Package: PlotNormTest (via r-universe)

May 24, 2026

Title Graphical Univariate/Multivariate Assessments for Normality Assumption

Version 1.0.2

Maintainer Huong Tran <quynhhuong5335@gmail.com>

Description Graphical methods testing multivariate normality assumption. Methods including assessing score function, and moment generating functions,independent transformations and linear transformations. For more details see Tran (2024),`` Contributions to Multivariate Data Science: Assessment and Identification of Multivariate Distributions and Supervised Learning for Groups of Objects." , PhD thesis, <https://our.oakland.edu/items/c8942577-2562-4d2f-8677-cb8ec0bf6234>.

License MIT + file LICENSE

Encoding UTF-8

Imports Rdpack, ggplot2, stats, Matrix, MatrixExtra, MASS, rlang

RdMacros Rdpack

Depends R (>= 3.5.0)

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

VignetteBuilder knitr

SysDataCompression xz

Repository https://huongtran53.r-universe.dev

Date/Publication 2026-01-22 02:55:49 UTC

RemoteUrl https://github.com/huongtran53/plotnormtest

RemoteRef HEAD

RemoteSha a364b677139bfdd6b0226f757029c0c50500b311

Contents

covLtLs 2
covTtTs 3
covZtZs 4
dCGF 5
dMGF 7
get_params 8
Independent_transformation 9
linear_transform 9
matrix_A 11
mt3_rev_pos 11
Multivariate_CGF_PLot 13
skewness_kurtosis 14
Univariate_CGF_plot 15
Univariate_Score_function 15

Index **18**

`covLtLs` *Linear combinations of distinct derivatives of empirical cumulant generating function (CGF).*

Description

Linear combination of third/fourth derivatives of CGF gives an asymptotically univariate Gaussian process with mean 0 and covariance between two points $t \in \mathbb{R}^p$ and $s \in \mathbb{R}^p$ is defined. We consider vector t and s as the form $t = t^* 1_p$ and $s = s^* 1_p$.

Usage

```
mt3_covLtLs(l, p, bigt = seq(-1, 1, 0.05)/sqrt(p), sTtTs = NULL, seed = 1)

mt4_covLtLs(l, p, bigt = seq(-1, 1, 0.05)/sqrt(p), sTtTs = NULL, seed = 1)
```

Arguments

`l`	vector of linear combination of size equal to the number of distinct derivatives, see `l_dhCGF()`.
`p`	dimension of multivariate random vector which data are collected.
`bigt`	array of value t^* and s^*.
`sTtTs`	Covariance matrix of derivatives vector, see `covTtTs()`. Default is `NULL`, when the algorithm will call `mt3_covTtTs()` or `mt4_covTtTs()`.
`seed`	Random seed to get the estimate of the supremum of the univariate Gaussian process obtained from the linear combination.

Value

- `sLtLs` covariance matrix of the linear combination of distinct derivatives, which is a zero-mean Gaussian process.
- `m.supLt` Monte-Carlo estimates of supremum of this Gaussian process

`mt3_covLtLs` returns values related to the use of third derivatives. `mt4_covLtLs` returns values related to the use of fourth derivatives.

Examples

```
bigt <- seq(-1, 1, .5)
p <- 2
# Third derivatives
lT3 <- l_dhCGF(p)[[1]]
l3 <- rep(1/sqrt(lT3), lT3)
mt3_covLtLs(l = l3, p = p, bigt = bigt/sqrt(p), seed = 1)
#fourth derivatives
lT4 <- l_dhCGF(p)[[2]]
l4 <- rep(1/sqrt(lT4), lT4)
mt4_covLtLs(l = l4, p = p, bigt = bigt/sqrt(p), seed = 1)
```

---

covTtTs *Covariance matrix of derivatives of sample cumulant generating function (CGF).*

Description

Stacking third/fourth derivatives of sample CGF together to obtain a vector, which (under normality assumption on data) approaches a normally distributed vector with zero mean and a covariance matrix. More specifically, `covTsTs` computes covariance between any two points as the form $t = t^* 1_p$ and $s = s^* 1_p$.

