta_cov()`, `plot_pmx.eta_pairs()`, `plot_pmx.individual()`, `plot_pmx.pmx_dens()`, `plot_pmx.pmx_gpar()`, `plot_pmx.pmx_qq()`, `plot_pmx()`

`plot_shrink` *Plot shrink in eta matric*

Description

Plot shrink in eta matric

Usage

```
plot_shrink(x, shrink.dx, shrink)
```

Arguments

x	pmx_gpar object
shrink.dx	data.table of shrinkage
shrink	list graphical parameter

Value

ggplot2 object

---

pmx *Create a pmx object*

Description

Create a pmx object from a data source

Usage

```
pmx(
  config,
  sys = c("mlx", "nm"),
  directory,
  input,
  dv,
  dvid,
  cats = NULL,
  conts = NULL,
  occ = NULL,
  strats = NULL,
  settings = NULL,
  endpoint = NULL,
  sim = NULL,
  bloq = NULL,
  id = NULL,
  time = NULL
)

pmx_mlx(
  config,
  directory,
  input,
  dv,
  dvid,
  cats,
  conts,
  occ,
```

```
  strats,
  settings,
  endpoint,
  sim,
  bloq,
  id,
  time
)

pmx_mlxtran(file_name, config = "standing", call = FALSE, endpoint, ...)
```

Arguments

config	Can be either : The complete path for the configuration file, the name of configuration within the built-in list of configurations, or a configuration object.
sys	the system name can be MLX/NM
directory	character modelling output directory.
input	character complete path to the modelling input file
dv	character the name of measurable variable used in the input modelling file
dvid	*[Optional]* character observation type parameter. This is mandatory in case of multiple endpoint (PKPD).
cats	*[Optional]*character vector of categorical covariates
conts	*[Optional]*character vector of continuous covariates
occ	*[Optional]*character occasional covariate variable name
strats	*[Optional]*character extra stratification variables
settings	*[Optional]*pmxSettingsClass pmx_settings shared between all plots
endpoint	pmxEndpointClass or integer or charcater defalut to NULL of the endpoint code. pmx_endpoint
sim	pmxSimClass default to NULL. pmx_sim
bloq	pmxBLOQClass default to NULL. pmx_bloq
id	*[Optional]* character the name of Indvidual variable used in the input modelling file
time	*[Optional]* character Time variable.
file_name	character mlxtran file path.
call	logical if TRUE the result is the parameters parsed
...	extra arguments passed to pmx_mlx.

Details

pmx_mlx is a wrapper to mlx for the MONOLIX system (sys="mlx")

pmx_mlxtran parses mlxtran file and guess pmx_mlx arguments. In case of multi endpoint the first endpoint is selected. You can though set the endpoint through the same argument. When you set call=TRUE,no controller is created but only the parameters parsed by mlxtran. This can be very helpful, in case you would like to customize parameters (adding settings vi pmx_settings, chnag eth edefault endpoint.)

Value

pmxClass controller object.

Examples

```
## Example to create the controller using theophylline data
theophylline <- file.path(system.file(package = "ggPMX"), "testdata",
                          "theophylline")
WORK_DIR <- file.path(theophylline, "Monolix")
input_file <- file.path(theophylline, "data_pk.csv")

## using only mandatory variables
ctr <- pmx(
  sys="mlx",
  config = "standing",
  directory = WORK_DIR,
  input = input_file,
  dv = "Y",
  dvid ="DVID"
)
## Using covariates
ctr <- pmx(
  sys="mlx",
  config = "standing",
  directory = WORK_DIR,
  input = input_file,
  dv = "Y",
  dvid ="DVID",
  cats=c("SEX"),
  conts=c("WT0","AGE0"),
  strats="STUD"
)
## using settings parameter
ctr <- pmx(
  sys="mlx",
  config = "standing",
  directory = WORK_DIR,
  input = input_file,
  dv = "Y",
  dvid ="DVID",
  settings=list(is.draft=FALSE)
)

## using mlxtran file
mlxtran_file <-
  file.path(system.file(package = "ggPMX"),
    "testdata","1_popPK_model","project.mlxtran")
pmx_mlxtran(mlxtran_file)

## mlxtran , call =TRUE to get the pmx_mlx argument parsed by  pmx_mlxtran
```

```
params <- pmx_mlxtran(mlxtran_file,call=TRUE)

str(params)
# $ directory: chr results_pathile
# $ input    : chr observation file path
# $ dv       : chr "DV"
# $ cats     : chr [1:4] "SEX" "RACE" "DISE" "ILOW"
# $ conts    : chr [1:4] "AGE0" "WT0" "HT0" "TRT"
# $ occ      : chr "ISS"
# $ dvid     : chr "YTYPE"
# $ endpoint :List of 5
# ..$ code     : chr "1"
# ..$ label    : chr ""
# ..$ unit     : chr ""
# ..$ file.code: chr "1"
# ..$ trans    : NULL
# ..- attr(*, "class")= chr "pmxEndpointClass"
# $ config   : chr "standing"
```

`pmxOptions` *This function can be used to set ggPMX options*

Description

`getPmxOption` retrieves the value of a ggPMX option. `ggPMXOptions` sets the value of ggPMX options; it can also be used to return a list of all currently-set ggPMX options.

Usage

```
pmxOptions(...)
```

Arguments

`...`	Options to set, with the form `name = value`.

Details

There is a global option set, which is available by default.

Options used in ggPMX

- **template_dir:** path to template directory

Examples

```
## Not run:
pmxOptions(template_dir = PATH_TO_CUSTOM_CONFIGURATION)

## End(Not run)
```

pmx_bloq *Creates BLOQ object attributes*

Description

Creates BLOQ object attributes

Usage

```
pmx_bloq(
  cens = "CENS",
  limit = "LIMIT",
  colour = "pink",
  size = 2,
  alpha = 0.9,
  show = TRUE,
  ...
)
```

Arguments

cens	character the censoring column name
limit	character the limit column name (optional)
colour	character the color of the geom
size	numeric the size of the geom
alpha	numeric the alpha of the geom
show	logical if FALSE remove all censory observations
...	any other graphical parameter

Details

To define that a measurement is censored, the observation data set should include a CENSORING column (default to 'CENS') and put 1 for lower limit or -1 for upper limit.
Optionally, data set can contain have a limit column (default to 'LIMIT') column to set the other limit.

pmx_comp_shrink *Compute Shrinkage*

Description

Compute Shrinkage

Usage

```
pmx_comp_shrink(
  ctr,
  fun = c("var", "sd"),
  strat.facet,
  strat.color,
  filter,
  ...
)
```

Arguments

ctr	pmxClass controller object
fun	character can be sd or var , var by default
strat.facet	formula optional stratification parameter
strat.color	character optional stratification parameter
filter	optional filter which will be applied to plotting data
...	others parameters not used for the moment

Value

data.table

pmx_config *This function can be used to define the pmx configuration used in plots. e.g. Monolox/Nonmem*

Description

This function can be used to define the pmx configuration used in plots. e.g. Monolox/Nonmem

Usage

```
pmx_config(sys = "mlx", inputs, plots, ...)
```

Arguments

sys	charcarter system used , monolix,nonmem,...
inputs	charcater path to the inputs settings file (yaml format)
plots	charcater path to the inputs settings file (yaml format)
...	extra arguments not used

Details

To create a controller user can create a pmxConfig object using
- either an input template file
- or a plot template file
- or both.

By default the 'standing' configuration will be used.

Value

pmxConfig object

Examples

```
# *************** Create a controller using custom plot configuration ***************** ------

library(ggPMX)
theophylline <- file.path(
  system.file(package = "ggPMX"), "testdata",
  "theophylline"
)
WORK_DIR <- file.path(theophylline, "Monolix")
input_file <- file.path(theophylline, "data_pk.csv")


# create a controller with a custom plots template
ctr <-  pmx_mlx(
  config = pmx_config(
    plots=file.path( system.file(package = "ggPMX"),"examples/plots.yaml"),
    inputs = system.file(package = "ggPMX","examples/custom_inputs.yaml")
  ),
  directory = WORK_DIR,
  input = input_file,
  dv = "Y",
  dvid = "DVID",
  cats = c("SEX"),
  conts = c("WT0", "AGE0"),
  strats = "STUD"
)

## get the list of plots
ctr %>% plots
ctr %>% get_plot("custom_res_time")
ctr %>% get_plot("custom_npde_time")
```

pmx_copy *Creates a deep copy of the controller*

Description

Creates a deep copy of the controller

Usage

```
pmx_copy(ctr, keep_globals = FALSE, ...)
```

Arguments

ctr	pmxClass object
keep_globals	logical if TRUE we keep the global parameters changed by pmx_settings
...	extra parameters passed to pmx_settings

Details

The controller is an 'R6' object, it behaves like a reference object. Some functions (methods) can have a side effect on the controller and modify it internally. Technically speaking we talk about chaining not piping here. However , using pmx_copy user can work on a copy of the controller.

