# Package 'statr'

June 7, 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

**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 |

#### Examples

```
compound(p = 10, n = 100)
```

---

| data_gen | *Normal Linear Data Generator* |
|---|---|

---

#### Description

True beta values are generated from p independent draws from N(0, 1/p) distribution. X_-1 are n independent draws from (p - 1) multivariate normal N(0, Sigma) where Sigma has (j, k) entry theta^abs(j - k).

Y is then generated using the X = (1, X_-1) and true beta values with an iid error term that follows distribution N(0, var). We can specify the desired number of replications (reps).

#### Usage

```
data_gen(n, p, theta, var = 0.5, reps = 200)
```

#### Arguments

| | |
|---|---|
| n | desired sample size |
| p | desired dimension |
| theta | parameter used to generate covariance matrix |
| var | variance of generated y values |
| reps | number of replications |

#### Value

generated design matrix (X), response values (Y)(matrix if reps > 1), true beta values

---

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 |

### Examples

```
dense(p = 10, base = 0.9)
```

---

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 |

### Examples

```
denseQR(p = 10, num = 10)
```

---

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.

---

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

---

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 |

### Examples

```
tridiag(p = 10, base = 0.7)
```

# Index