Usage

```
mt3_covTtTs(bigt, p = 1, pos.matrix = NULL)

mt4_covTtTs(bigt, p = 1, pos.matrix = NULL)
```

Arguments

`bigt`	array contains value of t^*.
`p`	dimension of multivariate random vector which data are collected.
`pos.matrix`	matrix containing information of position of any derivatives. Default is `NULL`, the function will call `mt3_pos()` or `mt4_pos()`.

Details

Number of distinct third derivatives is $l_{T_3} = p + 2 \times \binom{p}{2} + \binom{p}{3}$ Number of distinct fourth derivatives is $l_{T_4} = p + 3 \times \binom{p}{2} + 3 \times \binom{p}{3} + \binom{p}{4}$ For each pairs of (t^*, s^*), `covTsTt` results a covariance matrix of size $l_{T_3} \times l_{T_3}$ or $l_{T_4} \times l_{T_4}$.

Value

A 2 dimensional upper triangular array, with size equals to length of `bigt`. Each element contains a covariance matrix of derivatives sequences between any two points $t = t^*1_p$ and $s = s^*1_p$. `mt3_covTsTt` returns the resulting third derivatives.

`mt4_covTsTt` returns the resulting forth derivatives.

Examples

```
bigt <- seq(-1, 1, .5)
p <- 2
# Third derivatives
mt3_pos.matrix <- mt3_pos(p)
sTsTt3 <- mt3_covTtTs(bigt = bigt, p = p, pos.matrix = mt3_pos.matrix)
dim(sTsTt3)
sTsTt3[1:5, 1:5]
# Fourth derivatives
mt4_pos.matrix <- mt4_pos(p)
sTsTt4 <- mt4_covTtTs(bigt = bigt, p = p, pos.matrix = mt4_pos.matrix)
dim(sTsTt4)
sTsTt4[1:5, 1:5]
```

`covZtZs` *Covariance matrix of derivatives of sample moment generating function (MGF).*

Description

Stacking derivatives upto the third/fourth orders of sample MGF together to obtain a vector, which (under normality assumption) approaches a multivariate normally distributed vector with zero mean and a covariance matrix. `covZtZs` calculates covariance between any two points t and s in $\mathbb{R}^p$.

Usage

```
mt3_covZtZs(t, s, pos.matrix = NULL)

mt4_covZtZs(t, s, pos.matrix = NULL)
```

Arguments

t, s	a vector of length p.
pos.matrix	matrix contains information of positions of derivatives. Default is NULL, where the function will call mt3_pos() or mt4_pos().

Value

mt3_covZtZs Covariance matrix relating to the use of third derivatives.

mt4_covZtZs Covariance matrix relating to the use of fourth derivatives. This also contains information on the third third derivatives mt3_covZtZs.

Examples

```
set.seed(1)
p <- 3
x <- MASS::mvrnorm(100, rep(0, p), diag(p))
t <- rep(0.2, p)
s <- rep(-.3, p)
# Using third derivatives
pos.matrix3 <- mt3_pos(p)
sZtZs3 <- mt3_covZtZs(t, s, pos.matrix = pos.matrix3)
dim(sZtZs3)
sZtZs3[1:5, 1:5]
# Using fourth derivatives
sZtZs4 <- mt4_covZtZs(t, s)
dim(sZtZs4)
sZtZs4[1:5, 1:5]
```

---

dCGF *Calculation of derivatives of empirical cumulant generating function (CGF).*

Description

Get the third/fortth derivatives of sample CGF at a given point.

Usage

```
d3hCGF(myt, x)

d4hCGF(myt, x)

l_dhCGF(p)

dhCGF1D(t, x)
```

Arguments

myt, t	numeric vector of length p.
x	data matrix.
p	Dimension.

Details

Estimator of standardized cumulant function is

$$\log \hat{M}_X(t) = \log\left(\frac{1}{n}\sum_{i=1}^{n}\exp(t' S^{\frac{-1}{2}}(X_i - \bar{X}))\right)$$

and its

$$k^{th}$$

order derivatives is defined as

$$T_k(t) = \frac{\partial^k}{\partial t_{j_1} t_{j_2} \dots t_{j_k}} \log(\hat{M}_X(t)), t \in \mathbb{R}^p$$

where $t_{j_1} t_{j_2} \dots t_{j_k}$ are the corresponding components of vector $t \in \mathbb{R}^p$.

