Package 'blackbox'

July 22, 2025

Title Black Box Optimization and Exploration of Parameter Space

Encoding UTF-8

Version 1.1.46

Date 2023-12-08

Maintainer François Rousset <francois.rousset@umontpellier.fr>

Depends R (>= 3.1.0)

Imports numDeriv, rcdd, geometry (>= 0.3-6), proxy, spaMM (>= 3.1.0), lattice, Rcpp (>= 0.12.10), nloptr, MASS, pbapply, foreach, matrixStats

LinkingTo Rcpp, RcppEigen

Suggests testthat, minqa, lbfgsb3c, igraph

Description Performs prediction of a response function from simulated response values, allowing black-box optimization of functions estimated with some error. Includes a simple user interface for such applications, as well as more specialized functions designed to be called by the Migraine software (Rousset and Leblois, 2012 <doi:10.1093/molbev/MSR262>; Leblois et al., 2014 <doi:10.1093/molbev/msu212>; and see URL). ter functions are used for prediction of likelihood surfaces and implied likelihood ratio confidence intervals, and for exploration of predictor space of the surface. Prediction of the response is based on ordinary Kriging (with residual error) of the input. Estimation of smoothing parameters is performed by generalized cross-validation.

License CeCILL-2

ByteCompile true

URL https://kimura.univ-montp2.fr/~rousset/Migraine.htm

NeedsCompilation yes

Author François Rousset [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-4670-0371>), Raphaël Leblois [ctb] (ORCID: <https://orcid.org/0000-0002-3051-4497>)

Repository CRAN

Date/Publication 2023-12-08 03:20:11 UTC

Contents

`bboptim` *Black-box function optimization*

Description

`bboptim` implements optimization of a black-box function, possibly estimated with error, using prediction of the function by smoothing of its values in a given set of points, followed by a call to `optim` for optimization of the predicted function. `rbb` samples the parameter space of the function using a crude implementation of Expected Improvement (e.g. Bingham et al., 2014) methods: points with the highest predicted probability of improvement of the response value among a set of candidates sampled uniformly are retained.

Usage

```
bboptim(data, ParameterNames = NULL, respName = NULL, control = list(),
     force = FALSE, optimizers = blackbox.getOption("optimizers"), precision=1e-03)
rbb(object,n=NULL,from=NULL,focus=0.75)
```

Arguments

`data`	A data frame including both function parameters and function values (or "response" values).
`ParameterNames`	A character vector, identifying the columns of the data that correspond to the function parameters. If NULL, all columns except the last are assumed to hold parameter values.

respName	A character string, identifying the column of the data that corresponds to the function values. If NULL, the last column is assumed to hold function values.
control	A list passed to the control argument of the optim function; e.g., list(fnscale=-1) for maximization.
force	Boolean, passed to calcGCV. TRUE forces the analysis of data without pairs of response values for given parameter values. This is *not* recommended as there *should* be such pairs. If the response is estimated with error, this is required for good smoothing. If it is deterministic, bboptim will learn it from the information provided by the pairs.
optimizers	(A vector of) character strings, from which the optimization methods are selected. Default are that of calcGCV for the smoothing step, and nloptr with its own "NLOPT_LN_BOBYQA" method for the smoothed function maximization. See the source of the function for other possible methods (the latter being subject to change with little notice).
object	An object of class bboptim
n	Number of distinct points to be returned. n+1 points will be returned (see Details). If NULL, the following default value is used: min(2^(np+1),floor(10*(1+3*log(np)))), where np is the number of function parameters.
from	A larger (>2n) number of points from which n are selected by an expected inmprovement criterion. If NULL, a default value computed as n*floor(10*(1+3*log(np))), where np is the number of function parameters, is used.
focus	A number between 0 and 1. Determines the proportion of points that are sampled closer to the currently inferred maximum (see Details).
precision	target value of prediction variance in inferred optimum.

Details

rbb selects a proportion 1-focus of the returned points according to expected improvement, from points sampled uniformly in a space defined by a tesselation of the fitted object's parameter points. They are completed to n-1 points, by points similarly selected but within a space defined by a selection of fitted points with the best predicted response values. Finally, two replicates of the predicted optimum (the optim $par result contained in the object) are included. A total of n+1 points (n distinct) is thus returned.

