Package 'nlmixr2extra'

October 29, 2024

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

Version 3.0.1

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

Description Fit and compare nonlinear mixed-effects models in differential equations with flexible dosing information commonly seen in pharmacokinetics and pharmacodynamics (Almquist, Leander, and Jirstrand 2015 <doi:10.1007/s10928-015-9409-1>). Differential equation solving is by compiled C code provided in the 'rxode2' package (Wang, Hallow, and James 2015 <doi:10.1002/psp4.12052>). This package is for support functions like preconditioned fits <doi:10.1208/s12248-016-9866-5>, boostrap and stepwise covariate selection.

Depends R (>= 4.0)

License GPL (>= 3)

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

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

Imports checkmate, cli (>= 3.4.0), crayon, data.table, digest, dplyr, ggplot2, ggtext, knitr, lotri, methods, nlme, nlmixr2est (>= 2.1.1), Rcpp, rxode2 (>= 2.0.10), stats, symengine, utils

Suggests brms, nlmixr2data, testthat (>= 3.0.0), withr, devtools

LinkingTo Rcpp, RcppArmadillo

Biarch true

Config/testthat/edition 3

Encoding UTF-8

NeedsCompilation yes

RoxygenNote 7.3.2

LazyData true

Author Matthew Fidler [aut, cre] (<https://orcid.org/0000-0001-8538-6691>),
Vipul Mann [aut],
Vishal Sarsani [aut] (<https://orcid.org/0000-0002-9272-3438>),
Christian Bartels [ctb],
Bill Denney [aut] (<https://orcid.org/0000-0002-5759-428X>)

Repository CRAN

Date/Publication 2024-10-29 18:00:02 UTC

Contents

`adaptivelassoCoefficients` *Return Adaptive lasso coefficients after finding optimal t*

Description

Return Adaptive lasso coefficients after finding optimal t

Usage

```
adaptivelassoCoefficients(
  fit,
  varsVec,
  covarsVec,
  catvarsVec,
  constraint = 1e-08,
  stratVar = NULL,
  ...
)
```

Arguments

fit	nlmixr2 fit.
varsVec	character vector of variables that need to be added
covarsVec	character vector of covariates that need to be added
catvarsVec	character vector of categorical covariates that need to be added
constraint	theta cutoff. below cutoff then the theta will be fixed to zero.
stratVar	A variable to stratify on for cross-validation.
...	Other parameters to be passed to optimalTvaluelasso

Value

return data frame of final lasso coefficients

Author(s)

Vishal Sarsani

Examples

```
## Not run:
one.cmt <- function() {
  ini({
    tka <- 0.45; label("Ka")
    tcl <- log(c(0, 2.7, 100)); label("Cl")
    tv <- 3.45; label("V")
    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)
  })
```

```
}

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

fit <-
 nlmixr2(
   one.cmt, d,
   est = "saem",
   control = list(print = 0)
 )
varsVec <- c("ka","cl","v")
covarsVec <- c("WT")
catvarsVec <- c("SEX")

# Adaptive Lasso coefficients:

lassoDf <- adaptivelassoCoefficients(fit, varsVec, covarsVec, catvarsVec)

## End(Not run)
```

`addCatCovariates` *Make dummy variable cols and updated covarsVec*

Description

Make dummy variable cols and updated covarsVec

Usage

```
addCatCovariates(data, covarsVec, catcovarsVec)
```

Arguments

`data`	data frame used in the analysis
`covarsVec`	character vector of covariates that need to be added
`catcovarsVec`	character vector of categorical covariates that need to be added

Value

return updated Data along with the updated covarsVec

Author(s)

Vishal Sarsani

addorremoveCovariate *Add covariate*

Description

Add covariate

Usage

```
addorremoveCovariate(ui, varName, covariate, add = TRUE)
```

Arguments

ui	compiled rxode2 nlmir2 model or fit
varName	the variable name to which the given covariate is to be added
covariate	the covariate that needs string to be constructed
add	boolean indicating if the covariate needs to be added or removed.