Value

d3hCGF returns the sequence of third derivatives of empirical CGF, ordered by index of $j_1 \leq j_2 \leq j_3 \leq p$.

d4hCGF returns the sequence of fourth derivatives of empirical CGF ordered by index of $j_1 \leq j_2 \leq j_3 \leq j_4 \leq p$.

l_dhCGF returns number of distinct third and fourth derivatives.

dhCGF1D returns third/fourth derivatives of univariate empirical CGF, which are d3hCGF and d4hCGF when $p = 1$.

Examples

```
p <- 3
# Number of distinct derivatives
l_dhCGF(p)
set.seed(1)
x <- MASS::mvrnorm(100, rep(0, p), diag(p))
myt <- rep(.2, p)
d3hCGF(myt = myt, x = x)
d4hCGF(myt = myt, x = x)
#Univariate data
set.seed(1)
x <- rnorm(100)
t <- .3
dhCGF1D(t, x)
```

dMGF *Moment generating functions (MGF) of standard normal distribution.*

Description

Get the polynomial term in the expression of derivatives of moment generating function of $N_p(0, I_p)$, with respect to a given component and its exponent. Up to eighth order.

Usage

```
dMGF(tab, t, coef = TRUE)
```

Arguments

tab	a dataframe with the first column contain indices of components of a multivariate random vector $\boldsymbol{X}$, and the second column is the order derivatives with respect to that components.
t	vector in $\mathbb{R}^p$.
coef	take TRUE or FALSE value to obtain only polynomial or whole expression by multiplying the polynomial term with the exponent term $\exp(.5t't)$.

Details

For a standard multivariate normal random variables $Y \sim N_p(0, I_p)$

$$\mathbb{E}\left(Y_1^{k_1}...Y_p^{k_p}\exp(t'X)\right) = \frac{\partial^{k_1}\dots\partial^{k_p}}{t_1^{k_1}\dots t_p^{k_p}}\exp(t't/2) = \mu^{(k_1)}(t_1)...\mu^{(k_p)}(t_p)\exp(t't/2)$$

For example, $\mathbb{E}Y_2^4\exp(t'Y) = \frac{\partial^4}{\partial t_2^4}\exp(t't/2) = \mu^{(4)}(t_2)\exp(t't/2)$.

Value

Value of derivatives.

Examples

```
#Calculation of above example
t <- rep(.2, 7)
tab <- data.frame(j = 2, exponent = 4)
dMGF(tab, t = t)
dMGF(tab, t = t, coef = FALSE)
```

get_params *Get parameters for plots derivatives of multivariate CGF to assess normality assumption.*

Description

Obtain necessary parameters to build a graphical test using the third/fourth derivatives of cumulant generating function.

Usage

```
mt3_get_param(p, bigt = seq(-1, 1, by = 0.05)/sqrt(p), l = NULL)

mt4_get_param(p, bigt = seq(-1, 1, by = 0.05)/sqrt(p), l = NULL)
```

Arguments

p	Dimension.
bigt	Array containing value of t^*.
l	Linear transformation of vector of third/fourth distinct derivatives, default is their average.

Value

- p Dimension.
- lT Number of distinct third/fourth order derivatives.
- sTtTs Two dimensional array, each element contains covariance matrix of vector of derivatives, the function called mt3_covTtTs(), or mt4_covTtTs().
- l.sTtTs Covariance matrix of linear combination of distinct derivatives, the function called mt3_covLtLs(), or mt4_covLtLs().
- m.supLT The Monte Carlo estimate of expected value supremum of the Gaussian process, see covLtLs().

mt3_get_param returns necessary parameters for the 2D plot relying on third derivatives. mt4_get_param returns necessary parameters for the 2D plot relying on fourth derivatives.

See Also

covZtZs(), covLtLs(), covTtTs()

Examples

```
p <- 2
mt3 <- mt3_get_param(p, bigt = seq(-1, 1, .5)/sqrt(p))
names(mt3)
mt4 <- mt4_get_param(p, bigt = seq(-1, 1, .5)/sqrt(p))
names(mt4)
```

Independent_transformation *Transformation to Independent Univariate Sample*

Description

Leave-one-out method gives approximately independent sample of standard multivariate normal distribution, which then produces sample of standard univariate normal distribution.

