

Package ‘energy’

May 8, 2026

Title E-Statistics: Multivariate Inference via the Energy of Data

Version 1.7-12

Date 2024-08-22

Description E-statistics (energy) tests and statistics for multivariate and univariate inference, including distance correlation, one-sample, two-sample, and multi-sample tests for comparing multivariate distributions, are implemented. Measuring and testing multivariate independence based on distance correlation, partial distance correlation, multivariate goodness-of-fit tests, k-groups and hierarchical clustering based on energy distance, testing for multivariate normality, distance components (disco) for non-parametric analysis of structured data, and other energy statistics/methods are implemented.

Imports Rcpp (>= 0.12.6), stats, boot, gsl

LinkingTo Rcpp

Suggests MASS, CompQuadForm, knitr, rmarkdown

Depends R (>= 3.1)

URL <https://github.com/mariarizzo/energy>

License GPL (>= 2)

LazyData true

NeedsCompilation yes

Repository CRAN

RoxygenNote 7.2.3

Author Maria Rizzo [aut, cre],
Gabor Szekely [aut]

Maintainer Maria Rizzo <mrizzo@bgsu.edu>

Date/Publication 2024-08-24 00:10:02 UTC

Contents

energy-package	2
centering distance matrices	3
dcor.ttest	4

dcorT	5
dcov.test	7
dcov2d	9
dcovU_stats	11
disco	12
distance correlation	14
Distance Matrix	17
edist	18
energy-deprecated	20
energy.hclust	20
eqdist.etest	23
EVnormal	25
indep.test	26
kgroups	28
mutual independence	30
mvI.test	32
mvnorm.test	34
normal.test	36
pdcor	37
Poisson Tests	39
sortrank	41
Unbiased distance covariance	42
U_product	44
Index	45

Description

Description: E-statistics (energy) tests and statistics for multivariate and univariate inference, including distance correlation, one-sample, two-sample, and multi-sample tests for comparing multivariate distributions, are implemented. Measuring and testing multivariate independence based on distance correlation, partial distance correlation, multivariate goodness-of-fit tests, clustering based on energy distance, testing for multivariate normality, distance components (disco) for non-parametric analysis of structured data, and other energy statistics/methods are implemented.

Author(s)

Maria L. Rizzo and Gabor J. Szekely

References

- G. J. Szekely and M. L. Rizzo (2013). Energy statistics: A class of statistics based on distances, *Journal of Statistical Planning and Inference*.
- M. L. Rizzo and G. J. Szekely (2016). Energy Distance, *WIREs Computational Statistics*, Wiley, Volume 8 Issue 1, 27-38. Available online Dec., 2015, doi:10.1002/wics.1375.
- G. J. Szekely and M. L. Rizzo (2017). The Energy of Data. *The Annual Review of Statistics and Its Application* 4:447-79.
- G. J. Szekely and M. L. Rizzo (2023). *The Energy of Data and Distance Correlation*. Chapman & Hall/CRC Monographs on Statistics and Applied Probability. ISBN 9781482242744. <https://www.routledge.com/The-Energy-of-Data-and-Distance-Correlation/Szekely-Rizzo/p/book/9781482242744>.

centering distance matrices

Double centering and U-centering

Description

Stand-alone double centering and U-centering functions that are applied in unbiased distance covariance, bias corrected distance correlation, and partial distance correlation.

Usage

```
Dcenter(x)
Ucenter(x)
U_center(Dx)
D_center(Dx)
```

Arguments

x	dist object or data matrix
Dx	distance or dissimilarity matrix

Details

In `Dcenter` and `Ucenter`, `x` must be a dist object or a data matrix. Both functions return a doubly centered distance matrix.

Note that `pdcor`, etc. functions include the centering operations (in `C`), so that these stand alone versions of centering functions are not needed except in case one wants to compute just a double-centered or U-centered matrix.

`U_center` is the Rcpp export of the `cpp` function. `D_center` is the Rcpp export of the `cpp` function.

Value

All functions return a square symmetric matrix.