Author(s)

Matthew Fidler, Vishal Sarsani

adjustedlassoCoefficients *Return Adjusted adaptive lasso coefficients after finding optimal t*

Description

Return Adjusted adaptive lasso coefficients after finding optimal t

Usage

```
adjustedlassoCoefficients(
  fit,
  varsVec,
  covarsVec,
  catvarsVec,
  constraint = 1e-08,
  stratVar = NULL,
  ...
)
```

Arguments

fit	nlmixr2 fit.
varsVec	character vector of variables that need to be added
covarsVec	character vector of covariates that need to be added
catvarsVec	character vector of categorical covariates that need to be added
constraint	theta cutoff. below cutoff then the theta will be fixed to zero.
stratVar	A variable to stratify on for cross-validation.
...	Other parameters to be passed to optimalTvaluelasso

Value

return data frame of final lasso coefficients

Author(s)

Vishal Sarsani

Examples

```
## Not run:
one.cmt <- function() {
  ini({
    tka <- 0.45; label("Ka")
    tcl <- log(c(0, 2.7, 100)); label("Cl")
    tv <- 3.45; label("V")
    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)
  })
}

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

fit <- nlmixr2(one.cmt, d, est = "saem", control = list(print = 0))
varsVec <- c("ka","cl","v")
covarsVec <- c("WT")
catvarsVec <- c("SEX")

# Adaptive Lasso coefficients:
```

```
lassoDf <- adjustedlassoCoefficients(fit,varsVec,covarsVec,catvarsVec)

## End(Not run)
```

`bootplot` *Produce delta objective function for boostrap*

Description

Produce delta objective function for boostrap

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

R Niebecker, MO Karlsson. (2013) *Are datasets for NLME models large enough for a bootstrap to provide reliable parameter uncertainty distributions?* PAGE 2013. https://www.page-meeting.org/?abstract=2899

`bootstrapFit` *Bootstrap nlmixr2 fit*

Description

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

Usage

```
bootstrapFit(
  fit,
  nboot = 200,
  nSampIndiv,
  stratVar,
  stdErrType = c("perc", "sd", "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), `"sd"` which gives the standard errors by the using the normal approximation of the mean with standard devaition, or `"se"` which uses the normal approximation with standard errors calculated with `nSampIndiv`
`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 boostrap. 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 boostrap 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

```
## Not run:
one.cmt <- function() {
  ini({
    tka <- 0.45; label("Ka")
    tcl <- 1; label("Cl")
    tv <- 3.45; label("V")
    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, est = "saem", control = list(print = 0))

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 10

bootstrapFit(fit, nboot=10)

# Note the boostrap standard error and variance/covariance matrix is retained.
# If you wish to switch back you can change the covariance matrix by

nlmixr2est::setCov(fit, "linFim")

# And change it back again

nlmixr2est::setCov(fit, "boot10")
```

```
# 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)
})

## End(Not run)
```

`buildcovInfo` *Build covInfo list from varsVec and covarsVec*

Description

Build covInfo list from varsVec and covarsVec

Usage

```
buildcovInfo(varsVec, covarsVec)
```

Arguments

`varsVec`	character vector of variables that need to be added
`covarsVec`	character vector of covariates that need to be added

Value

covInfo list of covariate info

Author(s)

Vishal Sarsani

`buildupatedUI` *Build updated from the covariate and variable vector list*

Description

Build updated from the covariate and variable vector list

Usage

```
buildupatedUI(ui, varsVec, covarsVec, add = TRUE, indep = FALSE)
```

Arguments

ui	compiled rxode2 nlmir2 model or fit
varsVec	character vector of variables that need to be added
covarsVec	character vector of covariates that need to be added
add	boolean indicating if the covariate needs to be added or removed
indep	a boolean indicating if the covariates should be added independently, or sequentially (append to the previous model). only applicable to adding covariate

Value

updated ui with added covariates

Author(s)

Vishal Sarsani

---

fixedControl *Control options for fixed-value likelihood profiling*

Description

Control options for fixed-value likelihood profiling

Usage

