Package 'rrscale'

October 14, 2022

Title Robust Re-Scaling to Better Recover Latent Effects in Data

Version 1.0

Description Non-linear transformations of data to better discover latent effects. Applies a sequence of three transformations (1) a Gaussianizing transformation, (2) a Z-score transformation, and (3) an outlier removal transformation. A publication describing the method has the following citation: Gregory J. Hunt, Mark A. Dane, James E. Korkola, Laura M. Heiser & Johann A. Gagnon-Bartsch (2020) ``Automatic Transformation and Integration to Improve Visualization and Discovery of Latent Effects in Imaging Data", Journal of Computational and Graphical Statistics, <doi:10.1080/10618600.2020.1741379>.

Date 2020-5-22

Depends R (>= 3.5.0)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Imports DEoptim, nloptr, abind

Suggests knitr, rmarkdown, testthat, ggplot2, reshape2

VignetteBuilder knitr

NeedsCompilation no

Author Gregory Hunt [aut, cre],
Johann Gagnon-Bartsch [aut]

Maintainer Gregory Hunt <ghunt@wm.edu>

Repository CRAN

Date/Publication 2020-05-26 11:30:02 UTC

R topics documented:

`asinh` *Arc-hyperbolic-sine transformation*

Description

- T the transformation with arguments Y, the data, lambda the parameter, and boolean inverse to calculate inverse transformation.
- T_deriv the transformation with arguments Y, the data, lambda the parameter.

Usage

```
asinh
```

Format

An object of class `list` of length 2.

`box_cox` *Traditional box-cox power transformation. Accepts one real parameter*

Description

- T the transformation with arguments Y, the data, lambda the parameter, and boolean inverse to calculate inverse transformation
- T_deriv the transformation with arguments Y, the data, lambda the parameter.

Usage

```
box_cox
```

Format

An object of class `list` of length 2.

`box_cox_exp` *Exponential of the tranditional box-cox transformation*

Description

- T the transformation with arguments Y, the data, lambda the parameter, and boolean inverse to calculate inverse transformation.
- T_deriv the transformation with arguments Y, the data, lambda the parameter.

Usage

```
box_cox_exp
```

Format

An object of class `list` of length 2.

`box_cox_negative` *A generalized box-cox transformation that can handle negative data*

Description

- T the transformation with arguments Y, the data, lambda the parameter, and boolean inverse to calculate inverse transformation.
- T_deriv the transformation with arguments Y, the data, lambda the parameter.

Usage

```
box_cox_negative
```

Format

An object of class `list` of length 2.

box_cox_plus1 *Box-cox transformation with a shift of 1 added to the data*

Description

- T the transformation with arguments Y, the data, lambda the parameter, and boolean inverse to calculate inverse transformation.
- T_deriv the transformation with arguments Y, the data, lambda the parameter.

Usage

```
box_cox_plus1
```

Format

An object of class `list` of length 2.

box_cox_plusmin *Box-cox transformation with the data shifted so that it is positive*

Description

- T the transformation with arguments Y, the data, lambda the parameter, and boolean inverse to calculate inverse transformation.
- T_deriv the transformation with arguments Y, the data, lambda the parameter.

Usage

```
box_cox_plusmin
```

Format

An object of class `list` of length 2.

box_cox_shift *Box-cox transformation of shifted variable*

Description

- T the transformation with arguments Y, the data, lambda the parameter, and boolean inverse to calculate inverse transformation. The parameter lambda has two real elements (1) the power and (2) the additive shift to the data.
- T_deriv the transformation with arguments Y, the data, lambda the parameter.

Usage

```
box_cox_shift
```

Format

An object of class `list` of length 2.

center *Centers the data column-wise*

Description

Centers the data column-wise

Usage

```
center(x)
```

Arguments

x	the data.

gm_mean *Calculate the geometric mean*

Description

Calculate the geometric mean

Usage

```
gm_mean(x)
```

Arguments

x	the data.

Examples

```
Y <- rlnorm(10)
gm <- gm_mean(Y)
```

list_transformations *List possible transformations*

Description

Returns list of transformations. Each transformation is a transformation function ("T") accepting a parameter and the derivative of this transformation function ("T_deriv").

Usage

```
list_transformations()
```

log_box_cox *Log of the traditional box-cox transformation*

Description

- T the transformation with arguments Y, the data, lambda the parameter, and boolean inverse to calculate inverse transformation.
- T_deriv the transformation with arguments Y, the data, lambda the parameter.