Global optimization cannot be proven, but it is tested by the following criteria: (1) the predicted optimum is close enough to the optimum among assessed parameter points (i.e. either the optimum parameters are well approached or the function is flat in some way), and (2) the prediction variance at the inferred optimum is low enough (so that the predictions used in the first criterion can be trusted). Accordingly, conv_crits has elements (1) objective that indicates whether optr$value betters optr_fitted$value by more than control$reltol, if given, or else by more than sqrt(.Machine$double.eps); and (2) precision that indicates whether variance of prediction error at the inferred optimum is lower than the target precision. This variance is computed as described for predict.HLfit, with variances=list(linPred=TRUE,dispVar=TRUE).

Value

bboptim returns an object of class bboptim, a list which includes

optr	the result of the optim call
RMSE	the root meant square prediction error of response at the optimum optr$par
optr_fitted	the best of the fitted points, with its fitted response value and prediction RMSE
fit	the predictor of the response (an HLfit object as returned by corrHLfit, with a predict method, etc.)
conv_crits	Indicators of convergence (see Details)

and some other elements.

rbb returns a data frame.

References

D. Bingham, P. Ranjan, and W.J. Welch (2014) Design of Computer Experiments for Optimization, Estimation of Function Contours, and Related Objectives, pp. 109-124 in Statistics in Action: A Canadian Outlook (J.F. Lawless, ed.). Chapman and Hall/CRC.

Examples

```
# Classical toy example with optional noise
fr <- function(v,sd) {   ## Rosenbrock Banana function
  10 * (v["y"] - v["x"]^2)^2 + (1 - v["x"])^2 + rnorm(1,sd=sd)
}
set.seed(123)

# Initial parameter values, including duplicates. See ?init_grid.
parsp <- init_grid(lower=c(x=0,y=0),upper=c(x=2,y=2),nUnique=25)

#### Without noise
# add function values
simuls <- cbind(parsp,bb=apply(parsp,1,"fr",sd=0))

# optimization
bbresu <- bboptim(simuls)
print(bbresu)

# refine with additional points
if (blackbox.getOption("example_maxtime")>4) {
 while ( any( ! bbresu$conv_crits) ) {
  print(unlist(bbresu$optr[c("par","value")]))
  candidates <- rbb(bbresu)
  newsimuls <- cbind(candidates,bb=apply(candidates,1,"fr",sd=0))
  bbresu <- bboptim(rbind(bbresu$fit$data,newsimuls))
 }
 print(bbresu)
}

#### With noise

if (blackbox.getOption("example_maxtime")>78) {
 set.seed(123)
 simuls <- cbind(parsp,bb=apply(parsp,1,"fr",sd=0.1))
```

```
 bbresu <- bboptim(simuls, precision=0.02)

 while ( any( ! bbresu$conv_crits) ) {
  print(unlist(bbresu$optr[c("par","value")]))
  candidates <- rbb(bbresu)
  newsimuls <- cbind(candidates,bb=apply(candidates,1,"fr",sd=0.1))
  bbresu <- bboptim(rbind(bbresu$fit$data,newsimuls), precision=0.02)
 }
 print(bbresu)
}

# basic plot
## Not run:
require(spaMM)
opt <- bbresu$optr$par
mapMM(bbresu$fit, decorations=points(opt[1],opt[2],cex=2,pch="+"))

## End(Not run)
```

`blackbox` *Black box optimization and response surface exploration*

Description

blackbox allows prediction and optimization of a response function from simulated response values. It also includes procedures designed mainly or only to be called, in a completely automated way without any input by users, by other R packages such as the **Infusion** package, or by R code automatically generated by the Migraine software (see Details). For prediction, blackbox interfaces a C++ library for "ordinary Kriging" (which is jargon for: prediction in a linear mixed model with a constant term as fixed effect). It uses generalized cross validation (GCV) by default to estimate smoothing parameters.

