Package ‘nlmixr2extra’

May 17, 2022

Title  Nonlinear Mixed Effects Models in Population PK/PD, Extra Support Functions

Version  2.0.6

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


License  GPL (>= 3)

URL  https://nlmixr2.github.io/nlmixr2extra/,
     https://github.com/nlmixr2/nlmixr2extra/

BugReports  https://github.com/nlmixr2/nlmixr2extra/issues/

Imports  checkmate, cli, crayon, data.table, digest, ggplot2, ggtext, lotri, nlme, nlmixr2est, Rcpp, rxode2, stats, symengine, utils

Suggests  nlmixr2data, testthat (>= 3.0.0), withr

LinkingTo  Rcpp, RcppArmadillo

Biarch  true

Config/testthat/edition  3

Encoding  UTF-8

NeedsCompilation  yes

RoxygenNote  7.1.2

Author  Matthew Fidler [aut, cre] (<https://orcid.org/0000-0001-8538-6691>), Vipul Mann [aut]

Repository  CRAN

Date/Publication  2022-05-17 09:20:02 UTC
**R topics documented:**

addCovariate . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
addCovVar . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
backwardSearch . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
bootplot . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
bootstrapFit . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
covarSearchAuto . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
forwardSearch . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
initializeCovars . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
makeDummies . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
makeHockeyStick . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
nlmixrDataToMonolix . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13
performNorm . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
preconditionFit . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16
removeCovariate . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
removeCovMultiple . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
removeCovVar . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

Index 19

---

**addCovariate**

*Add covariate expression to a function string*

**Description**

Add covariate expression to a function string

**Usage**

```r
addCovariate(funstring, varName, covariate, theta, isLog)
```

**Arguments**

- `funstring`: a string giving the expression that needs to be modified
- `varName`: the variable to which the given string corresponds to in the model expression
- `covariate`: the covariate expression that needs to be added (at the appropriate place)
- `theta`: a list of names of the 'theta' parameters in the 'fit' object
- `isLog`: a boolean signifying the presence of log-transformation in the funstring

**Value**

returns the modified string with the covariate added to function string

**Author(s)**

Vipul Mann, Matthew Fidler
**Usage**

```r
addCovVar(
  fitobject,
  varName,
  covariate,
  norm = c("median", "mean", "autoscale"),
  norm_type = c("mul", "div", "sub", "add", "autoscale"),
  categorical = FALSE,
  isHS = FALSE,
  initialEst = 0,
  initialEstLB = -Inf,
  initialEstUB = Inf
)
```

**Arguments**

- `fitobject` an nlmixr2 'fit' object
- `varName` a string giving the variable name to which covariate needs to be added
- `covariate` a string giving the covariate name; must be present in the data used for 'fit'
- `norm` the kind of normalization to be used while normalizing covariates; must be either 'mean' or 'median'
- `norm_type` a string defining operator to be used for transforming covariates using 'norm'; must be one among 'mul', 'div', 'sub', 'add'
- `categorical` a boolean indicating if the 'covariate' is categorical
- `isHS` a boolean indicating if 'covariate' is of Hockey-stick kind
- `initialEst` the initial estimate for the covariate parameters to be estimated; default is 0
- `initialEstLB` a lower bound for the covariate parameters to be estimated; default is -Inf
- `initialEstUB` an upper bound for the covariate parameters to be estimated; default is Inf

**Value**

a list with the updated model expression and data with columns corresponding to normalized covariate(s) appended

**Author(s)**

Vipul Mann, Matthew Fidler
backwardSearch | Backward covariate search

Description
Backward covariate search

Usage
backwardSearch(
covInfo,
fitorig,
fitupdated,
pVal = 0.01,
reFitCovars = FALSE,
outputDir,
restart = FALSE
)

Arguments
covInfo | a list containing information about each variable-covariate pair
fitorig | the original 'fit' object before forward search
fitupdated | the updated 'fit' object, if any, after the forward search
pVal | p-value that should be used for selecting covariates in the forward search
reFitCovars | if the covariates should be added before performing backward search - useful for directly performing backward search without forward search; default is FALSE
outputDir | the name of the output directory that stores the covariate search result
restart | a boolean that controls if the search should be restarted; default is FALSE

Value
returns the updated 'fit' object at the end of the backward search and a table of information for all the covariates tested

Author(s)
Vipul Mann, Matthew Fidler
bootplot

Produce delta objective function for bootstrap

Description

Produce delta objective function for bootstrap

Usage

bootplot(x, ...)

