Package ‘mapbayr’

July 27, 2021

Title MAP-Bayesian Estimation of PK Parameters

Version 0.5.0

Description Performs maximum a posteriori Bayesian estimation of individual pharmacokinetic parameters from a model defined in ‘mrgsolve’, typically for model-based therapeutic drug monitoring. Internally computes an objective function value from model and data, performs optimization and returns predictions in a convenient format.

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.1

Imports dplyr, ggplot2, magrittr, mrgsolve (>= 0.10.9), optimx, purrr, rlang, stats, stringr, tibble, tidyr

URL https://github.com/FelicienLL/mapbayr

BugReports https://github.com/FelicienLL/mapbayr/issues

Suggests knitr

NeedsCompilation no

Author Felicien Le Louedec [aut, cre]
   (<https://orcid.org/0000-0003-3699-2729>)

Maintainer Felicien Le Louedec <lelouedec.felicien@iuct-oncopole.fr>

Repository CRAN

Date/Publication 2021-07-27 11:40:02 UTC

R topics documented:

add_covariates.mrgmod .................................................. 2
adm_0_cmt ................................................................. 3
adm_cmt ................................................................. 3
adm_lines.mrgmod ........................................................ 4
as.data.frame.mapbayests ............................................. 4
augment ................................................................. 5
augment.mapbayests ................................................... 5
add_covariates.mrgmod

Add covariates columns to data

Description
Add covariates columns to data

Usage
```r
## S3 method for class 'mrgmod'
add_covariates(x, covariates = list(), ...)
```

Arguments
- **x**: model object
- **covariates**: a list of named covariates, with a single value or exact number of lines than data
- ...: not used

Value
model object with dataset
**adm_0_cmt**  

Get zero-order infusion compartment from mrgsolve model

**Description**

Get zero-order infusion compartment from mrgsolve model

**Usage**

```r
adm_0_cmt(x)
```

**Arguments**

- `x`  
  model object

**Value**

vector of integer

**Examples**

```r
model <- mrgsolve::mread("ex_mbr1", mbplib())
adm_0_cmt(model)
```

---

**adm_cmt**  

Get administration compartment numbers from mrgsolve model

**Description**

Get administration compartment numbers from mrgsolve model

**Usage**

```r
adm_cmt(x)
```

**Arguments**

- `x`  
  model object

**Value**

vector of integer

**Examples**

```r
model <- mrgsolve::mread("ex_mbr1", mbplib())
adm_cmt(model)
```
adm_lines.mrgmod  

Add administrations lines to data

Description
Add administrations lines to data

Usage
## S3 method for class 'mrgmod'
adm_lines(x, ...)

Arguments
x  
model object
...
  passed to mrgsolve::ev

Value
model object with dataset

as.data.frame.mapbayests

Return the mapbay_tab as a data.frame

Description
Return the mapbay_tab as a data.frame

Usage
## S3 method for class 'mapbayests'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

Arguments
x  
A mapbayests object.
row.names, optional, ...
  passed to as.data.frame

Value
  a data.frame (the mapbay_tab from estimation)
**augment**

*Compute full PK profile prediction from mapbayr estimates.*

**Description**

Compute full PK profile prediction from mapbayr estimates.

**Usage**

```r
augment(x, ...)
```

**Arguments**

- `x` object to augment
- `...` additional arguments

**Value**

an augmented object (depending on the object passed)

---

**augment.mapbayests**

*Compute full PK profile prediction from mapbayr estimates.*

**Description**

Compute full PK profile prediction from mapbayr estimates.

**Usage**

```r
## S3 method for class 'mapbayests'
augment(x, data = NULL, end = NULL, ...)
```

**Arguments**

- `x` A mapbayests object.
- `data` dataset to pass to mrgsolve for simulation (default is dataset used for estimation)
- `end` end of infusion time (passed to mrgsim)
- `...` additional argument to pass to mrgsim

**Value**

a mapbayests object, augmented of an aug_tab
check_mapbayr_model  
Check if model is valid for mapbayr

Description
Check if model is valid for mapbayr

Usage
check_mapbayr_model(x)

Arguments
x  model file

Value
TRUE value if check is passed, a vector of character with errors otherwise.

Examples
library(mapbayr)
library(mrgsolve)
check_mapbayr_model(house())

compute_ofv  
Compute the objective function value

Description
Compute the objective function value

Usage
compute_ofv(
  eta,
  mrgsolve_model,
  data,
  sigma,
  log_transformation,
  DVobs,
  omega.inv,
  obs_cmt
)
Arguments

- **eta**: a named vector of numeric (ETA1, ETA2...)
- **mrgsolve_model**: model object (updated for purpose)
- **data**: NMTRAN-like data (of one individual)
- **sigma**: matrix
- **log_transformation**: logical. If TRUE, prediction will be log-transformed to compute ofv (useful for log-additive model)
- **DVobs**: vector of observation to fit
- **omega.inv**: inverse of omega matrix
- **obs_cmt**: vector of compartments with observations to fit

Details

This function is called iteratively by the optimization function. Arguments should not be passed directly, but generated by the pre-processing functions (see `preprocess.ofv`)

Value

a single numeric value (the objective function value)

Description

Helpful functions to pass information about administrations (`adm_lines()`), observations (`obs_lines()`), and covariates (`add_covariates()`). These functions are passed to a `mrgmod` object (mrgsolve model), and return a `mrgmod` object with a dataset inside, so that mrgsolve or mapbayr functions can be passed along within a pipe-friendly workflow.