Usage

```
Multi.to.Uni(x)
```

Arguments

x	multivariate data matrix

Details

Let $\bar{X}_{-k}$ and S_{-k} are the sample mean sample variance covariance matrix obtained by using all but k^{th} data point. Then $S_{-k}^{-1/2}(X_k - \bar{X}_{-k}), k = 1, ...n$ are approximately independently distributed as $N_p(0, I)$. Thus all $n \times p$ entries in the data matrix so constructed can be treated as univariate samples of size $n \times p$ from $N(0, 1)$.

Value

Data frame contains univariate data and the index from multivariate data.

Examples

```
set.seed(1)
x <- MASS::mvrnorm(100, mu = rep(0, 5), diag(5))
df <- Multi.to.Uni(x)
qqnorm(df$x.new); abline(0, 1)
```

linear_transform *Best Linear Transformations*

Description

The algorithm uses gradient descent algorithm to obtain the maximum of the square of sample skewness, of the kurtosis or of their average under any univariate linear transformation of the multivariate data.

Usage

```
linear_transform(
  x,
  l0 = rep(1, ncol(x)),
  method = "both",
  epsilon = 1e-10,
  iter = 5000,
  stepsize = 0.001
)
```

Arguments

x	multivariate data matrix.
l0	starting point for projection algorithm, default is rep(1, ncol(x)).
method	character strings, one of c("skewness", "kurtosis", "both").
epsilon	bounds on error of optimal solution, default is 1e-10.
iter	number of iteration of projection algorithm, default is 5000.
stepsize	gradient descent stepsize, default is .001.

Value

- max_result: The maximum value after linear transformation.
- x_uni: Univariate data after transformation.
- vector_k: Vector of the "best" linear transformation.
- error: Error of projection algorithm.
- iteration: Number of iteration.

See Also

skewness(), kurtosis()

Examples

```
set.seed(1)
x <- MASS::mvrnorm(100, mu = rep(0, 2), diag(2))
linear_transform(x, method = "skewness")$max_result
linear_transform(x, method = "kurtosis")$max_result
linear_transform(x, method = "both")$max_result
```

matrix_A *From derivatives of MGF to derivatives of CGF.*

Description

Taylor expansion implies that vectors of derivatives of $\log(\hat{M}_X(t))$ can be approximated by a linear combination of vectors of derivatives of $\hat{M}_X(t)$. `matrix_A` results the corresponding linear combinations.

Usage

```
mt3_matrix_A(t)

mt4_matrix_A(t)
```

Arguments

`t`	vector of $\mathbb{R}^p$

Value

`mt3_matrix_A` returns coefficient matrix relating to the use of third derivatives.

`mt4_matrix_A` returns coefficient matrix relating to the use of fourth derivatives.

Examples

```
p <- 3
t <- rep(.2, p)
A3 <- mt3_matrix_A(t)
dim(A3)
A3[1:5, 1:5]
A4 <- mt4_matrix_A(t)
dim(A4)
A4[1:5, 1:5]
```

mt3_rev_pos *Derivatives of empirical moment generating function (MGF).*

Description

Given dimension p, returns a dataframe containing the position of all derivatives of estimator of moment generating function $\hat{M}_X(t)$, upto third/fourth order.

Usage

```
mt3_rev_pos(j1, j2, j3, p)

mt3_pos(p)

mt4_pos(p)
```

Arguments

j1	Index of the first variables
j2	Index of the first variables, should be at least j1
j3	Index of the first variables, should be at least j2
p	Dimension

Details

The estimator of multivariate moment generating function is $\hat{M}_X(t) = \frac{1}{n}\sum_{i=1}^{n}\exp(t'X_i)$ The chain containing all derivatives up to the third order is

$$Z = \left(\hat{M}, \hat{M}^{001}, \ldots \hat{M}^{00p}, \hat{M}^{011}, \hat{M}^{012}, \ldots \hat{M}^{0pp}, \hat{M}^{111}, \hat{M}^{112}, \ldots \hat{M}^{ppp}\right)'$$

and

$$\hat{M} = \hat{M}^{000}(t) = \hat{M}_X(t)$$

$$\hat{M}^{j_1 j_2 j_3}(t) = \frac{\partial^k}{\partial t_{j_1} t_{j_2} t_{j_3}} \hat{M}(t)$$

where k is the number of j_1, j_2, j_3 different from 0. Similar notation is applied when fourth derivatives is used.