## S3 method for class 'nlmixr2FitCore'
bootplot(x, ...)

Arguments

x fit object
...
other parameters

Value

Fit traceplot or nothing.

Author(s)

Vipul Mann, Matthew L. Fidler

References


bootstrapFit

Bootstrap nlmixr2 fit

Description

Bootstrap input dataset and rerun the model to get confidence bounds and aggregated parameters
Usage

```r
bootstrapFit(
  fit,
  nboot = 200,
  nSampIndiv,
  stratVar,
  stdErrType = c("perc", "se"),
  ci = 0.95,
  pvalues = NULL,
  restart = FALSE,
  plotHist = FALSE,
  fitName = as.character(substitute(fit))
)
```

Arguments

- **fit**: the `nlmixr2` fit object
- **nboot**: an integer giving the number of bootstrapped models to be fit; default value is 200
- **nSampIndiv**: an integer specifying the number of samples in each bootstrapped sample; default is the number of unique subjects in the original dataset
- **stratVar**: Variable in the original dataset to stratify on; This is useful to distinguish between sparse and full sampling and other features you may wish to keep distinct in your bootstrap
- **stdErrType**: This gives the standard error type for the updated standard errors; The current possibilities are: "perc" which gives the standard errors by percentiles (default) or "se" which gives the standard errors by the traditional formula.
- **ci**: Confidence interval level to calculate. Default is 0.95 for a 95 percent confidence interval
- **pvalues**: a vector of pvalues indicating the probability of each subject to get selected; default value is NULL implying that probability of each subject is the same
- **restart**: A boolean to try to restart an interrupted or incomplete bootstrap. By default this is FALSE
- **plotHist**: A boolean indicating if a histogram plot to assess how well the bootstrap is doing. By default this is turned off (FALSE)
- **fitName**: is the fit name that is used for the name of the bootstrap files. By default it is the fit provided though it could be something else.

Value

Nothing, called for the side effects; The original fit is updated with the bootstrap confidence bands

Author(s)

Vipul Mann, Matthew Fidler
Examples

```r
one.cmt <- function() {
  ini(
    ## You may label each parameter with a comment
    tka <- 0.45 # Log Ka
    tcl <- 1 # Log Cl
    ## This works with interactive models
    ## You may also label the preceding line with label("label text")
    tv <- 3.45
    label("log V")
    ## the label("Label name") works with all models
    eta.ka ~ 0.6
    eta.cl ~ 0.3
    eta.v ~ 0.1
    add.sd <- 0.7
  )
  model(
    ka <- exp(tka + eta.ka)
    cl <- exp(tcl + eta.cl)
    v <- exp(tv + eta.v)
    linCmt() ~ add(add.sd)
  )
}

fit <- nlmixr2(one.cmt, nlmixr2data::theo_sd, "focei")

withr::with_tempdir({ # Run example in temp dir
  bootstrapFit(fit, nboot = 5, restart = TRUE) # overwrites any of the existing data or model files
  bootstrapFit(fit, nboot = 7) # resumes fitting using the stored data and model files
  # Note this resumes because the total number of bootstrap samples is not 50
  bootstrapFit(fit, nboot=50)
  # Note the bootstrap standard error and variance/covariance matrix is retained.
  # If you wish to switch back you can change the covariance matrix by
  nlmixr2est::setCov(fit,"r,s")
  # And change it back again
  nlmixr2est::setCov(fit,"boot50")
  # This change will affect any simulations with uncertainty in their parameters
  # You may also do a chi-square diagnostic plot check for the bootstrap with
  bootplot(fit)
}
covarSearchAuto

**Stepwise Covariate Model-selection (SCM) method**

Description

Stepwise Covariate Model-selection (SCM) method

Usage