Details

Beyond the usage illustrated below, this package is used in particular for smoothing the output of the `Migraine` software for likelihood analysis of population genetic data (https://kimura.univ-montp2.fr/~rousset/Migraine.htm). In the latter application the response function is a simulated log-likelihood surface and the procedures generate plots of the (profile) log-likelihood, compute (profile) likelihood ratio confidence intervals, and design new parameter points where the likelihood should be simulated. This package provides documentation for all user-level functions in the R script written by Migraine. Control from Migraine uses many variables stored globally in the list of options accessible through `blackbox.options()`.

The C++ DLL was originally a C++ reimplementation of some of the internal functions of the **fields** package, circa 2005-2006. To estimate smoothing parameters, it requires pairs of responses values for some values of the predictor variables, but will not allow more than pairs.

Author(s)

François Rousset, with contributions by Raphaël Leblois.

References

Fields Development Team (2006). fields: Tools for Spatial Data. National Center for Atmospheric Research, Boulder, CO. https://www.image.ucar.edu/GSP/Software/Fields/.

Examples

```
fr <- function(v) { ## Rosenbrock Banana function with noise
  10 * (v["y"] - v["x"]^2)^2 + (1 - v["x"])^2 + rnorm(1,sd=0.1)
}
set.seed(123)

# Initial parameter values, including duplicates. See ?init_grid.
parsp <- init_grid(lower=c(x=0,y=0),upper=c(x=2,y=2))

# add function values
simuls <- cbind(parsp,bb=apply(parsp,1,"fr"))

## The following shows the backbone of the 'bboptim' code:

sorted_etc <- prepareData(data=simuls)
#   Then smoothing using GCV (beware of implicit 'decreasing=FALSE' argument)
gcvres <- calcGCV(sorted_etc)

## The results can be used as input to functions from other packages,
##  e.g. fitme from spaMM:
## Not run:
require(spaMM)
fitme(bb ~ 1+Matern(1|x+y),data=sorted_etc,
          fixed=list(rho=1/gcvres$CovFnParam[c("x", "y")],
          #         note '1/...'
                      nu=gcvres$CovFnParam[["smoothness"]],
                      phi=gcvres$pureRMSE^2,
          # note distinct meaning of lambda notation in spaMM and blackbox
                      lambda=with(gcvres,(pureRMSE^2)/lambdaEst)))

## GCV is distinct from an REML fit:
fitme(bb ~ 1+Matern(1|x+y),data=sorted_etc,
          init=list(rho=c(1,1)), method="REML")

## End(Not run)
```

`buildFONKgpointls` *Prepare data for smoothing*

Description

From a data frame, builds another data frame. The input data frame must contain values of the canonical parameters of the model and the variables required to construct the smoothed response. Which of the (output) parameters are variable is also determined for later use.

Usage

```
buildFONKgpointls(pointls)
```

Arguments

pointls	A data frame obtained as return value from buildPointls

Details

With controls set by the Migraine software, this can operate transformations of parameter space as well as transformations in logarithmic scale (see islogscale). The output frame will then contain values of transformed parameters.

Value

A data frame.

---

buildPointls *Read a data file*

Description

This reads a data file into a data frame, performs various checks, assign namesto columns, and can select rows.

Usage

```
buildPointls(dataFile = blackbox.getOption("dataFile"), respCols = NULL,
             subsetRows = NULL, ycolname, cleanResu = "")
```

Arguments

dataFile	Name of data file
respCols	A way to select response columns in later analyses (see Details). NULL or a numeric vector.
subsetRows	A set of rows to select. All rows are retained if this is NULL
ycolname	A name to be given to the response variable; willbe used in many further outputs.
cleanResu	A connection, or a character string naming a file for some nicely formated output. If "" (the default), print to the standard output connection.

Details

The input file is a an ASCII numeric data table with the following columns. The first columns contain values of all canonical parameters of the model in canonical order, as given by `blackbox.getOption{"ParameterNames`
Pairs of lines may have identical parameter vectors, but not more than pairs. The next columns may all be used as response variables.

`respCols` identifies columns that will be used to construct the smoothed response (but all columns are retained in this function's return value). If it is NULL, then the last column will be used. If a numeric vector, it identifies response columns (where column 1 is the first column after the parameters columns) which values will be summed to construct the response variable.

Value

A data frame with as many columns as the input table. As a side effect, the function sets the `blackbox.options` `ycolname` and `respCols` to respectively the input `ycolname` and to the column names deduced from the input `respCols` indices.