Usage

- `adm_lines(x, ...)`
- `obs_lines(x, time, DV, mdv = 0, cmt = NULL, DVmet = NULL, ...)`
- `add_covariates(x, covariates, ...)`
- `see_data(x, ...)`
derivatives

Arguments

- \( x \) model object
- ... passed to \texttt{mrgsolve::ev()} in \texttt{adm_lines()}
- \( \text{time}, \text{DV}, \text{mdv}, \text{cmt}, \text{DVmet} \) passed to \texttt{obs_lines()}
- \( \text{covariates} \) a list of named covariates, with a single value or same number of lines than data

Value

- a \texttt{mrgmod} object, with a dataset in the @\texttt{args$data} slot.

---

Derivatives  
Compute the derivatives

Usage

derivatives(v_DV, v_cmt, cmts)

Arguments

- \( \text{v}_\text{DV} \) vector of concentrations to derive (typically output$DV)
- \( \text{v}_\text{cmt} \) vector of compartment associated to the concentrations (typically output$cmt)
- \( \text{cmts} \) numbers of compartment compartment associated with an observations, to derive concentration (typically obs_cmt)

Value

- a matrix

Examples

derivatives(  
  v_DV = c(400, 40, 200, 20),  
  v_cmt = c(2, 3, 2, 3),  
  cmts = c(2,3)  
)
get_data.mapbayests

Return data from a mapbayests

Description
Return data from a mapbayests

Usage
## S3 method for class 'mapbayests'
get_data(x, ...)

Arguments
- x: mapbayests object
- ...: not used

Value
a tibble

get_data.mrgmod

Return data from a mrgmod

Description
Return data from a mrgmod

Usage
## S3 method for class 'mrgmod'
get_data(x, ...)

Arguments
- x: model object
- ...: not used

Value
a tibble
### get_eta.mapbayests

Return eta from a mapbayests

#### Usage

```r
## S3 method for class 'mapbayests'
get_eta(x, ..., output = NULL)
```

#### Arguments

- `x`: mapbayests object
- `...`: not used
- `output`: either a list ("list"), a data.frame ("df") or a vector of numeric ("num"). Default to "num" if only one ID.

#### Value

a tibble

### get_param.mapbayests

Return a posteriori param from a mapbayests

#### Usage

```r
## S3 method for class 'mapbayests'
get_param(x, ..., output = NULL, keep_ID = NULL, keep_names = NULL)
```

#### Arguments

- `x`: mapbayests object
- `...`: passed along
- `output`: either a data.frame ("df") or a vector of numeric ("num"). Default to "num" if only one ID
- `keep_ID`: a logical. By default, the ID variable is dropped if one ID in data.
- `keep_names`: a logical. By default, names are dropped if one parameter is requested, and output is not a data frame.
**get_x**

*Get content from object*

**Description**

Helpful functions to get content from a `mrgmod` object (i.e. data) or from a `mapbayests` object (data, eta, param).

**Usage**

- `get_data(x, ...)`
- `get_eta(x, ...)`
- `get_param(x, ...)`

**Arguments**

- **x** object to get content from
- **...** passed along

**Value**

the class of the object returned depends on the function, and on their arguments. Typically, a data.frame or a vector if the output can be reduced to one line.

**hist.mapbayests**

*Plot posterior distribution of bayesian estimates*

**Description**

Plot posterior distribution of bayesian estimates

**Usage**

```r
## S3 method for class 'mapbayests'
hist(x, ...)
```

**Arguments**

- **x** A `mapbayests` object.
- **...** additional arguments (not used)
mapbayest

Estimate parameters (maximum a posteriori)

Description

The main function of the mapbayr package. Performs a maximum a posteriori Bayesian estimation of parameters, from a mrgsolve model object and a dataset containing information about administrations and observed concentrations.

Usage

mapbayest(
  x,
  data = NULL,
  method = "L-BFGS-B",
  force_initial_eta = NULL,
  quantile_bound = 0.001,
  control = list(),
  check = TRUE,
  verbose = TRUE,
  reset = TRUE,
  output = NULL
)