```r
covarSearchAuto(
  fit, 
  varsVec, 
  covarsVec, 
  pVal = list(fwd = 0.05, bck = 0.01), 
  covInformation = NULL, 
  catCovariates = NULL, 
  searchType = c("scm", "forward", "backward"), 
  restart = FALSE
)
```

Arguments

- **fit** an nlmixr2 'fit' object
- **varsVec** a list of candidate variables to which the covariates could be added
- **covarsVec** a list of candidate covariates that need to be tested
- **pVal** a named list with names 'fwd' and 'bck' for specifying the p-values for the forward and backward searches, respectively
- **covInformation** a list containing additional information on the variables-covariates pairs that should be passed on to addCovMultiple function
- **catCovariates** a list of covariates that should be treated as categorical
- **searchType** one of 'scm', 'forward' and 'backward' to specify the covariate search method; default is 'scm'
- **restart** a boolean that controls if the search should be restarted; default is FALSE

Value

A list summarizing the covariate selection steps and output; This list has the "summaryTable" for the overall summary of the covariate selection as well as "resFwd" for the forward selection method and "resBck" for the backward selection method.
Author(s)

Vipul Mann, Matthew Fidler

Examples

eone.cmt <- function() {
  ini({
    ## You may label each parameter with a comment
    tka <- 0.45  # Log Ka
    tcl <- log(c(0, 2.7, 100))  # Log Cl
    ## This works with interactive models
    ## You may also label the preceding line with label("label text")
    tv <- 3.45; label("log V")
    ## the label("Label name") works with all models
    eta.ka ~ 0.6
    eta.cl ~ 0.3
    eta.v ~ 0.1
    add.sd <- 0.7
  })
  model({
    ka <- exp(tka + eta.ka)
    cl <- exp(tcl + eta.cl)
    v <- exp(tv + eta.v)
    linCmt() ~ add(add.sd)
  })
}

fit <- nlmixr2(one.cmt, nlmixr2data::theo_sd, "focei")
rxode2::.rxWithWd(tempdir(), {# with temporary directory
  auto1 <- covarSearchAuto(fit, varsVec = c("ka", "cl"),
                           covarsVec = c("WT", "SEX"), catCovariates = c("SEX"))
})

## Note that this didn't include sex, add it to dataset and restart model

d <- nlmixr2data::theo_sd
d$SEX <- 0
d$SEX[d$ID<=6] <- 1

fit <- nlmixr2(one.cmt, d, "focei")

# This would restart if for some reason the search crashed:
rxode2::.rxWithWd(tempdir(), {# with temporary directory
  auto2 <- covarSearchAuto(fit, varsVec = c("ka", "cl"), covarsVec = c("WT", "SEX"),
                           covarsVec = c("WT", "SEX"), catCovariates = c("SEX"))
})
catCovariates = c("SEX"), restart = TRUE)

auto3 <- covarSearchAuto(fit, varsVec = c("ka", "cl"), covarsVec = c("WT", "SEX"),
                         catCovariates = c("SEX"), restart = TRUE,
                         searchType = "forward")
})

---

### forwardSearch

Forward covariate search

#### Description

Forward covariate search

#### Usage

forwardSearch(covInfo, fit, pVal = 0.05, outputDir, restart = FALSE)

#### Arguments

- **covInfo**: a list containing information about each variable-covariate pair
- **fit**: an nlmixr2 'fit' object
- **pVal**: p-value that should be used for selecting covariates in the forward search
- **outputDir**: the name of the output directory that stores the covariate search result
- **restart**: a boolean that controls if the search should be restarted; default is FALSE

#### Value

returns the updated 'fit' object at the end of the forward search and a table of information for all the covariates tested

#### Author(s)

Vipul Mann, Matthew Fidler
**initializeCovars**

*Initializing covariates before estimation*

**Description**

Initializing covariates before estimation

**Usage**