Value

mt3_rev_pos returns the position of this particular derivative in the chain of all derivatives, up to third order.

mt3_pos an array contaning all position with respect to index of j_1, j_2, j_3.

mt4_pos an array contaning all position with respect to the index of j_1, j_2, j_3, j_4.

Examples

```
mt3_rev_pos(1, 2, 2, p = 3)
p <- 3
mt3_pos(p)
mt4_pos(p)
```

`Multivariate_CGF_PLot` *Graphical plots to assess multivariate normality assumption.*

Description

Cumulant generating functions of normally distributed random variables has derivatives of order higher than 3 are all 0. Hence, plots of empirical third/fourth order derivatives with large value or high slope gives indication of non-normality. `Multivariate_CGF_PLot` estimates and provides confidence region for average (or any linear combination) of third/fourth derivatives of empirical cumulant function at the points $t = t^* 1_p$. Plots for $p = 2, 3, \ldots, 10$ will be faster to obtain, as confidence regions and other necessary parameters are available in `mt3_lst_param.rda` and `mt4_lst_param.rda`. Higher dimension requires expensive computational cost.

Usage

```
d3hCGF_plot(x, alpha = 0.05)

d4hCGF_plot(x, alpha = 0.05)
```

Arguments

`x`	Data matrix of size $n \times p$
`alpha`	Significant level (default is $.05$)

Value

`d3hCGF_plot` returns plot relying in third derivatives.

`d4hCGF_plot` returns plot relying in forth derivatives.

See Also

`dhCGF_plot1D()`

Examples

```
set.seed(1234)
p <- 3
x <- MASS::mvrnorm(500, rep(0, p), diag(p))
d3hCGF_plot(x)
d4hCGF_plot(x)
```

skewness_kurtosis *Sample skewness and Sample Kurtosis.*

Description

Sample skewness and Sample Kurtosis.

Usage

```
kurtosis(x)

skewness(x)
```

Arguments

x	univariate data sample

Details

Sample kurtosis is

$$\hat{\kappa}_4 = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{X_i - \bar{X}}{S}\right)^4.$$

Sample skewness is

$$\hat{\kappa}_3 = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{X_i - \bar{X}}{S}\right)^3.$$

Value

`kurtosis` returns sample kurtosis.

`skewness` returns sample skewness.

Examples

```
set.seed(123)
y <- rnorm(100)
kurtosis(y)
set.seed(123)
x <- rnorm(100)
skewness(x)
```

`Univariate_CGF_plot` *Graphical plots to assess multivariate univarite assumption of data.*

Description

Plots the empirical third/fourth derivatives of cumulant generating function together with confidence probability region. Indication of non-normality is either violation of probability bands or curves with high slope.

Usage

```
dhCGF_plot1D(x, alpha = 0.05, method)
```

Arguments

`x`	Univariate data
`alpha`	Significant level (default is .05)
`method`	string, `"T3"` used the third derivatives, and `"T4"` uses the fourth derivatives.

Value

Plots

References

Ghosh S (1996). "A New Graphical Tool to Detect Non-Normality." *Journal of the Royal Statistical Society: Series B (Methodological)*, **58**(4), 691-702. doi:10.1111/j.25176161.1996.tb02108.x.

Examples

```
set.seed(123)
x <- rnorm(100)
dhCGF_plot1D(x, method = "T3")
dhCGF_plot1D(x, method = "T4")
```

`Univariate_Score_function`

Graphical plots to assess the univarite noramality assumption of data.

Description

Score function of a univariate normal distribution is a straight line. A non-linear graph of score function estimator shows evidence of non-normality.

Outliers are detected using the 2-sigma bands method.