`calc1DCIs` *Compute 1D confidence intervals*

Description

This computes 1D confidence intervals from an inferred likelihood surface by profile likelihood ratio methods

Usage

```
calc1DCIs(oneDimCIvars, FONKgNames, fittedNames, CIlevel = blackbox.getOption("CIlevel"),
          nextBounds = blackbox.getOption("nextBounds"),
          NextBoundsLevel = blackbox.getOption("NextBoundsLevel"),
          boundsOutfile = "", dataString = "", cleanResu = "")
```

Arguments

`oneDimCIvars`	The names of parameters for which confidence intervals are computed
`FONKgNames`	The names of "Fitted Or Not" parameters (see Details in `blackbox.options` for this concept)
`fittedNames`	The names of fitted parameters (see Details in `blackbox.options` for this concept)
`CIlevel`	Level (1-coverage) of the confidence intervals. Default is 0.05.
`nextBounds`	For development purposes, not documented
`NextBoundsLevel`	For development purposes, not documented
`boundsOutfile`	For development purposes, not documented
`dataString`	A prefix string in some outputs.
`cleanResu`	A connection, or a character string naming a file for some nicely formated output. If `""` (the default), print to the standard output connection.

Value

Returns invisibly a list of profile points that met the CI level for each parameter.

`calc1Dprofiles` *One and two-dimensional profiles, and surface plots*

Description

Assuming that `calcPredictorOK` and `maximizeOK` have been first run: `calc1Dprofiles` plots 1D profiles of a predicted likelihood surface for each of the parameters. Poor profiles mayresult when only local optima are found for some parameter values. The next function provides an improvement over this. `calcProfileLR` plots 2D profiles of the predicted response surface relative to its maximum for pairs of parameters. It also prots 1D profiles taking benefit of the computation effort for the 2D profiles. `calc2D3Dplots` plots the predicted response surface (no profile) in different ways depending on the number of parameters.

These functions have almost no arguments, as almost all control is through global controls. See in particular `gridStepsNbr` (for profile plots) and `graphicPars` in `blackbox.options`.

Usage

```
calc1Dprofiles(varNames=blackbox.getOption("spec1DProfiles"))
calcProfileLR(varNames=blackbox.getOption("fittedNames"),
              pairlist=list(),
              cleanResu="")
calc2D3Dplots(plotFile=NULL,pairlist=list())
```

Arguments

`plotFile`	If a character string, the name of the file where plots are written. Otherwise, plots are output to the screen.
`varNames`	A character vector specifying the names of predictor variables to be considered. For `calc1Dprofiles` (used in conjunction with the `Migraine` software), if the default argument is `NULL`, all variable canonical parameters plus some composite ones may be considered (see the source code for details).
`pairlist`	A list of character vectors. Each vector describes a pair of predictor variables. With the default value `list()`, a default non-empty list may be constructed when `calc2D3Dplots` or `calcProfileLR` is typically used in conjunction with the `Migraine` software (see the source code for details).
`cleanResu`	A connection, or a character string naming a file for some nicely formated output. If it is `""` (the default), print to the standard output connection.

Details

If there is only **one** parameter, `calc2D3Dplots` plots the predicted response as function of this parameter

If there are **two** parameters, `calc2D3Dplots` plots the response surface both as a 2D surface plot and as a 3D perspective plot, and `calcProfileLR` also produces a plot of the response surface (no profiling is needed) relative to its maximum (hence, a likelihood ratio, if the response is a likelihood).

If there are **more** parameters, `calc2D3Dplots` plots a "slice" of the predicted surface, both as a 2D surface plot and as a 3D perspective plot, for each pair of parameters. A slice plot for a pair of parameters fixes all other parameters to values maximizing the response (hence, maximum likelihood estimates, if the response is a likelihood). `calcProfileLR` plots the profile response surface relative to its maximum (hence, a profile likelihood ratio, if the response is a likelihood) for pairs of parameters in `varNames`.

Two dimensional profile plots not only require many numerical maximizations, but will look ugly whenever one of these maximizations fails to find the right maximum, hence additional intensive computations are performed to minimize this problem. As a result, they are quite slow to compute, unless a low `gridStepsNbr` (say < 16) is used, in which case they do not look smooth.