```
fixedControl()
```

Value

A validated list of control options for fixed-value likelihood profiling

See Also

`profileFixed()`

Other Profiling: `llpControl()`, `profile.nlmixr2FitCore()`, `profileFixed()`, `profileLlp()`, `profileNlmixr2FitCoreRet()`

`foldgen` *Stratified cross-validation fold generator function, inspired from the caret*

Description

Stratified cross-validation fold generator function, inspired from the caret

Usage

```
foldgen(data, nfold = 5, stratVar = NULL)
```

Arguments

`data`	data frame used in the analysis
`nfold`	number of k-fold cross validations. Default is 5
`stratVar`	Stratification Variable. Default is NULL and ID is used for CV

Value

return data.frame with the fold column attached

Author(s)

Vishal Sarsani, caret

Examples

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

covarsVec <- c("WT")

# Stratified cross-validation data with CMT
df1 <- foldgen(d, nfold=5, stratVar="CMT")

# Stratified cross-validation data with ID (individual)
df2 <- foldgen(d, nfold=5, stratVar=NULL)
```

`horseshoeSummardf` *Create Horseshoe summary posterior estimates*

Description

Create Horseshoe summary posterior estimates

Usage

```
horseshoeSummardf(fit, covarsVec, ...)
```

Arguments

`fit`	compiled rxode2 nlmir2 model fit
`covarsVec`	character vector of covariates that need to be added
`...`	other parameters passed to brm(): warmup = 1000, iter = 2000, chains = 4, cores = 4, control = list(adapt_delta = 0.99, max_treedepth = 15)

Value

Horse shoe Summary data frame of all covariates

Author(s)

Vishal Sarsani, Christian Bartels

Examples

```
## Not run:
one.cmt <- function() {
  ini({
    tka <- 0.45; label("Ka")
    tcl <- log(c(0, 2.7, 100)); label("Cl")
    tv <- 3.45; label("V")
    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)
  })
}

d <- nlmixr2data::theo_sd
fit <- nlmixr2(one.cmt, d, est = "saem", control = list(print = 0))
```

```
covarsVec <- c("WT")

# Horseshoe summary posterior estimates:

#hsDf <- horseshoeSummardf(fit,covarsVec,cores=2)
#brms sometimes may throw a Error in sink(type = "output")
#Issue Should be fixed by uninstalling and re-installing rstan

## End(Not run)
```

`knit_print.nlmixr2FitCore` *Extract the equations from an nlmixr2/rxode2 model to produce a 'LaTeX' equation.*

Description

Extract the equations from an nlmixr2/rxode2 model to produce a 'LaTeX' equation.

Usage

```
## S3 method for class 'nlmixr2FitCore'
knit_print(x, ..., output = "equations")

## S3 method for class 'rxUi'
knit_print(x, ...)
```

Arguments

`x`	The model to extract equations from
`...`	Ignored
`output`	The type of output to request (currently, just "equations")

`lassoCoefficients` *Return Final lasso coefficients after finding optimal t*

Description

Return Final lasso coefficients after finding optimal t

Usage

```
lassoCoefficients(
  fit,
  varsVec,
  covarsVec,
  catvarsVec,
  constraint = 1e-08,
  stratVar = NULL,
  ...
)
```

Arguments

fit	nlmixr2 fit.
varsVec	character vector of variables that need to be added
covarsVec	character vector of covariates that need to be added
catvarsVec	character vector of categorical covariates that need to be added
constraint	theta cutoff. below cutoff then the theta will be fixed to zero
stratVar	A variable to stratify on for cross-validation
...	Other parameters to be passed to optimalTvaluelasso

Value

return data frame of final lasso coefficients

Author(s)

Vishal Sarsani

Examples

```
## Not run:
one.cmt <- function() {
  ini({
    tka <- 0.45; label("Ka")
    tcl <- log(c(0, 2.7, 100)); label("Cl")
    tv <- 3.45; label("V")
    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)
  })
```

```
}

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

fit <- nlmixr2(one.cmt, d, est = "saem", control = list(print = 0))
varsVec <- c("ka","cl","v")
covarsVec <- c("WT")
catvarsVec <- c("SEX")

# Lasso coefficients:

lassoDf <- lassoCoefficients(fit, varsVec, covarsVec, catvarsVec, constraint=1e-08, stratVar = NULL)

## End(Not run)
```