By defaul the copy don't keep global parameters setted using pmx_settings.

Value

an object of pmxClass

Examples

```
ctr <- theophylline()
cctr <- ctr %>% pmx_copy()
## Any change in the ctr has no side effect in the ctr and vice versa
```

pmx_cov *Select/Map covariates using human labels*

Description

Select/Map covariates using human labels

Usage

```
pmx_cov(values, labels = NULL)
```

Arguments

values	list of covariates to use to create the plot
labels	list of covariates facets labels

Details

In case of 'pmx_plot_eta_cats' and 'pmx_plot_eta_conts' you can customize the covariates and covaraites labels using 'pmx_cov'.

Value

pmxCOVObject object

---

pmx_dens *Creates a density plot object*

Description

Creates a density plot object

Usage

```
pmx_dens(
  x,
  labels,
  dname = NULL,
  xlim = 3,
  var_line = NULL,
  snd_line = NULL,
  vline = NULL,
  ...
)
```

Arguments

x	character variable name to sample
labels	list of texts/titles used within the plot
dname	name of dataset to be used
xlim	numeric x axis limits
var_line	list variable density graphics parameters
snd_line	list normal density graphics parameters
vline	list vertical line graphics parameters
...	others graphics arguments passed to pmx_gpar internal object.

Details

labels is a list that contains:

- **title:** plot title default "IWRES density plot"
- **x:** x axis label default to "Etas"
- **y:** y axis label default to empty

var_line is a list that contains:

- **linetype:** default to 1
- **color:** default to black
- **size:** default to 1

snd_line is a list that contains:

- **linetype:** default to 2
- **color:** default to black
- **size:** default to 1

vline is a list that contains:

- **linetype:** default to 3
- **color:** default to black
- **size:** default to 1

`pmx_endpoint` *Creates pmx endpoint object*

Description

Creates pmx endpoint object

Usage

```
pmx_endpoint(code, label = "", unit = "", file.code = code, trans = NULL)
```

Arguments

`code`	`character` endpoint code : used to filter observations DVID==code.
`label`	`character` endpoint label: used to set title and axis labels
`unit`	`character` endpoint unit : used to set title and axis labels
`file.code`	`character` endpoint file code : used to set predictions and finegrid files extensions in case using code parameter is not enough.
`trans`	`list` Transformation parameter not used yet.

Details

In case of multiple endpoints, pkpd case for example, we need to pass endpoint to the pmx call. Internally , ggPMX will filter the observations data set to keep only rows satisfying `DVID==code`. The code is also used to find the right predictions and or fingrid files. ggPMX use the configuration file to fine the path of the predictions file (like the single endpoint case) and then filter the right file using the code parameter.
For example:

- predictions{code}.txt for mlx16
- predictions{code}.txt and y{code}_residual for mlx18

For some tricky examples the code parameter is not enough to find the files. In that case the `file.code` parameter is used to distinguish the endpoint files.

Examples

```
## Use file.code parameter
pk_pd_path <- file.path(system.file(package = "ggPMX"), "testdata","pk_pd")

WORK_DIR <- file.path(pk_pd_path, "RESULTS")

ep <- pmx_endpoint(
  code="4",
  file.code="2"
)

input_file <- file.path(pk_pd_path, "pk_pd.csv")

ctr <- pmx_mlx(
  config = "standing",
  directory = WORK_DIR,
  input = input_file,
  dv = "dv",
  dvid = "dvid",
  cats = "sex",
  conts = "wt",
  endpoint = ep
 )

## using mlxtran

ep <- pmx_endpoint(
  code="3",
 file.code="1"
)

mlxtran_file <- file.path(pk_pd_path, "pk_pd.mlxtran")
ctr <- pmx_mlxtran(mlxtran_file,endpoint=ep)
```

pmx_filter *filter data in a pmx controller*

Description

filter data in a pmx controller

Usage

```
pmx_filter(
  ctr,
  data_set = c("estimates", "predictions", "eta", "finegrid", "shrink", "input",
    "individual"),
  pmx_exp
)
```

Arguments

ctr	A controller. An object of 'pmxClass'
data_set	A data_set within the controller to apply a filter to.
pmx_exp	A filter expression

Value

Returns a pmx controller with a filtered data set.

Examples

```
## example of global filter
ctr <- theophylline()
ctr %>% pmx_filter(data_set = "prediction", ID == 5 & TIME < 2)
ctr %>% get_data("prediction")
```

pmx_get_configs *Get List of built-in configurations*

Description

Get List of built-in configurations

Usage

```
pmx_get_configs(sys = "mlx")
```

Arguments

sys	can be mlx, by default all configurations will be listed

Value

names of the config

Examples

```
pmx_get_configs()
```

---

pmx_gpar *Handling pmx Graphical parameters*

Description

Handling pmx Graphical parameters

Usage

```
pmx_gpar(
  labels,
  axis.title,
  axis.text,
  ranges,
  is.smooth,
  smooth,
  is.band,
  band,
  is.draft,
  draft,
  discrete,
  is.identity_line,
  identity_line,
  scale_x_log10,
  scale_y_log10,
  color.scales,
  is.legend,
  legend.position
)
```

Arguments

labels	list of labels, like title, subtitle, x , y
axis.title	list or element_text (same as ggplot2 axis.title theme)
axis.text	list or element_text (same as ggplot2 axis.text theme)

ranges	limits of x/y ranges
is.smooth	logical if set to TRUE add smooth layer
smooth	smooth layer parameters
is.band	logical if TRUE add horizontal band
band	horizontal band parameters
is.draft	logical if TRUE add draft layer
draft	draft layer parameters
discrete	logical if TRUE x axis is discrete(FALSE by default)
is.identity_line	logical if TRUE add y=x line
identity_line	list y=x aes properties
scale_x_log10	logical if TRUE add scale_x_log10 layer
scale_y_log10	logical if TRUE add scale_y_log10 layer
color.scales	list define scales parameter in case of strat.color pmx_settings
is.legend	logical if TRUE x axis is discrete(FALSE by default)
legend.position	charcater legend position it takes the same value as the equivalent ggplot2 parameter

Details

This object contains all general graphic settings. It used internally by all pmx_plot(generic function) to set the default behavior.

Value

An object of class "pmx_gpar".

---

pmx_nlmixr *Creates pmx controller from an nlimxr fit object*

Description

Creates pmx controller from an nlimxr fit object

Usage

```
pmx_nlmixr(fit, dvid, conts, cats, strats, endpoint, settings, vpc = TRUE)
```

Arguments

fit	nlmixr object
dvid	*[Optional]* character observation type parameter.
conts	*[Optional]*character vector of continuous covariates
cats	*[Optional]*character vector of categorical covariates
strats	*[Optional]*character extra stratification variables
endpoint	pmxEndpointClass or integer or charcater defalut to NULL of the endpoint code. pmx_endpoint
settings	*[Optional]*pmxSettingsClass pmx_settings
vpc	*[Optional]* logical a boolean indiacting if vpc should be calculated (by default TRUE)

Value

pmxClass controller object.