```r
initializeCovars(
  fitobject,
  fstring,
  covNames,
  initialEst,
  initialEstLB,
  initialEstUB
)
```

**Arguments**

- `fitobject` an nlmixr2 'fit' object
- `fstring` a string giving the entire expression for the model function string
- `covNames` a list of covariate names (parameters) that need to be estimates
- `initialEst` the initial estimate for the covariate parameters to be estimated; default is 0
- `initialEstLB` a lower bound for the covariate parameters to be estimated; default is -Inf
- `initialEstUB` an upper bound for the covariate parameters to be estimated; default is Inf

**Value**

updated model object with the modified initial values

**Author(s)**

Vipul Mann, Matthew Fidler
makeDummies  
*Create categorical covariates*

Description

Create categorical covariates

Usage

makeDummies(data, covariate, varName)

Arguments

data  
a dataframe containing the dataset that needs to be used

covariate  
the covariate that needs to be converted to categorical; must be present in the data

varName  
the variable name to which the given covariate is to be added

Value

a list of updated data with covariates added, an expression that needs to be added to the model expression, the list of covariate names, and the column names corresponding to the categorical covariates

Author(s)

Vipul Mann, Matthew Fidler

makeHockeyStick  
*Creating Hockey-stick covariates*

Description

Creating Hockey-stick covariates

Usage

makeHockeyStick(data, covariate, varName)

Arguments

data  
a dataframe containing the dataset that needs to be used

covariate  
the covariate that needs to be converted to hockey-stick; must be present in the data

varName  
the variable name to which the given covariate is to be added
Value

  a list of updated data with covariates added, an expression that needs to be added to the model expression, the list of covariate names, and the column names corresponding to the hockey-stick covariates

Author(s)

Vipul Mann, Matthew Fidler

---

Convert nlmixr compatible data to other formats (if possible)

Description

Convert nlmixr compatible data to other formats (if possible)

Usage

```r
nlmixrDataToMonolix(
  model,
  data,
  table = nlmixr2est::tableControl(),
  env = NULL
)
```

```r
nlmixrDataToNonmem(model, data, table = nlmixr2est::tableControl(), env = NULL)
```

```r
nlmixrDataToRxode(model, data, table = nlmixr2est::tableControl(), env = NULL)
```

```r
nlmixrDataToMrgsolve(
  model,
  data,
  table = nlmixr2est::tableControl(),
  env = NULL
)
```

Arguments

- **model**: rxode2 model for conversion
- **data**: Input dataset.
- **table**: is the table control; this is mostly to figure out if there are additional columns to keep.
- **env**: When NULL (default) nothing is done. When an environment, the function `nlmixr2est::foceiPreProcessData(data, env, model)` is called on the provided environment.
Value

With the function `nlmixrDataToMonolix()` return a list with:

- Monolix compatible dataset ($monolix$
- Monolix ADM information ($adm$

With the function `nlmixrDataToNonmem()` return a dataset that is compatible with NONMEM.
With the function `nlmixrDataToMrgsolve()` return a dataset that is compatible with mrgsolve. Unlike NONMEM, it supports replacement events with `evid=8` (note with `rxode2` replacement `evid` is 5).

With the function `nlmixrDataToRxode()` this will normalize the dataset to use newer evid definitions that are closer to NONMEM instead of any classic definitions that are used at a lower level

Author(s)

Matthew L. Fidler

Examples

```r
pk.turnover.emax3 <- function() {
  ini({
    tktr <- log(1)
    tka <- log(1)
    tcl <- log(0.1)
    tv <- log(10)
    ##
    eta.ktr ~ 1
    eta.ka ~ 1
    eta.cl ~ 2
    eta.v ~ 1
    prop.err <- 0.1
    pkadd.err <- 0.1
    ##
    temax <- logit(0.8)
    tec50 <- log(0.5)
    tkout <- log(0.05)
    te0 <- log(100)
    ##
    eta.emax ~ .5
    eta.ec50 ~ .5
    eta.kout ~ .5
    eta.e0 ~ .5
    ##
    pdadd.err <- 10
  })
  model({
    ktr <- exp(tktr + eta.ktr)
    ka <- exp(tka + eta.ka)
    cl <- exp(tcl + eta.cl)
    v <- exp(tv + eta.v)
  })
}
Perform normalization of the covariate