Dcenter returns a matrix

$$A_{ij} = a_{ij} - \bar{a}_{i.} - \bar{a}_{.j} + \bar{a}_{..}$$

as in classical multidimensional scaling. Ucenter returns a matrix

$$\tilde{A}_{ij} = a_{ij} - \frac{a_{i.}}{n-2} - \frac{a_{.j}}{n-2} + \frac{a_{..}}{(n-1)(n-2)}, \quad i \neq j,$$

with zero diagonal, and this is the double centering applied in pdcov and pdcor as well as the unbiased dCov and bias corrected dCor statistics.

Note

The c++ versions D_center and U_center should typically be faster. R versions are retained for historical reasons.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G.J. and Rizzo, M.L. (2014), Partial Distance Correlation with Methods for Dissimilarities, *Annals of Statistics*, Vol. 42, No. 6, pp. 2382-2412.

Examples

```
x <- iris[1:10, 1:4]
dx <- dist(x)
Dx <- as.matrix(dx)
M <- U_center(Dx)

all.equal(M, U_center(M))      #idempotence
all.equal(M, D_center(M))     #invariance
```

dcor.ttest

Distance Correlation t-test for High Dimensions

Description

Defunct: use dcorT.test and dcorT.

Usage

```
dcor.t(x, y, distance = FALSE)
dcor.ttest(x, y, distance = FALSE)
```

Arguments

x	data or distances of first sample
y	data or distances of second sample
distance	TRUE if x and y are distances, otherwise FALSE

Details

See [dcorT](#).

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

dcorT

Distance Correlation t-Test

Description

Distance correlation t-test of multivariate independence for high dimension.

Usage

```
dcorT.test(x, y)
dcorT(x, y)
```

Arguments

x	data or distances of first sample
y	data or distances of second sample

Details

`dcorT.test` performs a nonparametric t-test of multivariate independence in high dimension (dimension is close to or larger than sample size). As dimension goes to infinity, the asymptotic distribution of the test statistic is approximately Student t with $n(n-3)/2 - 1$ degrees of freedom and for $n \geq 10$ the statistic is approximately distributed as standard normal.

The sample sizes (number of rows) of the two samples must agree, and samples must not contain missing values.

The t statistic (`dcorT`) is a transformation of a bias corrected version of distance correlation (see SR 2013 for details).

Large values (upper tail) of the `dcorT` statistic are significant.

Value

dcorT returns the dcor t statistic, and dcorT.test returns a list with class htest containing

method	description of test
statistic	observed value of the test statistic
parameter	degrees of freedom
estimate	(bias corrected) squared dCor(x,y)
p.value	p-value of the t-test
data.name	description of data

Note

dcor.t and dcor.ttest are deprecated.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G.J. and Rizzo, M.L. (2013). The distance correlation t-test of independence in high dimension. *Journal of Multivariate Analysis*, Volume 117, pp. 193-213.
[doi:10.1016/j.jmva.2013.02.012](https://doi.org/10.1016/j.jmva.2013.02.012)

Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.
[doi:10.1214/009053607000000505](https://doi.org/10.1214/009053607000000505)

Szekely, G.J. and Rizzo, M.L. (2009), Brownian Distance Covariance, *Annals of Applied Statistics*, Vol. 3, No. 4, 1236-1265.
[doi:10.1214/09AOAS312](https://doi.org/10.1214/09AOAS312)

See Also

[bcdcor](#) [dcov.test](#) [dcor](#) [DCOR](#)

Examples

```
x <- matrix(rnorm(100), 10, 10)
y <- matrix(runif(100), 10, 10)
dcorT(x, y)
dcorT.test(x, y)
```

dcov.test

Distance Covariance Test and Distance Correlation test

Description

Distance covariance test and distance correlation test of multivariate independence. Distance covariance and distance correlation are multivariate measures of dependence.

Usage

```
dcov.test(x, y, index = 1.0, R = NULL)
dcor.test(x, y, index = 1.0, R)
```

Arguments

x	data or distances of first sample
y	data or distances of second sample
R	number of replicates
index	exponent on Euclidean distance, in (0,2]

Details

dcov.test and dcor.test are nonparametric tests of multivariate independence. The test decision is obtained via permutation bootstrap, with R replicates.