Usage

```
log_box_cox
```

Format

An object of class `list` of length 2.

`power` *Simple power transformation*

Description

- T the transformation with arguments Y, the data, lambda the parameter, and boolean inverse to calculate inverse transformation.
- T_deriv the transformation with arguments Y, the data, lambda the parameter.

Usage

```
power
```

Format

An object of class `list` of length 2.

`rrscale` *Re-scale a data matrix*

Description

This transformation is three steps (1) Gaussianize the data, (2) z-score Transform the data, and (3) remove extreme outliers from the data. The sequence of these transformations helps focus further analyses on consequential variance in the data rather than having it be focused on variation resulting from the feature's measurement scale or outliers.

Usage

```
rrscale(
  Y,
  trans_list = list(box_cox_negative = box_cox_negative, asinh = asinh),
  lims_list = list(box_cox_negative = c(-100, 100), asinh = list(0, 100)),
  opt_control = NULL,
  opt_method = "DEoptim",
  z = 4,
  q = 0.001,
  verbose = FALSE,
  log_dir = ".rrscale/",
  zeros = FALSE,
  opts = FALSE,
  seed = NULL
)
```

Arguments

Y	Data matrix, data.frame, or list of vectors, to be transformed.
trans_list	List of transformations to be considered. See function list_transformations. Each element of the list should be a list containing the transformation function as the first element and the derivative of the transformation function as the second argument. The first argument of each function should be the data, the second the transformation parameter.
lims_list	List of optimization limits for each transformation from trans_list. This should be a list the same length as trans_list. Each element of the list is a two-element vector that sets the optimization limits for the parameter of each transformation family.
opt_control	Optional optimization controlling parameters for DEoptim control argument. See the DEoptim package for details.
opt_method	Which optimization method to use. Defaults to DEoptim. Other choice is nloptr.
z	The O-step cutoff value. Points are removed if their robust z-score is above z in magnitude.
q	The Z-step winsorizing quantile cutoff. The quantile at which to winsorize the data when calculating the robust z-scores.
verbose	a boolean, if TRUE then save optimization output in log_dir.
log_dir	directory for verbose output. Defaults to ".rrscale/"
zeros	How to deal with zeros in the data set. If set to FALSE the algorithm will fail if it encounters a zero. If set to a number or 'NA' then the zeros are replaced by this number or 'NA'.
opts	Boolean determining if optimization output is returned. Defaults to FALSE.
seed	Sets the seed before running any other analyses.

Value

A list of output:

- opts: the optimization output for all transformation families and all columns
- pars: the optimal parameters for each column for the optimal family
- par_hat: the estimated optimal paramter
- NT: the original data
- RR: the robust-rescaled data
- G: gaussianized data
- Z: robust z-transformed data
- O: data with outliers removed
- rr_fn: a function to apply the estimated RR transformation to new data. Takes arguments
  - Y: the data,
  - z: the z-score cutoff (defaults to 4),
  - q: the winsorizing quantile cutoff (defaults to 0.001),

  - lambda: the transformation parameter to use (defaults to the estimated one),
  - T: the transformation function family (defaults to the optimal estimated family),
  - mu: the mean to be used in the robust z-score step (re-estimates if NULL)
  - sigma: the s.d. to be used in the robust z-score step (re-estimates if NULL)
- T: the optimal family
- T_deriv: the derivative of the optimal family
- T_name: name of the optimal family
- alg_control: the parameters passed to the algorithm

Examples

```
Y <- rlnorm(10)%*%t(rlnorm(10))
rr.out <- rrscale(Y)
Yt <- rr.out$RR
```

---

svdc *The completed SVD*

Description

This calculates right and left singular vectors of a data matrix possibly containing missing values.

Usage

```
svdc(X, nu = NULL, nv = NULL)
```

Arguments

X	the data matrix of which to calcluate the completed SVD.
nu	the number of left singular vectors to calculate
nv	the nubmer of right singular vectors to calculate

Examples

```
Y <- rnorm(10)%*%t(rnorm(10))
Y[1,1] <- NA
svdc.out <- svdc(Y)
```

`winsor` *Winsorizes the data*

Description

Winsorizes the data

Usage

```
winsor(x, fraction = 0.01)
```

Arguments

`x`	the data.
`fraction`	the top and bottom quantiles to cap.

Examples

```
Y <- rlnorm(10)%*%t(rlnorm(10))
Yw <- winsor(Y,1E-2)
```

Index