Usage

performNorm(
  data, 
  covariate, 
  varName, 
  normOp, 
  normValVec, 
  isLog = FALSE, 
  isCat = FALSE, 
  isHS = FALSE
)
Arguments

- **data**
  a dataframe consisting the covariates added
- **covariate**
  a string giving the covariate name; must be present in the data used for 'fit'
- **varName**
  the variable name to which the covariate is being added
- **normOp**
  an operator indicating the kind transformation to be done on the covariate
- **normValVec**
  a numeric value to be used for normalization of the covariate
- **isLog**
  a boolean indicating the presence of log-transformation in the funstring; default is FALSE
- **isCat**
  a boolean indicating if the covariate is categorical; default is FALSE
- **isHS**
  a boolean indicating if the covariate is of Hockey-stick kind; default is FALSE

Value

a list comprising the update dataframe, the expression for covariate, and a list of covariate names

Author(s)

Vipul Mann, Matthew Fidler

---

**preconditionFit**

*Linearily re-parameterize the model to be less sensitive to rounding errors*

Description

Linearly re-parameterize the model to be less sensitive to rounding errors

Usage

```r
preconditionFit(fit, estType = c("full", "posthoc", "none"), ntry = 10L)
```

Arguments

- **fit**
  A nlmixr2 fit to be preconditioned
- **estType**
  Once the fit has been linearly reparameterized, should a "full" estimation, "posthoc" estimation or simply a estimation of the covariance matrix "none" before the fit is updated
- **ntry**
  number of tries before giving up on a pre-conditioned covariance estimate

Value

A nlmixr2 fit object that was preconditioned to stabilize the variance/covariance calculation
References


removeCovariate

Remove covariate expression from a function string

Description

Remove covariate expression from a function string

Usage

removeCovariate(funstring, varName, covariate, theta)

Arguments

funstring    a string giving the expression that needs to be modified
varName     the variable to which the given string corresponds in the model expression
covariate   the covariate expression that needs to be removed (from the appropriate place)
theta       a list of names of the ‘theta’ parameters in the ‘fit’ object

Value

returns the modified string with the covariate removed from the function string

Author(s)

Vipul Mann, Matthew Fidler

removeCovMultiple

Removing multiple covariates

Description

Removing multiple covariates

Usage

removeCovMultiple(covInfo, fitobject)

Arguments

covInfo     a list containing information about each variable-covariate pair
fitobject   an nlmixr2 ‘fit’ object
removeCovVar

Value

a list with the updated fit object, the variable-covariate pair string, and the parameter names for the corresponding covariates removed

Author(s)

Vipul Mann, Matthew Fidler

removeCovVar  Remove covariate from function string

Description

Function to remove covariates from a given variable’s equation in the function string text

Usage

removeCovVar(fitobject, varName, covariate, categorical = FALSE, isHS = FALSE)

Arguments

fitobject  an nlmixr2 'fit' object
varName   a string giving the variable name to which covariate needs to be added
covariate a string giving the covariate name; must be present in the data used for 'fit'
categorical a boolean to represent if the covariate to be added is categorical
isHS      a boolean to represent if the covariate to be added is hockey-stick normalized

Value

returns a list containing the updated model and the parameter names for the covariates added

Author(s)

Vipul Mann, Matthew Fidler
Index

addCovariate, 2
addCovVar, 3

backwardSearch, 4
bootplot, 5
bootstrapFit, 5

covarSearchAuto, 8
forwardSearch, 10
initializeCovars, 11
makeDummies, 12
makeHockeyStick, 12

nlmixrDataToMonolix, 13
nlmixrDataToMrgsolve
    (nlmixrDataToMonolix), 13
nlmixrDataToNonmem
    (nlmixrDataToMonolix), 13
nlmixrDataToRxode
    (nlmixrDataToMonolix), 13

performNorm, 15
preconditionFit, 16

removeCovariate, 17
removeCovMultiple, 17
removeCovVar, 18