Value

Returns NULL invisibly

---

calcGCV *Estimate smoothing parameters by generalized cross-validation (GCV)*

Description

Smoothing is based on prediction in a linear mixed model ("Kriging") with non-zero residual variance. The correlation function for the random effect is the Matern function with argument the Euclidian distance between scaled coordinates (x/scale). The Matern function also has a smoothness parameter. These parameters are by default estimated by GCV. For large data sets (say >2000 rows), it is strongly recommended to select a subset of the data using `GCVptnbr`, as GCV will otherwise be very slow.

Usage

```
calcGCV(sorted_data=data, data, CovFnParam = NULL, GCVptnbr = Inf,
       topmode = FALSE, verbose = FALSE, cleanResu = "",
       force=FALSE, decreasing=FALSE,
       verbosity = blackbox.getOption("verbosity"),
       optimizers = blackbox.getOption("optimizers"))
```

Arguments

sorted_data	A data frame with both predictor and response variance, sorted and with attributes, as produced by prepareData
data	Obsolete, for Migraine back-compatibility, should not be used.
CovFnParam	Optional fixed values of scale factors for each predictor variable. Smoothness should not be included in this argument.
GCVptnbr	Maximum number of rows selected for GCV.
topmode	Controls the way rows are selected. For development purposes, should not be modified
verbose	Whether to print some messages or not. Distinct from verbosity
verbosity	Distinct from verbose. See verbosity in blackbox.options
cleanResu	A connection, or a character string naming a file for some nicely formated output. If "" (the default), print to the standard output connection.
force	Boolean. Forces the analysis of data without pairs of response values for given parameter values.
optimizers	A vector of) character strings, from which the optimization method is selected. Default is nloptr with its own "NLOPT_LN_BOBYQA" method. See the source of the function for other methods (the latter being subject to change with little notice).
decreasing	Boolean. Use TRUE if you want the result to be used in function maximization rather than minimization.

Value

A list with the following elements

CovFnParam	Scale parameters **and** smoothness parameter of the Matern correlation function
lambdaEst	Ratio of residual variance over random effect variance
pureRMSE	Estimate of root residual variance

and possibly other elements.

Global options CovFnParam is modified as a side effect.

References

Golub, G. H., Heath, M. and Wahba, G. (1979) Generalized Cross-Validation as a method for choosing a good ridge parameter. Technometrics 21: 215-223.

Examples

```
# see example on main doc page (?blackbox)
```

calcLRTs *Compute (profile) likelihood ratio tests*

Description

Assuming that calcPredictorOK and maximizeOK have been first run and that the predicted response surface is a likelihood surface , this performs likelihood ratio (LR) tests for a list of parameter points. Profiles are computed if appropriate, i.e. is the point is lower-dimensional than the the parameter space.

Usage

```
calcLRTs(testPointList, cleanResu = "")
```

Arguments

testPointList	A list of points in predictor (parameter) space. Each point is a numeric vector or list with named elements , the names being those of some parameters.
cleanResu	A connection, or a character string naming a file for some nicely formated output. If "" (the default), print to the standard output connection.

Value

Return a list with information about each LR test, except for tests that could not be performed (e.g. if the tested point is ousdie of the convex envelope of the parameter points from which the predictor has been built). The names of this list's elements are constructed from the tested points. Eachelement is itself a list with elements

LRT	The LR statistics (twice the differnece in log-likelihood between maximized likelihood and profile for the input parameters)
pval	Associated Pvalue by standard chi-square approximation
profpt	Information about the profile point for the input parameters
maxpt	Information about the maximum likelihood point

and other elements, not documented here.

calcPredictorOK *Generate smoothing predictor given smoothing parameters*

Description

Assuming that calcGCV has been first run to estimate smoothing parameter, this produces a "Kriging" predictor of the response.