`lassoSummardf` *Create Lasso summary posterior estimates*

Description

Create Lasso summary posterior estimates

Usage

```
lassoSummardf(fit, covarsVec, ...)
```

Arguments

`fit`	compiled rxode2 nlmir2 model fit
`covarsVec`	character vector of covariates that need to be added
`...`	other parameters passed to brm(): warmup = 1000, iter = 2000, chains = 4, cores = 4, control = list(adapt_delta = 0.99, max_treedepth = 15)

Value

Horse shoe Summary data frame of all covariates

Author(s)

Vishal Sarsani, Christian Bartels

Examples

```
## Not run:
one.cmt <- function() {
  ini({
    tka <- 0.45; label("Ka")
    tcl <- log(c(0, 2.7, 100)); label("Cl")
    tv <- 3.45; label("V")
    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)
  })
}

d <- nlmixr2data::theo_sd
fit <- nlmixr2(one.cmt, d, est = "saem", control = list(print = 0))
covarsVec <- c("WT")

# Horseshoe summary posterior estimates:

#lassoDf <- lassoSummardf(fit,covarsVec,cores=2)
#brms sometimes may throw a Error in sink(type = "output")
#Issue Should be fixed by uninstalling and re-installing rstan

## End(Not run)
```

`llpControl` *Control options for log-likelihood profiling*

Description

Control options for log-likelihood profiling

Usage

```
llpControl(
  ofvIncrease = qchisq(0.95, df = 1),
  rseTheta = 30,
  itermax = 10,
  ofvtol = 0.005,
  paramDigits = 3,
  extrapolateExpand = 1.5
)
```

Arguments

ofvIncrease	The targetted change in objective function value (3.84 corresponds to a Chi-squared test with a 95% confidence interval)
rseTheta	The relative standard error (percent) for the model parameters. It can be missing (the default) in which case a default value of 30% will be applied. If given as a single number, it will be applied to all parameters. If given as a named vector of numbers, it will be applied to each named parameter.
itermax	Maximum number of likelihood profiling iterations for each bound estimated
ofvtol	The relative tolerance for the objective function being close enough to the ofvIncrease.
paramDigits	The number of significant digits required for the parameter. When interpolation attempts to get smaller than that number of significant digits, it will stop.
extrapolateExpand	When extrapolating outside the range previously tested, how far should the step occur as a ratio

Value

A validated list of control options for log-likelihood profiling

See Also

profileLlp()

Other Profiling: fixedControl(), profile.nlmixr2FitCore(), profileFixed(), profileLlp(), profileNlmixr2FitCoreRet()

---

normalizedData *Function to return data of normalized covariates*

Description

Function to return data of normalized covariates

Usage

```
normalizedData(data, covarsVec, replace = TRUE)
```

Arguments

data	a dataframe with covariates to normalize
covarsVec	a list of covariate names (parameters) that need to be estimates
replace	replace the original covariate data with normalized data for easier updated model.

Value

data frame with all normalized covariates

Author(s)

Vishal Sarsani

Examples

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

covarsVec <- c("WT")

# Normalized covariate (replaced)
df1 <- normalizedData(d,covarsVec,replace=TRUE)

# Normalized covariate (without replacement)
df2 <- normalizedData(d,covarsVec,replace=FALSE)
```

---

optimUnisampling *Sample from uniform distribution by optim*

Description

Sample from uniform distribution by optim

Usage

```
optimUnisampling(xvec, N = 1000, medValue, floorT = TRUE)
```

Arguments

xvec	A vector of min,max values . Ex:c(10,20)
N	Desired number of values
medValue	Desired Median
floorT	boolean indicating whether to round up

Value

Samples with approx desired median.

Author(s)

Vishal Sarsani

Examples