---

pmx_plot *Generic pmx plot*

Description

Generic pmx plot

Usage

```
pmx_plot(ctr, pname, ...)
```

Arguments

ctr	pmxClass pmx controller
pname	plot name
...	others graphics parameters passed :

- pmx_gpar internal function to customize shared graphical parameters
- pmx_qq quantile-quantile plot object
- pmx_update function.

pmx_plot_cats *Generic pmx stratified plot*

Description

Generic pmx stratified plot

Usage

```
pmx_plot_cats(ctr, pname, cats, chunk = "", print = TRUE, ...)
```

Arguments

ctr	pmxClass pmx controller
pname	plot name
cats	list of categorical variables. By default all of them
chunk	chunk name
print	logical if TRUE print plots otherwise the list of plots is returned
...	others graphics parameters passed : • pmx_gpar internal function to customize shared graphical parameters • pmx_qq quantile-quantile plot object • pmx_update function.

pmx_plot_eta_matrix *Eta matrix plot*

Description

Eta matrix plot

Usage

```
pmx_plot_eta_matrix(
  ctr,
  title,
  dname,
  type.eta,
  text_color,
  is.shrink,
  shrink,
  point,
  is.smooth,
  smooth,
```

```
  is.hline,
  hline,
  filter,
  strat.facet,
  facets,
  strat.color,
  trans,
  pmxgpar,
  labels,
  axis.title,
  axis.text,
  ranges,
  is.band,
  band,
  is.draft,
  draft,
  is.identity_line,
  identity_line,
  scale_x_log10,
  scale_y_log10,
  color.scales,
  ...
)
```

Arguments

ctr	pmx controller
title	character the plot title
dname	name of dataset to be used
type.eta	character type of eat can be 'mode' or 'mean'.'mode' by default
text_color	color of the correlation text in the upper matrix
is.shrink	logical if TRUE add shrinkage to the plot
shrink	list shrinkage graphical parameter
point	list geom_point graphical parameter
is.smooth	logical if TRUE add smoothing to lower matrix plots
smooth	list geom_smooth graphical parameters
is.hline	logical if TRUE add horizontal line to lower matrix plots
hline	list geom_hline graphical parameters
	pmx_update parameters
filter	expression filter which will be applied to plotting data.
strat.facet	formula optional stratification parameter by facetting. This split plot by strats(each strat in a facet)
facets	list facet_wrap parameters.
strat.color	character optional stratification parameter by grouping. This will split the plot by group (color) of strat.

trans	character define the transformation to apply on x or y or both variables
pmxgpar	a object of class pmx_gpar possibly the output of the
	pmx_gpar: Shared basic graphics parameters
labels	list list containing plot and/or axis labels: title, subtitle, x , y
axis.title	list containing element_text attributes to customize the axis title. (similar to ggplot2 axis.title theme)
axis.text	list containing element_text attributes to customize the axis text (similar to ggplot2 axis.text theme)
ranges	list limits of x/y ranges
is.band	logical if TRUE add horizontal band
band	list horizontal band parameters. geom_hline graphical parameters.
is.draft	logical if TRUE add draft layer
draft	list draft layer parameters. geom_text graphical parameters.
is.identity_line	logical if TRUE add an identity line
identity_line	listgeom_abline graphical parameters.
scale_x_log10	logical if TRUE use log10 scale for x axis.
scale_y_log10	logical if TRUE use log10 scale for y axis.
color.scales	list define scales parameter in case of strat.color pmx_settings
...	others graphics parameters passed :
	• pmx_gpar internal function to customize shared graphical parameters
	• eta_pairs ggPMX internal function for eta matrix plot.
	• pmx_update function.
	eta_pairs parameters

Value

ggplot2 object

Examples

```
# basic use  ---------------------------------------

ctr <- theophylline()
p <- ctr %>% pmx_plot_eta_matrix

# update graphical parameter  ----------------------

## update labels
ctr %>% pmx_plot_eta_matrix(
  labels = list(title = "Eta matrix new title")
```

```
)

## remove draft
ctr %>% pmx_plot_eta_matrix(is.draft = FALSE)

## change text color line
ctr %>% pmx_plot_eta_matrix(
  text_color="red",
  shrink=list(mapping=aes(color="magenta"))
  )

## custom point aes and static parameters
## we can customize any geom_point parameter
ctr %>% pmx_plot_eta_matrix(
  point = list(color = "blue", shape = 4)
)



# stratification  ---------------------------------

## IGNORE continuous stratification
ctr %>% pmx_plot_eta_matrix(strat.color = "WT0")
## IGNORE  categorical stratification
ctr %>% pmx_plot_eta_matrix(strat.facet = "SEX")

# subsetting  ---------------------------------------

## we can use any expression involving the data
ctr %>% pmx_plot_eta_matrix(filter = EFFECT%in%  c("Cl","ka"))
```

`pmx_plot_individual` *Individual plot*

Description

Individual plot

Usage

```
pmx_plot_individual(
  ctr,
  npage = 1,
  print = FALSE,
  dname,
  pred_line,
  ipred_line,
  point,
```

```
  is.legend,
  use.finegrid,
  bloq,
  filter,
  strat.facet,
  facets,
  strat.color,
  trans,
  pmxgpar,
  labels,
  axis.title,
  axis.text,
  ranges,
  is.smooth,
  smooth,
  is.band,
  band,
  is.draft,
  draft,
  is.identity_line,
  identity_line,
  scale_x_log10,
  scale_y_log10,
  color.scales,
  ...
)
```

Arguments

ctr	pmx controller
npage	integer page(s) to display , set npage to NULL
print	logical if TRUE the ouptut will be a print not a ggplot2. This is useful for rmarkdwon output to avoid verbose list index print.
dname	character name of dataset to be used. User can create his own dataset using set_data and pass it as dname to be plotted.
pred_line	list some ipred line geom properties aesthetics
ipred_line	list some pred line geom properties aesthetics
point	list some point geom properties aesthetics
is.legend	logical if TRUE add a legend
use.finegrid	logical if FALSE use predictions data set
bloq	pmxBLOQ object created by pmx_bloq.
	pmx_update parameters
filter	expression filter which will be applied to plotting data.
strat.facet	formula optional stratification parameter by facetting. This split plot by strats(each strat in a facet)

facets	list facet_wrap parameters.
strat.color	character optional stratification parameter by grouping. This will split the plot by group (color) of strat.
trans	character define the transformation to apply on x or y or both variables
pmxgpar	a object of class pmx_gpar possibly the output of the
	pmx_gpar: Shared basic graphics parameters
labels	list list containing plot and/or axis labels: title, subtitle, x , y
axis.title	list containing element_text attributes to customize the axis title. (similar to ggplot2 axis.title theme)
axis.text	list containing element_text attributes to customize the axis text (similar to ggplot2 axis.text theme)
ranges	list limits of x/y ranges
is.smooth	logical if set to TRUE add smooth layer
smooth	list geom_smooth graphical/smoothing fun parameters
is.band	logical if TRUE add horizontal band
band	list horizontal band parameters. geom_hline graphical parameters.
is.draft	logical if TRUE add draft layer
draft	list draft layer parameters. geom_text graphical parameters.
is.identity_line	logical if TRUE add an identity line
identity_line	listgeom_abline graphical parameters.
scale_x_log10	logical if TRUE use log10 scale for x axis.
scale_y_log10	logical if TRUE use log10 scale for y axis.
color.scales	list define scales parameter in case of strat.color pmx_settings
...	others graphics parameters passed : • pmx_gpar internal function to customize shared graphical parameters • individual generic object for individual plots. • pmx_update function.
	individual parameters

Value

ggplot2 or list of ggplot2 objects

Examples

```
# basic use  ---------------------------------------

ctr <- theophylline()
ctr %>% pmx_plot_individual(npage = 1)
## multiple pages
ctr %>% pmx_plot_individual(npage = c(1, 3))
## change faceting
```

```
ctr %>% pmx_plot_individual(facets = list(nrow = 5, ncol = 5), npage = 2)


# update graphical parameter  ----------------------

## update labels
ctr %>% pmx_plot_individual(
  labels = list(title = "Custom individual plot")
)

## remove draft
ctr %>% pmx_plot_individual(is.draft = FALSE)

## Customize ipred_line with any geom_line parameter
ctr %>% pmx_plot_individual(
  pred_line = list(color = "red", linetype = 20, alpha = 0.5)
)

## Customize ipred_line with any geom_line parameter
ctr %>% pmx_plot_individual(
  ipred_line = list(size = 5)
)


## Customize any geom_point parameter
ctr %>% pmx_plot_individual(
  point = list(aes(alpha = DV), color = "green", shape = 4)
)


## legend

p <- ctr %>% pmx_plot_individual(
  is.legend=TRUE,
  point=list(shape=20),
  pred_line=list(linetype=6)
)

# # stratification  ----------------------------------
#
# ## continuous stratification
ctr %>% pmx_plot_individual(strat.color = "WT0")

# # subsetting  --------------------------------------
#
# ## we can use any expression involving the data
# ## filter and stratify
ctr %>% pmx_plot_individual(
  filter = SEX == 1, strat.facet = ~SEX,
  facets = list(nrow = 5, ncol = 5))

# # transformation  ----------------------------------
#
```

```
# ## apply a log transformation in y
ctr %>% pmx_plot_individual(trans = "log10_y")
# ## apply a custonm trsnformation to normalize axis between 0 and 1

## get a list of parameter
p <- ctr %>% pmx_plot_individual(
  npage=NULL,
  point=list(shape=4,color='blue',size=10),
  facets = list(nrow = 5, ncol = 5),
  labels = list(title = "My individuals",x='my time',y='PD data')
)
```