The sample sizes (number of rows) of the two samples must agree, and samples must not contain missing values.

The index is an optional exponent on Euclidean distance. Valid exponents for energy are in (0, 2) excluding 2.

Argument types supported are numeric data matrix, data.frame, or tibble, with observations in rows; numeric vector; ordered or unordered factors. In case of unordered factors a 0-1 distance matrix is computed.

Optionally pre-computed distances can be input as class "dist" objects or as distance matrices. For data types of arguments, distance matrices are computed internally.

The dcov test statistic is $n\mathcal{V}_n^2$ where $\mathcal{V}_n(x, y) = \text{dcov}(x, y)$, which is based on interpoint Euclidean distances $\|x_i - x_j\|$. The index is an optional exponent on Euclidean distance.

Similarly, the dcor test statistic is based on the normalized coefficient, the distance correlation. (See the manual page for dcor.)

Distance correlation is a new measure of dependence between random vectors introduced by Szekely, Rizzo, and Bakirov (2007). For all distributions with finite first moments, distance correlation \mathcal{R} generalizes the idea of correlation in two fundamental ways:

- (1) $\mathcal{R}(X, Y)$ is defined for X and Y in arbitrary dimension.
- (2) $\mathcal{R}(X, Y) = 0$ characterizes independence of X and Y .

Characterization (2) also holds for powers of Euclidean distance $\|x_i - x_j\|^s$, where $0 < s < 2$, but (2) does not hold when $s = 2$.

Distance correlation satisfies $0 \leq \mathcal{R} \leq 1$, and $\mathcal{R} = 0$ only if X and Y are independent. Distance covariance \mathcal{V} provides a new approach to the problem of testing the joint independence of random vectors. The formal definitions of the population coefficients \mathcal{V} and \mathcal{R} are given in (SRB 2007). The definitions of the empirical coefficients are given in the energy `dcov` topic.

For all values of the index in $(0,2)$, under independence the asymptotic distribution of $n\mathcal{V}_n^2$ is a quadratic form of centered Gaussian random variables, with coefficients that depend on the distributions of X and Y . For the general problem of testing independence when the distributions of X and Y are unknown, the test based on $n\mathcal{V}_n^2$ can be implemented as a permutation test. See (SRB 2007) for theoretical properties of the test, including statistical consistency.

Value

`dcov.test` or `dcor.test` returns a list with class `htest` containing

<code>method</code>	description of test
<code>statistic</code>	observed value of the test statistic
<code>estimate</code>	<code>dCov(x,y)</code> or <code>dCor(x,y)</code>
<code>estimates</code>	a vector: [<code>dCov(x,y)</code> , <code>dCor(x,y)</code> , <code>dVar(x)</code> , <code>dVar(y)</code>]
<code>condition</code>	logical, permutation test applied
<code>replicates</code>	replicates of the test statistic
<code>p.value</code>	approximate p-value of the test
<code>n</code>	sample size
<code>data.name</code>	description of data

Note

For the `dcov` test of independence, the distance covariance test statistic is the V-statistic $n \text{dCov}^2 = n\mathcal{V}_n^2$ (not `dCov`).

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

- Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.
[doi:10.1214/009053607000000505](https://doi.org/10.1214/009053607000000505)
- Szekely, G.J. and Rizzo, M.L. (2009), Brownian Distance Covariance, *Annals of Applied Statistics*, Vol. 3, No. 4, 1236-1265.
[doi:10.1214/09AOAS312](https://doi.org/10.1214/09AOAS312)
- Szekely, G.J. and Rizzo, M.L. (2009), Rejoinder: Brownian Distance Covariance, *Annals of Applied Statistics*, Vol. 3, No. 4, 1303-1308.

See Also

[dcov](#) [dcor](#) [pdcov.test](#) [pdcor.test](#) [dcor.ttest](#)

Examples

```
x <- iris[1:50, 1:4]
y <- iris[51:100, 1:4]
set.seed(1)
dcor.test(dist(x), dist(y), R=199)
set.seed(1)
dcov.test(x, y, R=199)
```

dcov2d

Fast dCor and dCov for bivariate data only

Description

For bivariate data only, these are fast $O(n \log n)$ implementations of distance correlation and distance covariance statistics. The U-statistic for dcov^2 is unbiased; the V-statistic is the original definition in SRB 2007. These algorithms do not store the distance matrices, so they are suitable for large samples.