Usage

```
cox(x, P = NULL, lambda = NULL, x.dist = NULL)

score_plot1D(x, P = NULL, lambda = NULL, x.dist = NULL, ori.index = NULL)
```

Arguments

x	univariate data.
P	vector of weight.
lambda	smoothing parameter, default is 0.5.
x.dist	the minimum distance between two data points in vector x.
ori.index	original index of vector x, default is NULL when index is just the order.

Details

To avoid the singularity of coefficient matrices in spline method, points with distance less than x.dist are merged and weight of the representative points is updated by the summation of weight of discarded points.

Under null hypothesis, a unbiased estimator score function of a given data point x_k is

$$\hat{\psi}(x_k) = \frac{n-4}{n-2}\frac{x_k - \bar{X}_{-k}}{S^2_{-k}}$$

and if a_k is the estimate score from function cox at the point x_k, then

$$a_k \in \hat{\psi}(x_k) \pm 2\sqrt{\hat{\text{Var}}(\hat{\psi}(x_k))}.$$

Hence points outside the 2-sigma bands are outliers.

Value

cox returns the estimate of score function.

- x: The updated univariate data if merging happens.
- a: Score value estimated at x.
- P: Updated weight (if merging happens).
- slt: Index of merged data point (is NULL if x.dist = NULL).

score_plot1D returns score functions together with 2-sigma bands for outlier detection.

- plot: plot of estimate score function and its band.
- outlier: index of outliers.

References

Ng PT (1994). "Smoothing Spline Score Estimation." *SIAM Journal on Scientific Computing*, **15**(5), 1003-1025. doi:10.1137/0915061. https://doi.org/10.1137/0915061.

Examples

```
set.seed(1)
x <- rnorm(100, 2, 4)
re <- cox(sort(x))
plot(re$x, re$a, xlab = "x", ylab = "Estimated Score",
 main = "Estimator of score function")
abline(0, 1)

set.seed(1)
x <- rnorm(100, 2, 4)
score_plot1D(sort(x))
```

Index

covLtLs, 2
covLtLs(), *8*
covTtTs, 3
covTtTs(), *2*, *8*
covZtZs, 4
covZtZs(), *8*
cox *(*Univariate_Score_function*)*, 15

d3hCGF *(*dCGF*)*, 5
d3hCGF_plot *(*Multivariate_CGF_PLot*)*, 13
d4hCGF *(*dCGF*)*, 5
d4hCGF_plot *(*Multivariate_CGF_PLot*)*, 13
dCGF, 5
dhCGF1D *(*dCGF*)*, 5
dhCGF_plot1D *(*Univariate_CGF_plot*)*, 15
dhCGF_plot1D(), *13*
dMGF, 7

get_params, 8

Independent_transformation, 9

kurtosis *(*skewness_kurtosis*)*, 14
kurtosis(), *10*

l_dhCGF *(*dCGF*)*, 5
l_dhCGF(), *2*
linear_transform, 9

matrix_A, 11
mt3_covLtLs *(*covLtLs*)*, 2
mt3_covLtLs(), *8*
mt3_covTtTs *(*covTtTs*)*, 3
mt3_covTtTs(), *2*, *8*
mt3_covZtZs *(*covZtZs*)*, 4
mt3_get_param *(*get_params*)*, 8
mt3_matrix_A *(*matrix_A*)*, 11
mt3_pos *(*mt3_rev_pos*)*, 11
mt3_pos(), *3*, *5*
mt3_rev_pos, 11
mt4_covLtLs *(*covLtLs*)*, 2
mt4_covLtLs(), *8*
mt4_covTtTs *(*covTtTs*)*, 3
mt4_covTtTs(), *2*, *8*
mt4_covZtZs *(*covZtZs*)*, 4
mt4_get_param *(*get_params*)*, 8
mt4_matrix_A *(*matrix_A*)*, 11
mt4_pos *(*mt3_rev_pos*)*, 11
mt4_pos(), *3*, *5*
Multi.to.Uni
*(*Independent_transformation*)*, 9
Multivariate_CGF_PLot, 13

score_plot1D
*(*Univariate_Score_function*)*, 15
skewness *(*skewness_kurtosis*)*, 14
skewness(), *10*
skewness_kurtosis, 14

Univariate_CGF_plot, 15
Univariate_Score_function, 15