---

pmx_plot_iwres_dens *IWRES density plot*

Description

IWRES density plot

Usage

```
pmx_plot_iwres_dens(
  ctr,
  dname,
  xlim,
  var_line,
  snd_line,
  vline,
  filter,
  strat.facet,
  facets,
  strat.color,
  trans,
  pmxgpar,
  labels,
  axis.title,
  axis.text,
  ranges,
  is.smooth,
  smooth,
  is.band,
  band,
  is.draft,
  draft,
  is.identity_line,
  identity_line,
```

```
  scale_x_log10,
  scale_y_log10,
  color.scales,
  ...
)
```

Arguments

ctr	pmx controller
dname	character name of dataset to be used. User can create his own dataset using set_data and pass it as dname to be plotted.
xlim	numeric x axis limits
var_line	list variable denstiy graphics parameters
snd_line	list normal denstiy graphics parameters
vline	list vertical line graphics parameters
	pmx_update parameters
filter	expression filter which will be applied to plotting data.
strat.facet	formula optional stratification parameter by facetting. This split plot by strats(each strat in a facet)
facets	list facet_wrap parameters.
strat.color	character optional stratification parameter by grouping. This will split the plot by group (color) of strat.
trans	character define the transformation to apply on x or y or both variables
pmxgpar	a object of class pmx_gpar possibly the output of the
	pmx_gpar: Shared basic graphics parameters
labels	list list containing plot and/or axis labels: title, subtitle, x , y
axis.title	list containing element_text attributes to customize the axis title. (similar to ggplot2 axis.title theme)
axis.text	list containing element_text attributes to customize the axis text (similar to ggplot2 axis.text theme)
ranges	list limits of x/y ranges
is.smooth	logical if set to TRUE add smooth layer
smooth	list geom_smooth graphical/smoothing fun parameters
is.band	logical if TRUE add horizontal band
band	list horizontal band parameters. geom_hline graphical parameters.
is.draft	logical if TRUE add draft layer
draft	list draft layer parameters. geom_text graphical parameters.
is.identity_line	logical if TRUE add an identity line
identity_line	listgeom_abline graphical parameters.
scale_x_log10	logical if TRUE use log10 scale for x axis.

scale_y_log10	logical if TRUE use log10 scale for y axis.
color.scales	list define scales parameter in case of strat.color pmx_settings
...	others graphics parameters passed :

- pmx_gpar internal function to customize shared graphical parameters
- pmx_dens pmx density object.
- pmx_update function.

pmx_dens parameters

Value

ggplot2 or list of ggplot2 objects

---

pmx_plot_vpc *VPC plot*

Description

VPC plot

Usage

```
pmx_plot_vpc(
  ctr,
  type,
  idv,
  obs,
  pi,
  ci,
  rug,
  bin,
  is.legend,
  dname,
  filter,
  strat.facet,
  facets,
  strat.color,
  trans,
  pmxgpar,
  labels,
  axis.title,
  axis.text,
  ranges,
  is.smooth,
  smooth,
  is.band,
  band,
```

```
  is.draft,
  draft,
  is.identity_line,
  identity_line,
  scale_x_log10,
  scale_y_log10,
  color.scales,
  is.footnote,
  ...
)
```

Arguments

ctr	pmx controller
type	charcater can be either percentile or scatter
idv	chracater individual variable
obs	pmx_vpc_obs object observation layer pmx_vpc_obs
pi	pmx_vpc_pi object percentile layer pmx_vpc_pi
ci	pmx_vpc_ci object confidence interval layer pmx_vpc_ci
rug	pmx_vpc_rug object rug layer pmx_vpc_rug
bin	pmx_vpc_bin object pmx_vpc_bin
is.legend	logical if TRUE add legend
dname	added for compatibility with other ggPMX plots
	pmx_update parameters
filter	expression filter which will be applied to plotting data.
strat.facet	formula optional stratification parameter by facetting. This split plot by strats(each strat in a facet)
facets	list facet_wrap parameters.
strat.color	character optional stratification parameter by grouping. This will split the plot by group (color) of strat.
trans	character define the transformation to apply on x or y or both variables
pmxgpar	a object of class pmx_gpar possibly the output of the
	pmx_gpar: Shared basic graphics parameters
labels	list list containing plot and/or axis labels: title, subtitle, x , y
axis.title	list containing element_text attributes to customize the axis title. (similar to ggplot2 axis.title theme)
axis.text	list containing element_text attributes to customize the axis text (similar to ggplot2 axis.text theme)
ranges	list limits of x/y ranges
is.smooth	logical if set to TRUE add smooth layer
smooth	list geom_smooth graphical/smoothing fun parameters
is.band	logical if TRUE add horizontal band

band	list horizontal band parameters. geom_hline graphical parameters.
is.draft	logical if TRUE add draft layer
draft	list draft layer parameters. geom_text graphical parameters.
is.identity_line	logical if TRUE add an identity line
identity_line	listgeom_abline graphical parameters.
scale_x_log10	logical if TRUE use log10 scale for x axis.
scale_y_log10	logical if TRUE use log10 scale for y axis.
color.scales	list define scales parameter in case of strat.color pmx_settings
is.footnote	logical if TRUE add footnote
...	others graphics parameters passed :

- pmx_gpar internal function to customize shared graphical parameters
- pmx_vpc pmx vpc object.
- pmx_update function.

pmx_vpc parameters

Details

You can use pmx_vpc_bin to set the bin parameters. In case of stratification, binning can be different for each strat level (case within_strat equal to FALSE).

Value

ggplot2 or list of ggplot2 objects

See Also

Other vpc: pmx_vpc_bin(), pmx_vpc_ci(), pmx_vpc_obs(), pmx_vpc_pi(), pmx_vpc_rug(), pmx_vpc()

Examples

```
library(ggPMX)

theo_path <- file.path(
  system.file(package = "ggPMX"), "testdata",
  "theophylline"
)
WORK_DIR <- file.path(theo_path, "Monolix")
input_file <- file.path(theo_path, "data_pk.csv")
vpc_file <- file.path(theo_path, "sim.csv")

ctr <- pmx_mlx(
  config = "standing",
  directory = WORK_DIR,
  input = input_file,
```

```
  dv = "Y",
  dvid = "dvid",
  cats = c("SEX"),
  conts = c("WT0", "AGE0"),
  strats = "STUD",
  settings = pmx_settings(
    use.labels=TRUE,
    cats.labels=list(
      SEX=c("0"="Male","1"="Female")
    )
  ),
  sim = pmx_sim(
    file = vpc_file,
    irun ="rep",
    idv="TIME"
  )
)


ctr %>% pmx_plot_vpc(
  strat.facet="SEX",
  facets=list(nrow=2),
  type="percentile",
  is.draft = FALSE,
  pi = pmx_vpc_pi(interval = c(0.1,0.9),
              median=list(color="green"),
              extreme= list(color="green")),
  obs = pmx_vpc_obs(color="blue",shape=18,size=2),
  ci = pmx_vpc_ci(interval = c(0.1,0.9),
              median=list(fill="pink")),
  bin=pmx_vpc_bin("kmeans",n=5)
)

ctr %>%
  pmx_plot_vpc(bin= pmx_vpc_bin(
     style = "fixed",
     fixedBreaks=c(-10,2, 5, 10,15,50))
  )

# example with legend

ctr %>% pmx_plot_vpc(
  is.legend = TRUE,
  pi = pmx_vpc_pi(interval=c(0.02,0.98),median = list(linetype="dotted")),
  ci = pmx_vpc_ci(interval = c(0.05,0.95),median=list(fill="red"))
)
```

---

pmx_qq *This function creates a qq plot object*

Description

This function creates a qq plot object

Usage

```
pmx_qq(
  x,
  labels,
  dname = NULL,
  point = NULL,
  xmax = TRUE,
  facets = NULL,
  is.reference_line = NULL,
  reference_line = NULL,
  is.shrink = NULL,
  shrink = NULL,
  is.hline = NULL,
  hline = NULL,
  ...
)
```

Arguments

x	character variable name to sample
labels	list of texts/titles used within the plot
dname	name of dataset to be used
point	list geom_point attributes color, shape,...
xmax	logical if FALSE do not use max(aes(x)) as limits default to TRUE
facets	list
is.reference_line	logical if TRUE add reference line to the plot
reference_line	list geom_line attributes. Used only for pmx_plot_eta_qq
is.shrink	logical if TRUE add shrinkage to the plot
shrink	list shrinkage graphical parameter
is.hline	logical if TRUE add horizontal line y=0 (TRUE by default)
hline	geom hline graphical parameters
...	others graphics arguments passed to pmx_gpar internal object.