Usage

```
calcPredictorOK(FONKgpointls, minKrigPtNbr = blackbox.getOption("minKrigPtNbr"),
                krigmax = NULL, topmode = FALSE, rawPlots = TRUE, cleanResu = "")
```

Arguments

FONKgpointls	Input data frame as produced by buildFONKgpointls
minKrigPtNbr	NULL or numeric. At least this many rows (if available) should be selected for Kriging. The default value depends on the number p of predictor variables and is 90, 159, 500, 1307, 3050, 6560 for p from 1 to 6 (beyond which it is strongly advised to use a non-default value).
krigmax	NULL or Numeric. For large data sets the selected points are not "Kriged" all together. Rather, overlapping blocks of rows are selected and are Kriged separately. This sets the size of the blocks. Default depends on the operating system (see source code).
topmode	Controls the way rows are selected. For development purposes, should not be modified
rawPlots	Boolean. Whether to plot one-dimensional "profiles" of the raw data.
cleanResu	A connection, or a character string naming a file for some nicely formated output. If "" (the default), print to the standard output connection.

Value

Returns invisibly a list with many undocumented elements. Thislist is also stored as a global option "fitobject".

---

init_grid *Define starting points in parameter space.*

Description

This function samples the space of estimated parameters. It also handles other fixed arguments that need to be passed to the function simulating the summary statistics (sample size is likely to be one such argument). The current sampling strategy is crude but achieves three desirable effects: it samples the points uniformly but not independently from each other, avoiding large gaps more than independent ampling would allow; it is not exactly a regular grid; and it can include replicates of some parameter points, required for good inference of a response surface when this inference includes a smoothing step of response values evaluated with some error (as is typical in applications of the Migraine software, for which this function was first conceived).

Usage

```
init_grid(lower=c(par=0), upper=c(par=1), steps=NULL, nUnique=NULL,
          nRepl=min(10L,nUnique), maxmin=TRUE, jitterFac=0.5)
```

Arguments

lower	A vector of lower bounds for the parameters, as well as fixed arguments to be passed to the function simulating the summary statistics. Elements must be named.
upper	A vector of upper bounds for the parameters, as well as fixed parameters. Elements must be named and match those of `lower`.
steps	Number of steps of the grid, in each dimension of estimated parameters. If NULL, a default value is defined from the other arguments. If a single value is given, it is applied to all dimensions. Otherwise, this must have the same length as `lower` and `upper` and named in the same way as the variable parameters in these arguments.
nUnique	Number of distinct values of parameter vectors in output. Default is an heuristic guess for good start from not too many points, computed as `floor(50^((v/3)^(1/3)))` where `v` is the number of variable parameters.
nRepl	Number of replicates of distinct values of parameter vectors in output.
maxmin	Boolean. If TRUE, use a greedy max-min strategy (GMM, inspired from Ravi et al. 1994) in the selection of points from a larger set of points generated by an hypercube-sampling step. If FALSE, `sample` is instead used for this second step. This may be useful as the default method becomes slow when thousands of points are to be sampled. GMM was always used in the second step prior to introduction of this argument.
jitterFac	Controls the amount of jitter of the points around regular grid nodes. The default value 0.5 means that a mode can move by up to half a grid step (independently in each dimension), so that two adjacent nodes moved toward each other can (almost) meet each other.

Value

A data frame. Each row defines a list of arguments of vector of the function simulating the summary statistics.

References

Ravi S.S., Rosenkrantz D.J., Tayi G.K. 1994. Heuristic and special case algorithms for dispersion problems. Operations Research 42, 299-310.

Examples

```
set.seed(123)
init_grid()
init_grid(lower=c(mu=2.8,s2=0.5,sample.size=20),
          upper=c(mu=5.2,s2=4.5,sample.size=20),
          steps=c(mu=7,s2=9),nUnique=63)
```

islogscale *Test for parameter log scale*

Description

This tests whether a log scale is used for a parameter.

Usage

```
islogscale(string, scale = blackbox.getOption("FONKgScale"),
           extraScale = blackbox.getOption("extraScale"))
```

Arguments

string	Name of the parameter tested
scale	A vector of scales for parameters of the smoothed object (i.e. parameters in blackbox.getOption("FONKgNames"), see Details in blackbox.options).
extraScale	A vector of scales for additional transformed parameters not in blackbox.getOption("FONKgNames").

Value

A boolean.

maximizeOK *Find maximum of predicted response surface*

Description

Assuming that calcPredictorOK has been first run to produce a predictor of the response surface, this finds its constrained maximum in the convex envelope of the smoothed data.

