Package: skUtils (via r-universe)

October 15, 2024

Type Package

Title Helper functions for repgames and dyngames

Version 0.1

Date 2012-03-17

Author Sebastian Kranz

Maintainer Sebastian Kranz <sebastian.kranz@uni-ulm.de>

Description Helper functions needed by my package repgames and dyngames

License GPL (>= 2)

Imports Rcpp (>= 0.11.2)

LinkingTo Rcpp

Repository https://skranz.r-universe.dev

RemoteUrl https://github.com/skranz/skUtils

RemoteRef master

RemoteSha 7355347abd20e6ed9d9c264d6e63ac504aa3ad79

Contents

`add.rowvec` *Add a vector v to each row of m*

Description

Add a vector v to each row of m

Usage

```
add.rowvec(m, v)
```

`approxeq` *APPROXEQ Are a and b approximately equal (to within a specified tolerance)? p = approxeq(a, b, thresh) 'tol' defaults to 1e-3.*

Description

APPROXEQ Are a and b approximately equal (to within a specified tolerance)? p = approxeq(a, b, thresh) 'tol' defaults to 1e-3.

Usage

```
approxeq(a, b, tol = 0.001)
```

`assign.cols` *Assigns all columns of df into variables with the same name in environment env*

Description

Assigns all columns of df into variables with the same name in environment env

Usage

```
assign.cols(df, dest = sys.frame(sys.parent(1)))
```

`calc.mean.from.F.fun` *Calculate numerically the expected value given a cdf*

Description

Calculate numerically the expected value given a cdf

Usage

```
calc.mean.from.F.fun(F.fun, x.min = 0, x.max = Inf, abs.tol = 10^(-10),
  x.seq = NULL, use.num.integrate = TRUE, ...)
```

`check.global.vars` *Some functions that are useful for coding Looks through all loaded functions and searches for global variables that are used within the functions this is a common source for errors*

Description

Some functions that are useful for coding Looks through all loaded functions and searches for global variables that are used within the functions this is a common source for errors

Usage

```
check.global.vars()
```

clone.environment *Clones an environment and its children*

Description

Clones an environment and its children

Usage

```
clone.environment(env, made.clones = as.environment(list(org = list(), copy =
  list())), clone.parents = TRUE, clone.global = FALSE, exclude = NULL,
  clone.children = TRUE)
```

col.matrix *Generates a matrix in which all cols are equal to col*

Description

Generates a matrix in which all cols are equal to col

Usage

```
col.matrix(row = NULL, col, ncol = length(row), dim = 2)
```

colMaxs *Computes quickly the maxima of each column of a matrix*

Description

Computes quickly the maxima of each column of a matrix

Usage

```
colMaxs(mat)
```

colMins *Computes quickly the minima of each column of a matrix*

Description

Computes quickly the minima of each column of a matrix

Usage

```
colMins(mat)
```

`copy.env` *Copies an environment*

Description

Copies an environment

Usage

```
copy.env(dest = sys.frame(sys.parent(1)), source = sys.frame(sys.parent(1)),
  names = NULL, name.change = NULL, exclude = NULL)
```

`discretize.given.F.vec` *Helper function to discretize a continous distribution. F.vec is a finite vector containing the value of the cdf at M different points. The function generates an M dimension vector of probabilities summing up to 1 that discretize the distribution*

Description

Helper function to discretize a continous distribution. F.vec is a finite vector containing the value of the cdf at M different points. The function generates an M dimension vector of probabilities summing up to 1 that discretize the distribution

Usage

```
discretize.given.F.vec(F.vec)
```

`findzero` *Finds position where the function f becomes zero First tries find.root and if this fails tries optimize*

Description

Finds position where the function f becomes zero First tries find.root and if this fails tries optimize

Usage

```
findzero(f, lower, upper, tol = .Machine$double.eps * 10, result.tol = tol,
  try.uniroot = TRUE, ...)
```

`grid.matrix.permutation` *Gives the corresponding rows for a permutated grid.matrix given a permutation x.perm of the elements of the original list x*

Description

Gives the corresponding rows for a permutated grid.matrix given a permutation x.perm of the elements of the original list x

Usage

```
grid.matrix.permutation(x, perm.col)
```

`grid.to.matrix` *Transforms a grid in long format into a matrix*

Description

Transforms a grid in long format into a matrix

Usage

```
grid.to.matrix(grid, nrow = length(unique(grid[, 1])),
  ncol = length(unique(grid[, 2])), val.col = 3)
```

`ls.funs` *List all functions*

Description

List all functions

Usage

```
ls.funs(env = sys.frame(-1))
```

`ls.vars` *List all variables*

Description

List all variables

Usage

```
ls.vars(env = sys.frame(-1))
```

`matrix.to.grid` *Some functions that are useful for manipulating or creating matrices and data.frames and working with lists of vectors, lists of lists or lists of matrices Transforms a matrix into grid in long format*

Description

Some functions that are useful for manipulating or creating matrices and data.frames and working with lists of vectors, lists of lists or lists of matrices Transforms a matrix into grid in long format