Details

labels is a list that contains:

- **title:** plot title default "EBE vs. covariates"
- **x:** x axis label default to "Etas"
- **y:** y axis label default to empty

point is a list that contains:

- **shape:** default to 1
- **color:** default to black
- **size:** default to 1

Value

pmx_qq object

---

pmx_qq_plot *Quantile-quantile plots*

Description

Quantile-quantile plots

Quantile-quantile plot of IWRES

Quantile-quantile plot of eta variables

Quantile-quantile plot of NPDE

Usage

```
pmx_qq_plot(
  dname,
  point,
  is.reference_line,
  reference_line,
  is.shrink,
  shrink,
  is.hline,
  hline,
  filter,
  strat.facet,
  facets,
  strat.color,
  trans,
  pmxgpar,
  labels,
  axis.title,
  axis.text,
  ranges,
  is.smooth,
  smooth,
  is.band,
  band,
  is.draft,
```

```
  draft,
  is.identity_line,
  identity_line,
  scale_x_log10,
  scale_y_log10,
  color.scales,
  ...
)

pmx_plot_iwres_qq(ctr, ...)

pmx_plot_eta_qq(ctr, ...)

pmx_plot_npde_qq(ctr, ...)
```

Arguments

dname	name of dataset to be used
point	list geom_point parameters.
is.reference_line	logical if TRUE add reference line to the plot
reference_line	list geom_abline parameters.
is.shrink	logical if TRUE add shrinkage to the plot
shrink	list shrinkage graphical parameter (geom_text)
is.hline	logical if TRUE add horizontal line y=0 (TRUE by default)
hline	list geom hline graphical parameters
	pmx_update parameters
filter	expression filter which will be applied to plotting data.
strat.facet	formula optional stratification parameter by facetting. This split plot by strats(each strat in a facet)
facets	list facet_wrap parameters.
strat.color	character optional stratification parameter by grouping. This will split the plot by group (color) of strat.
trans	character define the transformation to apply on x or y or both variables
pmxgpar	an object of class pmx_gpar
labels	list list containing plot and/or axis labels: title, subtitle, x , y
axis.title	list containing element_text attributes to customize the axis title. (similar to ggplot2 axis.title theme)
axis.text	list containing element_text attributes to customize the axis text (similar to ggplot2 axis.text theme)
ranges	list limits of x/y ranges
is.smooth	logical if set to TRUE add smooth layer
smooth	list geom_smooth graphical/smoothing fun parameters

is.band	logical if TRUE add horizontal band
band	list horizontal band parameters. geom_hline graphical parameters.
is.draft	logical if TRUE add draft layer
draft	list draft layer parameters. geom_text graphical parameters.
is.identity_line	logical if TRUE add an identity line
identity_line	listgeom_abline graphical parameters.
scale_x_log10	logical if TRUE use log10 scale for x axis.
scale_y_log10	logical if TRUE use log10 scale for y axis.
color.scales	list define scales parameter in case of strat.color pmx_settings
...	others graphics parameters passed : • pmx_gpar internal function to customize shared graphical parameters • pmx_qq quantile-quantile plot object. • pmx_update function.
	pmx_qq parameters
ctr	pmx controller

Value

ggplot2 object

Examples

```
# *************** basic use ***************** ------

ctr <- theophylline()
ctr %>% pmx_plot_eta_qq
ctr %>% pmx_plot_npde_qq
ctr %>% pmx_plot_iwres_qq

# update graphical parameter  ----------------------

## add reference line
ctr %>% pmx_plot_npde_qq(reference_line=list(color="blue"))

## remove reference line
ctr %>% pmx_plot_eta_qq(reference_line=NULL)


# stratification  ---------------------------------

## categorical stratification color parameter
ctr %>% pmx_plot_iwres_qq(strat.facet="STUD",strat.color="SEX")
## categorical stratification facetting
ctr %>% pmx_plot_eta_qq(strat.facet = "SEX")
```

```
## do not use symmetric axis
ctr %>% pmx_plot_npde_qq(xmax=FALSE,reference_line=list())
```

`pmx_register_plot` *Register plot*

Description

Register plot

Usage

```
pmx_register_plot(ctr, pp, pname = NULL)
```

Arguments

`ctr`	`pmxClass` controller
`pp`	`ggplot2` plot
`pname`	`character` plot nme

`pmx_report` *Generates ggpmX report from a pre-defined template*

Description

Generates ggpmX report from a pre-defined template

Usage

```
pmx_report(
  contr,
  name,
  save_dir,
  format = c("both", "plots", "report"),
  template = "standing",
  footnote = format == "both",
  edit = FALSE,
  extension = NULL,
  title,
  ...
)
```

Arguments

contr	pmxClass controller
name	character The report name
save_dir	Output directory. A directory to write the results files to
format	character the result type, can be a standalone directory of plots or a report document as defined in the template (pdf, docx,..) ,or both
template	character ggPMX predefined template or the path to a custom rmarkdwon template. Use pmx_report_template to get the list of available templates
footnote	logical TRUE to add a footnote to the generated plots. The default footnote is to add the path where the plot is saved.
edit	logical TRUE to edit the template immediately
extension	character The output document format. By default, a word report is generated. User can specify one or more formats from c("word","pdf","html","all"). extnestion "all" to generate all formats.
title	character report title (optional)
...	extra parameters depending in the template used

Details

pmx_report uses pre-defined template .Rmd to generate the report. The idea is to pass the controller as a report argument using knitr params artifact.

Examples

```
library(ggPMX)
## list of templates
## ctr %>% pmx_report_template()

report_dir <- tempdir()
## case1: generate a single report
## We use default save dir,
ctr <- theophylline()
ctr %>% pmx_report(
  name = "my_report",
  save_dir = report_dir,
  format="report")

## case2: generate standalone plots
## Note here the use of a custom dir to save results
ctr <- theophylline()
ctr %>% pmx_report(
  name = "my_report",
```

```
  save_dir = report_dir,
  format="plots")


## case3: generate both : reports + plots
## by default add footnote
## Note , you can force footnote to FALSE using footnote parameter
ctr <- theophylline()
ctr %>% pmx_report(
  name = "my_report",
  save_dir = report_dir,
  format="both")

## case4 : generate standalone plots with footnotes
ctr <- theophylline()
ctr %>% pmx_report(
  name = "my_report",
  save_dir = report_dir,
  footnote=TRUE,
  format="plots")




##  case6: dynamic edit
## uncomment to run
# ctr <- theophylline()
# ctr %>% pmx_report(
#   save_dir = file.path(report_dir,"case6"),
#   name = "my_report",
#   format="report",
#   edit = TRUE)


## case7 : use custom template file

ctr <- theophylline()
custom_template <-
  file.path( system.file(package = "ggPMX"),"examples","templates","custom_report.Rmd")
ctr %>% pmx_report(
  name="report2",
  save_dir = report_dir,
  template=custom_template,
  format="both"
)

## case7 : generate individual plots report

ctr <- theophylline()
ctr %>% pmx_report(
  name="report2",
  save_dir = report_dir,
```

```
  template="individual",
  format="both",
  npage=1:2
)

## case8: misc example with complicated features
## see github issue : #179
ctr <- theophylline()
misc_template <-
  file.path( system.file(package = "ggPMX"),"examples","templates","misc.Rmd")
ctr %>% pmx_report(
  name="misc",
  save_dir = report_dir,
  template=misc_template,
  format="both"
)
```

