Package 'statr'

July 26, 2018

Type Package

Title Matt Galloway Personal R Package

Version 0.1.0

Description This is a personal R package. It contains a number of various R functions for organization and convenience purposes.

URL https://github.com/MGallow/statr

BugReports https://github.com/MGallow/statr/issues

License GPL (>= 2)

ByteCompile TRUE

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Imports dplyr,
ggplot2,
magrittr,
formatR,
grid,
devtools,
ADMMsigma,
glmnet

Suggests testthat,
knitr,
rmarkdown,
pkgdown

SystemRequirements GNU make

VignetteBuilder knitr

R topics documented:

`bsearch` *Bisection search*

Description

Minimizes a univariate strictly pseudoconvex function over the interval [a, b]. This is augmented code from Adam Rothman's STAT 8054 course (2017).

Usage

```
bsearch(dg, a, b, L = 1e-07, quiet = FALSE)
```

Arguments

`dg`	the derivative of the function to minimize, where dg(u, ...) is the function evaluated at u.
`a`	left endpoint of the initial interval of uncertainty.
`b`	right endpoint of the initial interval of uncertainty.
`L`	the maximum length of the final interval of uncertainty.
`quiet`	should the function stay quiet?
`...`	additional argument specifications for dg

Value

returns the midpoint of the final interval of uncertainty.

compound *Generate compound symmetric matrices*

Description

Generate a p-dimensional compound symmetric matrix.

Usage

```
compound(p = 8, n = NULL)
```

Arguments

p	desired dimension
n	option to generate n observations from covariance matrix S

Value

Omega, S

Examples

```
# generate compound symmetric matrix with p = 5
compound(p = 5)
```

CVsplit *CV split*

Description

splits data objects into training and testing sets

Usage

```
CVsplit(X, Y, split = 0.5, N = NULL)
```

Arguments

X	nxp data matrix. Each row corresponds to a single observation and each column contains n observations of a single feature/variable.
Y	nxr response matrix. Each row corresponds to a single response and each column contains n response of a single feature/response.
split	fraction of objects devoted to training set
N	option to provide number of objects devoted to training set

Value

X.train, Y.train, X.test, Y.test

data_gen *Normal regression data generator*

Description

True beta values are generated from p*r independent draws from N(0, 1/p) distribution. X are n independent draws from p multivariate normal N(0, SigmaX). Y is then generated using X and true beta values with an iid error term that follows r multivariate normal distribution N(0, Sigma).

Usage

```
data_gen(n, p, r = 1, sparsity = 0.5, Sigma = c("tridiag", "dense",
  "denseQR", "compound"), s = NULL, SigmaX = c("tridiag", "dense",
  "denseQR", "compound"), sx = NULL, ...)
```

Arguments

n	desired sample size
p	desired dimension
r	number of responses
sparsity	desired sparsity for beta
Sigma	covariance matrix structure used to generate Y | X
s	option to specify diagonal elements in Sigma
SigmaX	covariance matrix structure used to generate data X
sx	option to specify diagonal elements in SigmaX
...	additional arguments to pass to data generating functions

Value

Y, X, betas, Sigma, SigmaX

Author(s)

Matt Galloway <gall0441@umn.edu>

Examples

```
# generate 100 observations with predictor dimension equal to 10 and response dimension equal to 5
data = data_gen(n = 100, p = 10, r = 5)
```

dense *Generate dense matrices*

Description

Generate p-dimensional matrices so that its inverse is dense.

Usage

```
dense(p = 8, base = 0.9, n = NULL)
```

Arguments

p	desired dimension
base	base multiplier
n	option to generate n observations from covariance matrix S

Value

Omega, S

Examples

```
# generate dense matrix with p = 5
dense(p = 5)
```

denseQR *Generate dense matrices (via spectral decomposition)*

Description

Generate p-dimensional matrices so that its inverse is dense. The matrix will be generated so its first 'num' eigen values are 1000 and the remaining are 1. The orthogonal basis is generated via QR decomposition of

Usage

```
denseQR(p = 8, num = 5, n = NULL)
```

Arguments

p	desired dimension
num	number of 'large' eigen values. Note num must be less than p
n	option to generate n observations from covariance matrix S

Value

Omega, S

Examples

```
# generate denseQR matrix with p = 5
denseQR(p = 5)
```

`derivative` *Derivative*

Description

Takes the approximate derivative for a given function

Usage

```
derivative(g, x, delta = 1e-07)
```

Arguments

`g`	the derivative of the function to minimize, where dg(u, ...) is the function evaluated at u.
`x`	value to evaluate the derivative at
`delta`	defaults to 10e-8

`diagnostic` *Diagnostic*

Description

This function simply streamlines the process of creating diagnostic plots with ggplot

Usage

```
diagnostic(data., x., y.)
```

Arguments

`data.`	data frame
`x.`	x-axis
`y.`	y-axis

Value

a residual plot and QQ plot

Examples

```
diagnostic(iris, Sepal.Length, Sepal.Width)
```

dsearch *Dichotomous search*

Description

Minimizes a univariate strictly quasiconvex function over the interval [a, b]. This is augmented code from Adam Rothman's STAT 8054 course (2017).