Usage

```
maximizeOK(fitobject = blackbox.getOption("fitobject"), cleanResu = "")
```

Arguments

fitobject	Return object of calcPredictorOK.
cleanResu	A connection, or a character string naming a file for some nicely formated output. If "" (the default), print to the standard output connection.

Value

A list with element

par	predictor values maximizing the predicted response (in the parameter space used for Kriging)
value	maximum of the predicted response
canonVP	Representation of par in canonical parameter space

and possibly other elements (i) returned by an optimization function such as optim; (ii) values of additional transformed parameters; (iii) cryptic information whether maximization occurred at some boundary of the convex envelope.

---

options *blackbox options settings*

Description

Allow the user to examine a variety of "options" (most of which are not true user options) which affect operations of the blackbox package.

Usage

```
blackbox.options(...)

blackbox.getOption(x)
```

Arguments

x	a character string holding an option name.
...	A named value or a list of named values. Most are not to be manipulated by users and are undocumented. Exceptions are:

ParameterNames See Details

FONKgNames See Details

fittedNames See Details

gridStepsNbr Number of steps of the grid of value for each parameter in profile plots.

graphicPars Graphic parameters used for most plots.

coreNbr Number of cores that R can use for parallel profile computations (see Details for implementation of these).

verbosity=0: Controls display of information about generalized cross-validation. 0 suppresses (most) messages. 1 displays information about estimates and progress of the procedure. Higher values display more information from the optimizer and possibly additional information.

Details

blackbox.options() provides an interface for changing options, many of which are undocumented has they are intended to by used only in conjunction with the Migraine software, in which case the Migraine documentation should be consulted.

The package has been designed first to infer likelihood surfaces by smoothing estimated likelihood points in a model with some canonical parameters (**ParameterNames**). A transformed parameter space may be considered for smoothing, wherein some parameters are variable (**fittedNames**) and others may be constant. The transformed parameter space including constant parameters has names **FONKgNames** (FON for Fitted Or Not).

`blackbox` can perform in parallel manner the Migraine-specific computations of grids of profile log-likelihood values. See the Migraine documentation for user control of the requested number of cores; direct control through R code is possible by `blackbox.options(coreNbr=.)`. If the `doSNOW` back-end is attached (by explicit request from the user), it will be used; otherwise, `pbapply` will be used. Both provide progress bars, but `doSNOW` may provide more efficient load-balancing.

Value

For `blackbox.getOption`, the current value set for option `x`, or `NULL` if the option is unset.

For `blackbox.options()`, a list of all set options. For `blackbox.options(name)`, a list of length one containing the set value, or `NULL` if it is unset. For uses setting one or more options, a list with the previous values of the options changed (returned invisibly).

Examples

```
  blackbox.getOption("verbosity")
  ## Not run:
  blackbox.options(verbosity=1)
  blackbox.options()

## End(Not run)
```

`prepareData` *Prepare data and controls for smoothing*

Description

This sorts the data, identifies parameters and function value (response), identifies pairs of response values for identical parameter values, and may set some global controls in `blackbox.options()`.

Usage

```
prepareData(data, ParameterNames=NULL, respName=NULL,
            verbose=TRUE)
```

Arguments

data	A data frame including variables in ParameterNames and respName
ParameterNames	Names of the variables to be used as predictors of the smoothed surface. If NULL, all columns except the last are assumed to hold parameter values.
respName	Name of the variable to be used as response of the smoothed surface. If NULL, the last column is assumed to hold function values.
verbose	Whether to print some information (in particular a message if replicate responses values are identical for given parameter values, whichwill be suspect in some applications)

Value

A data frame with the required variables, ordered by increasing values as in do.call(order,data). This may set some global controls in blackbox.options() as a side effect.

Examples

```
require(spaMM)
data(blackcap) ## use dataset as template
sorted_etc <- prepareData(data=blackcap,ParameterNames=c("longitude", "latitude"),
                   respName="means")
```

---

preprocessbboptions *Set controls for most functiosn in the package*

Description

Preprocesses a list of argument. The return value of this function serves as argument to blackbox.options (see Examples). Providing in this way the information described in the Details section of blackbox.options is essential for further usage of the package functions.