`pmx_report_template` *Gets build-in report templates*

Description

Gets build-in report templates

Usage

```
pmx_report_template()
```

Value

list of templates names

Examples

```
pmx_report_template()
```

`pmx_settings` *Create controller global settings*

Description

Create controller global settings

Usage

```
pmx_settings(
  is.draft = TRUE,
  use.abbrev = FALSE,
  color.scales = NULL,
  cats.labels = NULL,
  use.labels = FALSE,
  use.titles = TRUE,
  effects = NULL,
  ...
)
```

Arguments

is.draft	logical if FALSE any plot is without draft annotation
use.abbrev	logical if TRUE use abbreviations mapping for axis names
color.scales	list list containing elements of scale_color_manual
cats.labels	list list of named vectors for each factor
use.labels	logical if TRUE replace factor named by cats.labels
use.titles	logical FALSE to generate plots without titles
effects	list list of effects levels and labels
...	extra parameter not used yet

Value

pmxSettingsClass object

Examples

```
library(ggPMX)
library(ggplot2)
ctr <- theophylline(
  settings=
    pmx_settings(
      color.scales=list(
        "Study",
        labels=c("Study 1","Study 2"),
        values=c("1"="lightyellow","2"="lightblue")),
      cats.labels=list(
        SEX=c("0"="M","1"="F"),
        STUD=c("1"="Study 1","2"="Study 2")
      ),
      use.abbrev=TRUE,
      is.draft=TRUE,
      use.labels=TRUE
    )
```

```
)



ctr %>%
  pmx_plot_npde_time(strat.color="STUD",strat.facet=~SEX)
#
#
ctr  %>%
  pmx_plot_eta_box(strat.color="STUD", strat.facet =~SEX)

ctr  %>%  pmx_plot_eta_hist
```

`pmx_sim` *Create simulation object*

Description

Create simulation object

Usage

```
pmx_sim(file, data, irun, idv)
```

Arguments

`file`	`character` path to the simulation file
`data`	`data.table` simulation data
`irun`	`character` name of the simulation column
`idv`	`character` name of the ind. variable

Examples

```
library(ggPMX)

theo_path <- file.path(
  system.file(package = "ggPMX"), "testdata",
  "theophylline"
)
WORK_DIR <- file.path(theo_path, "Monolix")
input_file <- file.path(theo_path, "data_pk.csv")
vpc_file <- file.path(theo_path, "sim.csv")

ctr <- pmx_mlx(
  config = "standing",
  directory = WORK_DIR,
  input = input_file,
```

```
  dv = "Y",
  dvid = "dvid",
  cats = c("SEX"),
  conts = c("WT0", "AGE0"),
  strats = "STUD",
  settings = pmx_settings(
    use.labels=TRUE,
    cats.labels=list(
      SEX=c("0"="Male","1"="Female")
    )
  ),
  sim = pmx_sim(
    file = vpc_file,
    irun ="rep",
    idv="TIME"
  )
)


ctr %>% pmx_plot_vpc(
  strat.facet="SEX",
  facets=list(nrow=2),
  type="percentile",
  is.draft = FALSE,
  pi = pmx_vpc_pi(interval = c(0.1,0.9),
              median=list(color="green"),
              extreme= list(color="green")),
  obs = pmx_vpc_obs(color="blue",shape=18,size=2),
  ci = pmx_vpc_ci(interval = c(0.1,0.9),
              median=list(fill="pink")),
  bin=pmx_vpc_bin("kmeans",n=5)
)

ctr %>%
  pmx_plot_vpc(bin= pmx_vpc_bin(
     style = "fixed",
     fixedBreaks=c(-10,2, 5, 10,15,50))
  )

# example with legend

ctr %>% pmx_plot_vpc(
  is.legend = TRUE,
  pi = pmx_vpc_pi(interval=c(0.02,0.98),median = list(linetype="dotted")),
  ci = pmx_vpc_ci(interval = c(0.05,0.95),median=list(fill="red"))
)
```

---

pmx_theme *Define ggPMX theme*

Description

This theme is a simple wrapper gdoc theme from ggthems package.

Usage

```
pmx_theme(...)
```

Arguments

...	can contain any valid argument of ggplot2 theme object.

`pmx_update` *Update plot object*

Description

Update plot object

Usage

```
pmx_update(
  ctr,
  pname,
  strat.color = NULL,
  strat.facet = NULL,
  color.scales = NULL,
  filter = NULL,
  trans = NULL,
  ...,
  pmxgpar = NULL
)
```

Arguments

ctr	pmxClass controller object
pname	character the plot name to update
strat.color	character optional stratification parameter
strat.facet	formula optional stratification parameter
color.scales	list can be used with strat.color to set scale_color_manual pmx_gpar function.
filter	optional filter which will be applied to plotting data
trans	character define the transformation to apply on x or y or both variables
...	others graphical parameters given to set the plot
pmxgpar	a object of class pmx_gpar possibly the output of the

Details

trans is a transformation that user can apply to x, or y coordinates. The transformation is applied to the data before the plotting. This gives more felxiblilty to the user and also conserves all static positions like annotations (draft specially)

For example:

var_x apply variance to x coordinates the variance function

var_xy apply variance to both This mechanism is applied internally to scale log.

Value

controller object with the plot updated

See Also

Other pmxclass: `get_cats()`, `get_conts()`, `get_covariates()`, `get_data()`, `get_occ()`, `get_plot_config()`, `get_plot()`, `get_strats()`, `plot_names()`, `plots()`, `set_data()`, `set_plot()`

---

pmx_vpc *Creates vpc object*

Description

Creates vpc object

Usage

```
pmx_vpc(
  type = c("percentile", "scatter"),
  idv = "TIME",
  obs = pmx_vpc_obs(),
  pi = pmx_vpc_pi(),
  ci = pmx_vpc_ci(),
  rug = pmx_vpc_rug(),
  bin = pmx_vpc_bin(),
  labels = NULL,
  facets = NULL,
  is.legend = TRUE,
  is.footnote = TRUE,
  dname = NULL,
  ...
)
```

Arguments

type	charcater can be either percentile or scatter
idv	chracater individual variable
obs	pmx_vpc_obs object observation layer pmx_vpc_obs
pi	pmx_vpc_pi object percentile layer pmx_vpc_pi
ci	pmx_vpc_ci object confidence interval layer pmx_vpc_ci
rug	pmx_vpc_rug object rug layer pmx_vpc_rug
bin	pmx_vpc_bin object pmx_vpc_bin
labels	list define title and axis labels
facets	is a list of parameters passed to facet_wrap in case of startification
is.legend	logical if TRUE add legend
is.footnote	logical if TRUE add footnote
dname	added for compatibility with other ggPMX plots
...	extra parameters passed to base graphical parameters

See Also

Other vpc: pmx_plot_vpc(), pmx_vpc_bin(), pmx_vpc_ci(), pmx_vpc_obs(), pmx_vpc_pi(), pmx_vpc_rug()

---

pmx_vpc_bin *Creates vpc bins*

Description

Creates vpc bins

Usage

```
pmx_vpc_bin(style, within_strat = FALSE, ...)
```

Arguments

style	character style chosen on of the: "fixed", "sd", "equal", "pretty", "quantile", "kmeans", "hclust" or "jenks"
within_strat	logical if TRUE compute the bining for each strat level. By default t is false and bining are equal for all stratifications levels.
...	other classInt::classIntervals parameters excpet style and n

Details

This is a warraper to

See Also

Other vpc: pmx_plot_vpc(), pmx_vpc_ci(), pmx_vpc_obs(), pmx_vpc_pi(), pmx_vpc_rug(), pmx_vpc()

---

pmx_vpc_ci *Sets vpc confidence interval layer*

Description

Sets vpc confidence interval layer

Usage

```
pmx_vpc_ci(
  show = c("all", "median"),
  interval = c(0.025, 0.975),
  method = c("ribbon", "rectangle"),
  median = list(fill = "red", alpha = 0.3),
  extreme = list(fill = "#3388cc", alpha = 0.3)
)
```

Arguments

show	charcater how areas are displayed: • **show="all"** areas will be displayed for each of the 3 percentiles. • **show="median"** Show only median area.
interval	numeric quantiles values default to c(.05,.95)
method	charcater which areas are displayed: • **method="ribbon"** areas are ribbons. • **method="rectangle"** ares are horizontal rectangles.
median	list containing: • **fill** character Color of the area representing the CI for the median. Default: "#3388cc". • **alpha** numeric Transparency of the area representing the PI for the median. Default=0.3.
extreme	list containing: • **fill** character Color of the area representing the CI for the extreme percentiles. Default: "#3388cc". • **alpha** numeric Transparency of the area representing the PI for the extreme percentiles. Default=0.3.