Usage

```
dsearch(g, a, b, L = 1e-07, eps = (L/2.1), quiet = FALSE)
```

Arguments

g	the function to minimize, where g(u, ...) is the function evaluated at u.
a	left endpoint of the initial interval of uncertainty.
b	right endpoint of the initial interval of uncertainty.
L	the maximum length of the final interval of uncertainty.
eps	search parameter, must be less than L/2
quiet	should the function stay quiet?
...	additional argument specifications for g

Value

returns the midpoint of the final interval of uncertainty.

fro *Mean Frobenius norm*

Description

calculates the average squared value of an object. That is, all elements are squared, summed, and divided by the total number of elements

Usage

```
fro(X)
```

Arguments

X	object

Value

norm

LASSO *Lasso regression*

Description

calculate lasso regression coefficients using the optimal tuning parameter from the glmnet package.

Usage

```
LASSO(X, Y, lam = NULL, intercept = FALSE, standardize = FALSE, ...)
```

Arguments

X	nxp data matrix. Each row corresponds to a single observation and each column contains n observations of a single feature/variable.
Y	nxr response matrix. Each row corresponds to a single response and each column contains n response of a single feature/response.
lam	tuning parameter
intercept	option to include intercept, defaults to FALSE
standardize	option to standardize the data, defaults to FALSE
...	other options to pass to glmnet

Value

betas, lam

LDA *Linear Discriminant Analysis*

Description

this function fit the LDA model

Usage

```
LDA(X, y, method = c("MLE", "diagonal", "ridge"), lam = NULL)
```

Arguments

X	n x p matrix where the ith row is the values of the predictor for the ith case
y	n entry response vector where the ith entry is the response category in 1, ..., C for the ith case
method	estimation method
lam	optional tuning parameter specification

Value

returns a list with the parameter estimates

multiplot *Multiple Plot*

Description

Taken from: http://www.cookbook-r.com/Graphs/Multiple_graphs_on_one_page_(ggplot2)/

Usage

```
multiplot(..., plotlist = NULL, cols = 1, layout = NULL)
```

Arguments

`...`	object can be passed in
`plotlist`	plotlist
`cols`	number of columns in layout
`layout`	a matrix specify the layout. If present, 'cols' is ignored

Value

plots

predict_QDA *Predict QDA*

Description

this function classifies test data using a fitted QDA model

Usage

```
predict_QDA(fit, Xtest)
```

Arguments

`fit`	this is a list with elements pi.hats, mu.hats, and Sigma.hats where pi.hats is a list of C response category sample proportions, mu.hats is a list of C p-dimensional sample mean proportions, Sigma.hats is a list of C p by p Sample covariance matrices
`Xtest`	this is a matrix with ntest rows and p column, each row is a test case

Value

returns a vector of ntest entries, where the ith entry is the estimated response category (some value in 1, ..., C) for the ith test case.

QDA *Quadratic Discriminant Analysis*

Description

this function fit the QDA model

Usage

```
QDA(X, y, method = c("MLE", "diagonal", "ridge"), lam = NULL)
```

Arguments

X	n x p matrix where the ith row is the values of the predictor for the ith case
y	n entry response vector where the ith entry is the response category in 1, ..., C for the ith case
method	estimation method
lam	optional tuning parameter specification

Value

returns a list with the parameter estimates

RIDGE *Ridge regression*

Description

calculate ridge regression coefficients using the optimal tuning parameter from the glmnet package.

Usage

```
RIDGE(X, Y, lam = NULL, intercept = FALSE, standardize = FALSE, ...)
```

Arguments

X	nxp data matrix. Each row corresponds to a single observation and each column contains n observations of a single feature/variable.
Y	nxr response matrix. Each row corresponds to a single response and each column contains n response of a single feature/response.
lam	tuning parameter
intercept	option to include intercept, defaults to FALSE
standardize	option to standardize the data, defaults to FALSE
...	other options to pass to glmnet

Value

betas, lam

scatter *Scatter*

Description

This function simply streamlines the process of creating a scatterplot with ggplot

Usage

```
scatter(data., x., y.)
```

Arguments

data.	data frame
x.	x-axis
y.	y-axis

Value

a scatterplot

Examples

```
scatter(iris, Sepal.Length, Sepal.Width)
```

tidy *Tidy*

Description

tidys package R code and updates package documentation. Directly uses Yihui Xie's 'formatR' package.

Usage

```
tidy()
```

timeit *Time-It*

Description

Simple function that prints the computation time of a function

Usage

```
timeit(f)
```

Arguments

f	the function to time

Value

returns the elapsed time

tridiag *Generate tri-diagonal matrices*

Description

Generate p-dimensional matrices so that its inverse is tri-diagonal.

Usage

```
tridiag(p = 8, base = 0.7, n = NULL)
```

Arguments

p	desired dimension
base	base multiplier
n	option to generate n observations from covariance matrix S

Value

Omega, S

Examples

```
# generate tridiagonal matrix with p = 5
tridiag(p = 5)
```

Index