Arguments

  x the model object
  data NMTRAN-like data set
  method optimization method; possible values are L-BFGS-B (the default) and newuoa
  force_initial_eta a vector of numeric values to start the estimation from (default to 0 for "L-BFGS-B")
  quantile_bound a numeric value representing the quantile of the normal distribution admitted to define the bounds for L-BFGS-B (default is 0.001, i.e. 0.1%)
  control a list passed to the optimizer (see optimx documentation)
  check check model code for mapbayr specification (a logical, default is TRUE)
  verbose print the steps of the estimations to the console (a logical, default is TRUE)
  reset reset optimizer with new initial eta values if numerical difficulties, or with new bounds (L-BFGS-B) if estimate equal to a bound. (a logical, default is TRUE)
if NULL (the default) a mapbayests object is returned; if df a mapbay tab dataframe is returned

passed to mapbayest

Value

a mapbayests model object
### obs_lines.mrgmod

**Add observations lines to data**

**Description**

Add observations lines to data

**Usage**

```r
## S3 method for class 'mrgmod'
obs_lines(x, time, DV, mdv = 0, cmt = NULL, DVmet = NULL, ...)
```

**Arguments**

- `x`: model object
- `time`: vector of time
- `DV`: vector of values to fit
- `mdv`: optional should the DV be ignored (1) or not (0)
- `cmt`: an integer. Optional, cmt where observation is measured
- `DVmet`: optional : metabolite data to fit
- `...`: not used

**Value**

model object with dataset

---

### plot.mapbayests

**Plot predictions from mapbayests object**

**Description**

Plot predictions from mapbayests object

**Usage**

```r
## S3 method for class 'mapbayests'
plot(x, ...)
```

**Arguments**

- `x`: A mapbayests object.
- `...`: additional arguments (not used)

**Value**

a ggplot object. Observed and predicted concentration vs time profile for every patients.
Postprocess mapbayr

Description

Functions to generate postprocess after optimization.

Usage

postprocess.optim(x, data, opt.value)

postprocess.output(
    x,
    arg.optim,
    arg.ofv.fix,
    arg.ofv.id,
    opt.value,
    post,
    output,
    times
)

Arguments

x the model object
data NMTRAN-like data set
opt.value value returned by optimizer
arg.optim, arg.ofv.fix, arg.ofv.id argument passed to optimizer
post, times output of the post.process function
output if NULL (the default) a mapbayests object is returned; if df a mapbay_tab dataframe is returned

Value

postprocess.optim() returns a list with final parameters and mapbay_tab. postprocess.output() returns a mapbayests class object.
**preprocess.ofv**

*Preprocess model and data for ofv computation*

**Description**

Functions to generate arguments passed to `compute.ofv`. Arguments that are fixed between individuals are created once (preprocess.ofv.fix), while other are specific of each individual (preprocess.ofv.id).

**Usage**

```r
preprocess.ofv.fix(x, data)
preprocess.ofv.id(x, iddata)
```

**Arguments**

- `x` the model object
- `data`, `iddata` NMTRAN-like data set. `iddata` is likely a dataset of one individual

**Value**

A list of arguments used to `compute.ofv()`.

---

**preprocess.optim**

*Pre-process: arguments for optimization function*

**Description**

Pre-process: arguments for optimization function

**Usage**

```r
preprocess.optim(x, method, control, force_initial_eta, quantile_bound)
```

**Arguments**

- `x` the model object
- `method` optimization method; possible values are L-BFGS-B (the default) and newuoa
- `control` a list passed to the optimizer (see `optimx` documentation)
- `force_initial_eta` a vector of numeric values to start the estimation from (default to 0 for "L-BFGS-B")
- `quantile_bound` a numeric value representing the quantile of the normal distribution admitted to define the bounds for L-BFGS-B (default is 0.001, i.e. 0.1%)
Value

a list of named arguments passed to optimizer (i.e. arg.optim)

Description

Print a mapbayests object

Usage

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

Arguments

x A mapbayests object.
...
additional arguments

Value

print the results of the estimation to the console, and returns it invisibly.

Description

Use posterior param and covariates

Usage

use_posterior(x, .zero_re = c("both", "omega", "sigma"))

Arguments

x A mapbayests object.
.zero_re Default is "both", meaning all matrices are zeroed. Pass "omega" to zero between-subject variability, and keep simulating residual error.

Details

Updates the param values of the model object with the estimated etas, and the covariates of the individual. Returns an updated mrgmod, so that the user can derive simulations from it. Works only with one individual. Does not handle time-varying covariates.
Value

a mrgmod
Index

add_covariates (data_helpers), 7
add_covariates.mrgmod, 2
adm_0_cmt, 3
adm_cmt, 3
adm_lines (data_helpers), 7
adm_lines.mrgmod, 4
as.data.frame.mapbayests, 4
augment, 5
augment.mapbayests, 5
check_mapbayr_model, 6
compute_ofv, 6, 16
data_helpers, 7
derivatives, 8
get_data (get_x), 11
get_data.mapbayests, 9
get_data.mrgmod, 9
get_eta (get_x), 11
get_eta.mapbayests, 10
get_param (get_x), 11
get_param.mapbayests, 10
get_x, 11
hist.mapbayests, 11
mapbayest, 12
mbrest (mapbayest), 12
mbrlib, 13
obs_cmt, 13
obs_lines (data_helpers), 7
obs_lines.mrgmod, 14
optimx, 12, 16
plot.mapbayests, 14
postprocess, 15
preprocess.ofv, 7, 16
preprocess.optim, 16
print.mapbayests, 17
see_data (data_helpers), 7
use_posterior, 17