See Also

Other vpc: pmx_plot_vpc(), pmx_vpc_bin(), pmx_vpc_obs(), pmx_vpc_pi(), pmx_vpc_rug(), pmx_vpc()

---

pmx_vpc_obs *Sets vpc observation layer*

Description

Sets vpc observation layer

Usage

```
pmx_vpc_obs(show = TRUE, color = "#000000", size = 1, alpha = 0.7, shape = 1)
```

Arguments

show	logical if TRUE show observation points
color	character Color of the observed endpoint values. Default: "#000000".
size	numeric Size of the observed endpoint values. Default: 1.
alpha	numeric Transparency of the observed endpoint values. Default: 0.7.
shape	numeric Shape of the observed endpoint values. Default: 1.

See Also

Other vpc: pmx_plot_vpc(), pmx_vpc_bin(), pmx_vpc_ci(), pmx_vpc_pi(), pmx_vpc_rug(), pmx_vpc()

---

pmx_vpc_pi *Sets vpc percentile layer*

Description

Sets vpc percentile layer

Usage

```
pmx_vpc_pi(
  show = c("all", "median", "area"),
  interval = c(0.05, 0.95),
  median = list(color = "#000000", size = 1, alpha = 0.7, linetype = "solid"),
  extreme = list(color = "#000000", size = 1, alpha = 0.7, linetype = "dashed"),
  area = list(fill = "blue", alpha = 0.1)
)
```

Arguments

show	charcater how lines are displayed: • **show=all** lines will be displayed for each of the 3 percentiles. with a shaded area. • **show=median** Show only median line. • **show=area** Show only median line and the shaded area
interval	numeric quantiles values default to c(.05,.95)
median	list containing: • **color** charcater Color of the median percentile line. Default: "#000000". • **size** numeric Thickness of the median percentile line. Default: 1. • **alpha** numeric Transparency of the median percentile line. Default: 0.7. • **linetype** charcater Linetype of the median percentile line. Default: "solid".
extreme	list containing: • **color** charcater Color of the median percentile line. Default: "#000000". • **size** numeric Thickness of the median percentile line. Default: 1. • **alpha** numeric Transparency of the median percentile line. Default: 0.7. • **linetype** charcater Linetype of the median percentile line. Default: "solid"
area	list containing: • **fill** charcater Color of the shaded area. Default: "blue". • **alpha** numeric Transparency of the sahded area. Default: 0.1.

See Also

Other vpc: pmx_plot_vpc(), pmx_vpc_bin(), pmx_vpc_ci(), pmx_vpc_obs(), pmx_vpc_rug(), pmx_vpc()

---

pmx_vpc_rug *Sets vpc rug layer*

Description

Sets vpc rug layer

Usage

```
pmx_vpc_rug(show = TRUE, color = "#000000", size = 1, alpha = 0.7)
```

Arguments

show	logical If TRUE show bin separators
color	character Color of the rug. Default: "#000000".
size	numeric Thickness of the rug. Default: 1.
alpha	numeric Transparency of the rug. Default: 0.7.

Details

When the vpc confidence interval layer method is rectangles we don't show rug separators.

See Also

Other vpc: pmx_plot_vpc(), pmx_vpc_bin(), pmx_vpc_ci(), pmx_vpc_obs(), pmx_vpc_pi(), pmx_vpc()

---

print.abbreviation *S3 print abbreviation*

Description

S3 print abbreviation

Usage

```
## S3 method for class 'abbreviation'
print(x, ...)
```

Arguments

x	object of class configs
...	pass additional options (not used presently)

Value

print abbreviation

`print.configs` *This function can be used to print configuration of the defined object using S3 method.*

Description

This function can be used to print configuration of the defined object using S3 method.

Usage

```
## S3 method for class 'configs'
print(x, ...)
```

Arguments

`x`	object of class configs
`...`	pass additional options (not used presently)

Value

print result

`print.pmxClass` *Print pmxClass object*

Description

Print pmxClass object

Usage

```
## S3 method for class 'pmxClass'
print(x, ...)
```

Arguments

`x`	pmxClass object
`...`	additinal arguments to pass to print

Value

print object to screen

print.pmxConfig *S3 method print pmxConfig object*

Description

S3 method print pmxConfig object

Usage

```
## S3 method for class 'pmxConfig'
print(x, ...)
```

Arguments

x	pmxConfig object
...	addtional arguments to pass to print (unused currently)

Value

invisible object

print.pmx_gpar *Print pmx_gpar object*

Description

Print pmx_gpar object

Usage

```
## S3 method for class 'pmx_gpar'
print(x, ...)
```

Arguments

x	pmx_gpar object
...	argument passed to print (to satisfy generic)

Value

a character description of graphical parameters

read_input *Read Modelling input data*

Description

Read Modelling input data

Usage

```
read_input(
  ipath,
  dv,
  dvid,
  cats = "",
  conts = "",
  strats = "",
  occ = "",
  endpoint = NULL,
  id = NULL,
  time = NULL
)
```

Arguments

ipath	full path of the input file
dv	character the name of measurable variable used in the input modelling file
dvid	character observation type parameter
cats	*[Optional]*character vector of categorical covariates
conts	*[Optional]*character vector of continuous covariates
strats	*[Optional]*character extra stratification variables
occ	*[Optional]*character inter individual occasion variables
endpoint	integer null in case of a single endpoint otherwise the index of endpoints.
id	character the name of identifier variable used in the input modelling file.
time	character the name of time variable used in the input modelling file

Value

data.table well formatted containing modelling input data

read_mlx_ind_est *Read MONOLIX individual parameters*

Description

Read MONOLIX individual parameters

Usage

```
read_mlx_ind_est(path, x, ...)
```

Arguments

path	character path to the file
x	dataset object
...	extra paramter not used

Value

data.table object

read_mlx_par_est *Read MONOLIX parameter estimation file*

Description

Read MONOLIX parameter estimation file

Usage

```
read_mlx_par_est(path, x, ...)
```

Arguments

path	character path to the file
x	dataset object
...	extra parameter not used

Value

data.table object

read_mlx_pred *Read MONOLIX model predictions*

Description

Read MONOLIX model predictions

Usage

```
read_mlx_pred(path, x, ...)
```

Arguments

path	character path to the file
x	dataset object
...	extra paramter not used

Value

data.table object

residual *This function create a residual for each observed value and also generates a residual distribution*

Description

This function create a residual for each observed value and also generates a residual distribution

Usage

```
residual(
  x,
  y,
  labels = NULL,
  point = NULL,
  is.hline = FALSE,
  hline = NULL,
  dname = NULL,
  facets = NULL,
  bloq = NULL,
  ...
)
```

Arguments

x	x axis aesthetics
y	y axis aesthetics
labels	list that contain title,subtitle, axis labels
point	geom point graphical parameters
is.hline	logical if TRUE add horizontal line y=0 (TRUE by default)
hline	geom hline graphical parameters
dname	name of dataset to be used
facets	list wrap facetting in case of strat.facet
bloq	pmxBLOQ object created by pmx_bloq
...	others graphics arguments passed to pmx_gpar internal object.