```
# Simulate 1000 creatine clearance values with median of 71.7 within range of c(6.7,140)
creatCl <- optimUnisampling(xvec=c(6.7,140), N=1000, medValue = 71.7, floorT=FALSE)
```

preconditionFit *Linearly 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

```
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

Aoki Y, Nordgren R, Hooker AC. Preconditioning of Nonlinear Mixed Effects Models for Stabilisation of Variance-Covariance Matrix Computations. AAPS J. 2016;18(2):505-518. doi:10.1208/s12248-016-9866-5

profile.nlmixr2FitCore *Perform likelihood profiling on nlmixr2 focei fits*

Description

Perform likelihood profiling on nlmixr2 focei fits

Usage

```
## S3 method for class 'nlmixr2FitCore'
profile(
  fitted,
  ...,
  which = NULL,
  method = c("llp", "fixed"),
  control = list()
)
```

Arguments

fitted	The fit model
...	ignored
which	The parameter names to perform likelihood profiling on (NULL indicates all parameters)
method	Method to use for profiling (see the details)
control	Control arguments for the method

Value

A data.frame with one column named Parameter indicating the parameter being fixed on that row, one column for the OFV indicating the OFV estimated for the model at that step, one column named profileBound indicating the estimated value for the profile likelihood and its step above the minimum profile likelihood value, and columns for each parameter estimate (or fixed) in the model.

Log-likelihood profiling

method = "llp"

The search will stop when either the OFV is within ofvtol of the desired OFV change or when the parameter is interpolating to more significant digits than specified in paramDigits. The "llp" method uses the profileLlp() function. See its help for more details.

Fixed points

method = "fixed"

Estimate the OFV for specific fixed values. The "fixed" method uses the profileFixed() function. See its help for more details.

See Also

Other Profiling: fixedControl(), llpControl(), profileFixed(), profileLlp(), profileNlmixr2FitCoreRet()

Examples

```
## Not run:
# Likelihood profiling takes a long time to run each model multiple times, so
# be aware that running this example may take a few minutes.
oneCmt <- function() {
  ini({
    tka <- log(1.57)
    tcl <- log(2.72)
    tv <- fixed(log(31.5))
    eta.ka ~ 0.6
    add.sd <- 0.7
  })
  model({
    ka <- exp(tka + eta.ka)
    cl <- exp(tcl)
```

```
    v <- exp(tv)
    cp <- linCmt()
    cp ~ add(add.sd)
  })
}

fit <-
  nlmixr2(
    oneCmt, data = nlmixr2data::theo_sd, est="focei", control = list(print=0)
  )
# profile all parameters
profall <- profile(fit)

# profile a single parameter
proftka <- profile(fit, which = "tka")

## End(Not run)
```

`profileFixed` *Estimate the objective function values for a model while fixing defined parameter values*

Description

Estimate the objective function values for a model while fixing defined parameter values

Usage

```
profileFixed(fitted, which, control = list())

profileFixedSingle(fitted, which)
```

Arguments

`fitted`	The fit model
`which`	A data.frame with column names of parameters to fix and values of the fitted value to fix (one row only).
`control`	A list passed to `fixedControl()` (currently unused)

Value

`which` with a column named `OFV` added with the objective function value of the fitted estimate fixing the parameters in the other columns

Functions

- `profileFixedSingle()`: Estimate the objective function value for a model while fixing a single set of defined parameter values (for use in parameter profiling)

See Also

Other Profiling: fixedControl(), llpControl(), profile.nlmixr2FitCore(), profileLlp(), profileNlmixr2FitCoreRet()

Other Profiling: fixedControl(), llpControl(), profile.nlmixr2FitCore(), profileLlp(), profileNlmixr2FitCoreRet()

---

profileLlp *Profile confidence intervals with log-likelihood profiling*

Description

Profile confidence intervals with log-likelihood profiling

Usage

```
profileLlp(fitted, which, control)
```

Arguments

fitted	The fit model
which	Either NULL to profile all parameters or a character vector of parameters to estimate
control	A list passed to llpControl()

Value

A data.frame with columns named "Parameter" (the parameter name(s) that were fixed), OFV (the objective function value), and the current estimate for each of the parameters. In addition, if any boundary is found, the OFV increase will be indicated by the absolute value of the "profileBound" column and if that boundary is the upper or lower boundary will be indicated by the "profileBound" column being positive or negative, respectively.