Usage

```
matrix.to.grid(mat, x = 1:NROW(mat), y = 1:NCOL(mat), x.name = "x",
  y.name = "y")
```

`named.list` *Some functions that are useful for lists and environments in particular generating, transforming, copying and assigning values Creates a list that is named by the names of its arguments*

Description

Some functions that are useful for lists and environments in particular generating, transforming, copying and assigning values Creates a list that is named by the names of its arguments

Usage

```
named.list(...)
```

paste.matrix.cols *Paste together columns of a matrix or data.frame*

Description

Paste together columns of a matrix or data.frame

Usage

```
paste.matrix.cols(mat, cols = 1:NCOL(mat), ...)
```

paste.matrix.rows *Paste together rows of a matrix or data.frame*

Description

Paste together rows of a matrix or data.frame

Usage

```
paste.matrix.rows(mat, rows = 1:NROW(mat), ...)
```

plot.multi.lines *Plot several lines*

Description

Plot several lines

Usage

```
## S3 method for class 'multi.lines'
plot(mat = NULL, xvar, yvar, ynames = yvar,
  col = NULL, ylim = NULL, xlab = xvar, ylab = "", legend.pos = NULL,
  legend.title = NULL, add = FALSE, lwd = 1, ...)
```

`rbind.list` *rbinds a list of matrices, a list of lists, or a list of vectors into a data.frame (or matrix) each column is a list Assume that all columns in the sublists are in the same order*

Description

rbinds a list of matrices, a list of lists, or a list of vectors into a data.frame (or matrix) each column is a list Assume that all columns in the sublists are in the same order

Usage

```
rbind.list(li, cols = NULL, check.common.cols = FALSE)
```

`row.matrix` *Generates a matrix in which all rows are equal to row*

Description

Generates a matrix in which all rows are equal to row

Usage

```
row.matrix(row, col, nrow = length(col), dim = 1)
```

`rowMaxs` *Computes quickly the minima of each row of a matrix*

Description

Computes quickly the minima of each row of a matrix

Usage

```
rowMaxs(mat)
```

rowMins *Computes quickly the minima of each row of a matrix*

Description

Computes quickly the minima of each row of a matrix

Usage

```
rowMins(mat)
```

set.default *Need to check what it does*

Description

Need to check what it does

Usage

```
set.default(env, name, x, overwrite.null = TRUE, inherits = TRUE)
```

sk.levelplot *My wrapper to the lattice function levelplot. Allows for some own color schemes The parameter focus specifies at which z range stronger color changes shall appear*

Description

My wrapper to the lattice function levelplot. Allows for some own color schemes The parameter focus specifies at which z range stronger color changes shall appear

Usage

```
sk.levelplot(x = NULL, y = NULL, z = NULL, xnames = NULL,
  ynames = NULL, grid.xyz = NULL, col.scheme = "darkredgreen",
  na.col = NULL, at = NULL, at.scheme = "interval", focus = 0,
  cuts = 15, col.regions = NULL, xlab = NULL, ylab = NULL,
  panel = panel.levelplot, zlim = NULL, reverse.colors = FALSE, ...)
```

`sk.optim` *A wrapper for optimization. Allows to specify which variables shall be free Has the same syntax for one and multidimensional optmization Uses optim, omptimize or a grid search*

Description

A wrapper for optimization. Allows to specify which variables shall be free Has the same syntax for one and multidimensional optmization Uses optim, omptimize or a grid search

Usage

```
sk.optim(par, f, lower = NULL, upper = NULL, free.par = 1:NROW(par),
  method = "default", num.grid.steps = NULL, maximize = TRUE,
  f.can.take.matrix = FALSE, tol = .Machine$double.eps^0.25, ...)
```

`sk.pareto.frontier` *Calculates the 2dimensional paretofrontier of the points val1 and val2 The function returns the indices of the points that lie on the Pareto Frontier ordered by val1 and val2.*

Description

Calculates the 2dimensional paretofrontier of the points val1 and val2 The function returns the indices of the points that lie on the Pareto Frontier ordered by val1 and val2.

Usage

```
sk.pareto.frontier(val1, val2, tol = 0, ord = NULL)
```

`which.colMaxs` *Computes quickly the index of the largest element of each column of a matrix*

Description

Computes quickly the index of the largest element of each column of a matrix

Usage

```
which.colMaxs(mat)
```

`which.colMins` *Computes quickly the index of the smallest element of each column of a matrix*

Description

Computes quickly the index of the smallest element of each column of a matrix

Usage

```
which.colMins(mat)
```

`which.rowMaxs` *Computes quickly the index of the largest element of each row of a matrix*

Description

Computes quickly the index of the largest element of each row of a matrix

Usage

```
which.rowMaxs(mat)
```

`which.rowMins` *Computes quickly the index of the smallest element of each row of a matrix*

Description

Computes quickly the index of the smallest element of each row of a matrix

Usage

```
which.rowMins(mat)
```

Index