Usage

```
preprocessbboptions(optionList)
```

Arguments

optionList	A list, with named elements, which names will (mostly) match the names of options set by this function

Value

A list, returned invisibly

Examples

```
## Not run:
GP <- list(ParameterNames=c("theta_1","theta_2"))
pp <- preprocessbboptions(GP)
do.call(blackbox.options, pp) ## essential

## End(Not run)
```

`sampleByResp` *Sample predictor points according to predicted response*

Description

Assuming that `calcPredictorOK` and `maximizeOK` have been first run: predictor points can be sampled in several ways: the convex hull of predictor points with predicted response higher than some threshold value can be sampled uniformly. An Expected Improvement (e.g. Bingham et al., 2014) strategy can be used; whereby points with the highest predicted probability of improvement of the response value among a set of candidates sampled uniformly are retained. An expanded convex hull allowing further exploration of predictor space can also be considered. This function performs various combinations of these methods and (if the response was treated as a likelihood surface) can further use information from any previous likelihood ratio test of confidence interval computations.

Usage

```
sampleByResp(size = blackbox.getOption("nextPointNumber"), outfile = NULL, useEI,
             NextBoundsLevel = 0.001,
             threshold=qchisq(1-NextBoundsLevel, 1)/2,
             rnd.seed = NULL, verbose = FALSE)
```

Arguments

`size`	sample size
`outfile`	If not NULL, the name of an ASCII file where to print the result as a table.
`useEI`	Whether to use an expected improvement criterion
`NextBoundsLevel`	Controls `threshold` in a way meaningful for log-likelihood surfaces
`threshold`	Controls the threshold for selection of the vertices of the convex hull to be sampled, and for inclusion of candidate predictor points in the sample. This threshold corresponds to a difference between predicted value and maximum predicted value. The actual maximal difference for inclusion of vertices additionally depends on the residual error of the predictor.
`rnd.seed`	NULL (in which case nothing is done) or an integer (in which case `set.seed(seed=rnd.seed)` is called).
`verbose`	To print information about evaluation, for development purposes.

Details

The sampling procedure is designed to balance exploration of new regions of the predictor space and filling the top of a likelihood surface, or accurately locating the maximum and bounds of one-dimensional profile likelihood confidence interval. Details are yet to be documented.

Value

Returns the predictor points invisibly.

References

D. Bingham, P. Ranjan, and W.J. Welch (2014) Design of Computer Experiments for Optimization, Estimation of Function Contours, and Related Objectives, pp. 109-124 in Statistics in Action: A Canadian Outlook (J.F. Lawless, ed.). Chapman and Hall/CRC.

`saveOldFile` *Save a copy of an existing file.*

Description

This checks if a file of given name already exists in the current directory, and if so saves a copy of it under an automatically generated name (see below).

Usage

```
saveOldFile(filename)
```

Arguments

`filename`	Name of file to be saved.

Details

This function copies the file named "*first names.ext*" under a name created by inserting a string of the form `.old_`n between "*first names*" and "*.ext*", where n is one more than the highest value for any file, matching the first names and extension, already in the current directory, and 0 if no file matches. For example, if `filename` is `my.beautiful.pdf`, it is copied as `my.beautiful.old_0.pdf` if no `my.beautiful.old_`n`.pdf` file exists, and is is copied as `my.beautiful.old_4.pdf` if `my.beautiful.old_3.pdf` (and any lower n) file exists.

Value

Returns `""` if no file with given name was present on disk, FALSE if it failed to copy an existing old file, and the name of the copy if it successfully copied such a file.

Examples

```
## Not run:
saveOldFile("same.story")

## End(Not run)
```

`writeFinalInfo` *Pretty output, and management of output files*

Description

Final code of the R script written by the Migraine software (https://kimura.univ-montp2.fr/~rousset/Migraine.htm; see main documentation page for the package, for the context). This prints some information, close output files, and beeps to warn that a possibly long computation is finished.

Usage

```
writeFinalInfo(cleanResu = "")
```

Arguments

`cleanResu`	A connection, or a character string naming a file for some nicely formated output. If "" (the default), print to the standard output connection.

Value

returns NULL invisibly.

Index