See Also

Other Profiling: fixedControl(), llpControl(), profile.nlmixr2FitCore(), profileFixed(), profileNlmixr2FitCoreRet()

profileNlmixr2FitCoreRet *Give the output data.frame for a single model for profile.nlmixr2FitCore*

Description

Give the output data.frame for a single model for profile.nlmixr2FitCore

Usage

```
profileNlmixr2FitCoreRet(fitted, which, fixedVal)
```

Arguments

fitted	The fit model
which	The parameter names to perform likelihood profiling on (NULL indicates all parameters)
fixedVal	The value that which is fixed to in case the model does not converge.

Value

A data.frame with columns named "Parameter" (the parameter name(s) that were fixed), OFV (the objective function value), and the current estimate for each of the parameters. Omega values are given as their variances and covariances.

See Also

Other Profiling: fixedControl(), llpControl(), profile.nlmixr2FitCore(), profileFixed(), profileLlp()

regularmodel *Regular lasso model*

Description

Regular lasso model

Usage

```
regularmodel(
  fit,
  varsVec,
  covarsVec,
  catvarsVec,
  constraint = 1e-08,
  lassotype = c("regular", "adaptive", "adjusted"),
  stratVar = NULL,
  ...
)
```

Arguments

fit	nlmixr2 fit.
varsVec	character vector of variables that need to be added
covarsVec	character vector of covariates that need to be added
catvarsVec	character vector of categorical covariates that need to be added
constraint	theta cutoff. below cutoff then the theta will be fixed to zero.
lassotype	must be 'regular' , 'adaptive', 'adjusted'
stratVar	A variable to stratify on for cross-validation.
...	Other parameters to be passed to optimalTvaluelasso

Value

return fit of the selected lasso coefficients

Author(s)

Vishal Sarsani

Examples

```
## Not run:
one.cmt <- function() {
  ini({
    tka <- 0.45; label("Ka")
    tcl <- log(c(0, 2.7, 100)); label("Cl")
    tv <- 3.45; label("V")
    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)
  })
}

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

fit <- nlmixr2(one.cmt, d, est = "saem", control = list(print = 0))
varsVec <- c("ka","cl","v")
covarsVec <- c("WT")
catvarsVec <- c("SEX")

# Model fit with regular lasso coefficients:

lassoDf <- regularmodel(fit,varsVec,covarsVec,catvarsVec)
# Model fit with adaptive lasso coefficients:

lassoDf <- regularmodel(fit,varsVec,covarsVec,catvarsVec,lassotype='adaptive')
# Model fit with adaptive-adjusted lasso coefficients:

lassoDf <- regularmodel(fit,varsVec,covarsVec,catvarsVec, lassotype='adjusted')

## End(Not run)
```

`theoFitOde` *Example single dose Theophylline ODE model*

Description

This is a nlmixr2 model that is pre-run so that it can be used in package testing and development. It is regenerated whenever binaries of `nlmixr2extra` are created. If there is a binary incompatability between the fit objects, a simple rerun of the installation will fix this nlmixr2 fit object.

Format

A (modified) data frame with 132 rows and 22 columns.

ID Patient identifier

TIME Time (hr)

DV Dependent variable (concentration)

PRED Predictions without any between subject variability

RES Population Residual

WRES Weighted Residuals under the FO assumption

IPRED Individual Predictions

IRES Individual Residuals

IWRES Individual Weighted Residuals

CPRED Conditional Prediction under the FOCE assumption

CRES Conditional Residuals under the FOCE assumption

CWRES Conditional Weighted Residuals under the FOCE assumption

eta.ka Between subject changes for ka

eta.cl Between subject changes for v

depot amount in the depot compartment

center amount in the central compartment

ka Individual ka values

cl Individual cl values

v Individual volume of distribution

tad Time after dose

dosenum Dose number

Index