Details

Some parameters are a list of parameters :

point is a list that contains:

- **shape:** default to 1
- **color:** default to black
- **size:** default to 1

labels is a list that contains:

- **title:** plot title default to AES_X versue AES_Y
- **subtitle:** plot subtitle default empty
- **x:** x axis label default to AES_X
- **y:** y axis label default to AES_Y

Value

a residual object

See Also

plot_pmx.residual

`residual_scatter` *Scatter residual plots*

Description

Scatter residual plots

DV vs PRED plot

DV vs IPRED plot

IWRES vs IPRED plot

|IWRES| vs IPRED plot

IWRES vs TIME plot

NPDE vs TIME plot

NPDE vs PRED plot

Usage

```
residual_scatter(
  point,
  is.hline,
  hline,
  dname,
  bloq,
  filter,
  strat.facet,
  facets,
  strat.color,
  trans,
  pmxgpar,
  labels,
  axis.title,
  axis.text,
  ranges,
  is.smooth,
  smooth,
  is.band,
  band,
  is.draft,
  draft,
  is.identity_line,
  identity_line,
  scale_x_log10,
  scale_y_log10,
  color.scales,
  ...
```

```
)

pmx_plot_dv_pred(ctr, ...)

pmx_plot_dv_ipred(ctr, ...)

pmx_plot_iwres_ipred(ctr, ...)

pmx_plot_abs_iwres_ipred(ctr, ...)

pmx_plot_iwres_time(ctr, ...)

pmx_plot_npde_time(ctr, ...)

pmx_plot_npde_pred(ctr, ...)
```

Arguments

point	list geom_point graphical parameters.
is.hline	logical if TRUE add horizontal line y=0 (TRUE by default).
hline	list geom_hline graphical parameters.
dname	character name of dataset to be used. User can create his own dataset using set_data and pass it as dname to be plotted.
bloq	pmxBLOQ object created by pmx_bloq.
	pmx_update parameters
filter	expression filter which will be applied to plotting data.
strat.facet	formula optional stratification parameter by facetting. This split plot by strats(each strat in a facet)
facets	list facet_wrap parameters.
strat.color	character optional stratification parameter by grouping. This will split the plot by group (color) of strat.
trans	character define the transformation to apply on x or y or both variables
pmxgpar	a object of class pmx_gpar possibly the output of the
	pmx_gpar: Shared basic graphics parameters
labels	list list containing plot and/or axis labels: title, subtitle, x , y
axis.title	list containing element_text attributes to customize the axis title. (similar to ggplot2 axis.title theme)
axis.text	list containing element_text attributes to customize the axis text (similar to ggplot2 axis.text theme)
ranges	list limits of x/y ranges
is.smooth	logical if set to TRUE add smooth layer
smooth	list geom_smooth graphical/smoothing fun parameters
is.band	logical if TRUE add horizontal band

band	list horizontal band parameters. geom_hline graphical parameters.
is.draft	logical if TRUE add draft layer
draft	list draft layer parameters. geom_text graphical parameters.
is.identity_line	logical if TRUE add an identity line
identity_line	listgeom_abline graphical parameters.
scale_x_log10	logical if TRUE use log10 scale for x axis.
scale_y_log10	logical if TRUE use log10 scale for y axis.
color.scales	list define scales parameter in case of strat.color pmx_settings
...	others graphics parameters passed : • pmx_gpar internal function to customize shared graphical parameters • residual generic object for all residual (scatter) plots . • pmx_update function. **residual parameters**
ctr	pmx controller

Value

ggplot2 object

Examples

```
# NOTES ######################
# examples are availables for all residual plots:
# - pmx_plot_abs_iwres_ipred
# - pmx_plot_dv_ipred
# - pmx_plot_dv_pred
# - pmx_plot_iwres_ipred
# - pmx_plot_iwres_time
# - pmx_plot_npde_time


# basic use  ----------------------------------------

ctr <- theophylline()
p <- ctr %>% pmx_plot_dv_pred()
## p is a ggplot2 object you can add any layer here
p + ggplot2::theme_minimal()

# update graphical parameter  ----------------------

## update labels
ctr %>% pmx_plot_dv_pred(
  labels = list(title = "DV versus PRED new title")
)
```

```
## remove draft
ctr %>% pmx_plot_dv_pred(is.draft = FALSE)

## remove horizontal line
ctr %>% pmx_plot_dv_pred(is.hline = FALSE)

## custom point aes and static parameters
## we can customize any geom_point parameter
ctr %>% pmx_plot_dv_pred(
  point = list(aes(alpha = DV), color = "green", shape = 4)
)



# stratification  ---------------------------------

## continuous stratification
ctr %>% pmx_plot_dv_pred(strat.color = "WT0")
## categorical stratification
ctr %>% pmx_plot_dv_pred(strat.facet = "SEX")
## using formula notation
ctr %>% pmx_plot_dv_pred(strat.facet = STUD~SEX)

# subsetting  -------------------------------------

## we can use any expression involving the data
ctr %>% pmx_plot_dv_pred(filter = DV > mean(DV) & PRED < median(PRED))
## filter and stratify
ctr %>% pmx_plot_dv_pred(filter = SEX == 1, strat.facet = ~SEX)


# transformation  ---------------------------------

## apply a log transformation in y
ctr %>% pmx_plot_dv_pred(trans = "log10_y")
```

`set_abbrev` *update or add a new abbreviation*

Description

update or add a new abbreviation

Usage

```
set_abbrev(ctr, ...)
```

Arguments

ctr	pmxClass controller object
...	Options to set or add, with the form name = value.

Examples

```
ctr <- theophylline()
ctr %>% set_abbrev("new_param" = "new value")
ctr %>% get_abbrev("new_param")
```

---

set_data *Set a controller data set*

Description

Set a controller data set

Usage

```
set_data(ctr, ...)
```

Arguments

ctr	the controller object
...	a named list parameters (see example)

Details

This function can be used to set an existing data set or to create a new one. The basic idea is to change the built-in data set (change the factor level names, change some rows values or apply any other data set operation) and use the new data set using the dname parameter of pmx_plot family functions.

See Also

Other pmxclass: get_cats(), get_conts(), get_covariates(), get_data(), get_occ(), get_plot_config(), get_plot(), get_strats(), plot_names(), plots(), pmx_update(), set_plot()

Examples

```
ctr <- theophylline()
dx <- ctr %>% get_data("eta")
dx <- dx[, EFFECT := factor(
  EFFECT,
  levels = c("ka", "V", "Cl"),
  labels = c("Concentration", "Volume", "Clearance")
)]
## update existing data set
```

```
ctr %>% set_data(eta = dx)
## or create a new data set
ctr %>% set_data(eta_long = dx)
```

`set_plot` *Create a new plot of the desired type*

Description

Create a new plot of the desired type

Usage

```
set_plot(
  ctr,
  ptype = c("IND", "DIS", "SCATTER", "ETA_PAIRS", "ETA_COV", "PMX_QQ", "VPC",
    "PMX_DENS"),
  pname,
  use.defaults = TRUE,
  filter = NULL,
  strat.color = NULL,
  strat.facet = NULL,
  color.scales = NULL,
  trans = NULL,
  ...
)
```

Arguments

`ctr`	`pmxClass` controller object
`ptype`	plot type can be: • "IND" Individual plot type: `individual` • "DIS" Distribution plot type : `distrib` • "SCATTER" Residual plot type :`residual`
`pname`	plot name, if missing it will be created using function aestetics
`use.defaults`	`logical` if FALSE do not use defaults defined in yaml init files
`filter`	optional filter which will be applied to plotting data
`strat.color`	`character`
`strat.facet`	`formula` define categorical stratification as formula
`color.scales`	`list` can be used with strat.color to set scale_color_manual
`trans`	`list`transformation operator
`...`	other plot parameters to configure `pmx_gpar`.

Value

invisible ctr object

See Also

Other pmxclass: get_cats(), get_conts(), get_covariates(), get_data(), get_occ(), get_plot_config(), get_plot(), get_strats(), plot_names(), plots(), pmx_update(), set_data()

---

theophylline *Creates pmx controller using theophylline data*

Description

Creates pmx controller using theophylline data

Usage

```
theophylline(settings = NULL, ...)
```

Arguments

settings	pmxSettings object
...	other parameters of pmx_mlx like endpoint

Value

pmx controller

Examples

```
## Not run:
theophylline()

## End(Not run)
```

wrap_formula *merge facets formula with new formula*

Description

merge facets formula with new formula

Usage

```
wrap_formula(x, origin = "lfacet")
```

Arguments

x	formula object
origin	the origin formula default to ~lfacets

Value

formula object

[.pmx_gpar *Method for subsetting "pmx_gpar" objects*

Description

Method for subsetting "pmx_gpar" objects

Usage

```
## S3 method for class 'pmx_gpar'
x[index, ...]
```

Arguments

x	pmx_gpar object
index	can be character/integer of element
...	other parameter (not used just for generic)

Value

if exists the parameter description

Index