Usage

```
dcor2d(x, y, type = c("V", "U"))
dcov2d(x, y, type = c("V", "U"), all.stats = FALSE)
```

Arguments

x	numeric vector
y	numeric vector
type	"V" or "U", for V- or U-statistics
all.stats	logical

Details

The unbiased (squared) dcov is documented in `dcovU`, for multivariate data in arbitrary, not necessarily equal dimensions. `dcov2d` and `dcor2d` provide a faster $O(n \log n)$ algorithm for bivariate (x, y) only (X and Y are real-valued random vectors). The $O(n \log n)$ algorithm was proposed by Huo and Szekely (2016). The algorithm is faster above a certain sample size n . It does not store the distance matrix so the sample size can be very large.

Value

By default, `dcov2d` returns the V-statistic $V_n = dCov_n^2(x, y)$, and if `type="U"`, it returns the U-statistic, unbiased for $dCov^2(X, Y)$. The argument `all.stats=TRUE` is used internally when the function is called from `dcor2d`.

By default, `dcor2d` returns $dCor_n^2(x, y)$, and if `type="U"`, it returns a bias-corrected estimator of squared `dcor` equivalent to `bcdcor`.

These functions do not store the distance matrices so they are helpful when sample size is large and the data is bivariate.

Note

The U-statistic U_n can be negative in the lower tail so the square root of the U-statistic is not applied. Similarly, `dcor2d(x, y, "U")` is bias-corrected and can be negative in the lower tail, so we do not take the square root. The original definitions of `dCov` and `dCor` (SRB2007, SR2009) were based on V-statistics, which are non-negative, and defined using the square root of V-statistics.

It has been suggested that instead of taking the square root of the U-statistic, one could take the root of $|U_n|$ before applying the sign, but that introduces more bias than the original `dCor`, and should never be used.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Huo, X. and Szekely, G.J. (2016). Fast computing for distance covariance. *Technometrics*, 58(4), 435-447.

Szekely, G.J. and Rizzo, M.L. (2014), Partial Distance Correlation with Methods for Dissimilarities. *Annals of Statistics*, Vol. 42 No. 6, 2382-2412.

Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.

[doi:10.1214/009053607000000505](https://doi.org/10.1214/009053607000000505)

See Also

[dcov](#) [dcov.test](#) [dcor](#) [dcor.test](#) (multivariate statistics and permutation test)

Examples

```
## these are equivalent, but 2d is faster for n > 50
n <- 100
x <- rnorm(100)
y <- rnorm(100)
all.equal(dcov(x, y)^2, dcov2d(x, y), check.attributes = FALSE)
all.equal(bcdcor(x, y), dcor2d(x, y, "U"), check.attributes = FALSE)

x <- rlnorm(400)
y <- rexp(400)
```

```
dcov.test(x, y, R=199) #permutation test
dcor.test(x, y, R=199)
```

dcovU_stats *Unbiased distance covariance statistics*

Description

This function computes unbiased estimators of squared distance covariance, distance variance, and a bias-corrected estimator of (squared) distance correlation.

Usage

```
dcovU_stats(Dx, Dy)
```

Arguments

Dx	distance matrix of first sample
Dy	distance matrix of second sample

Details

The unbiased (squared) dcov is inner product definition of dCov, in the Hilbert space of U-centered distance matrices.

The sample sizes (number of rows) of the two samples must agree, and samples must not contain missing values. The arguments must be square symmetric matrices.

Value

dcovU_stats returns a vector of the components of bias-corrected dcor: [dCovU, bdcor, dVarXU, dVarYU].

Note

Unbiased distance covariance (SR2014) corresponds to the biased (original) dCov². Since dcovU is an unbiased statistic, it is signed and we do not take the square root. For the original distance covariance test of independence (SRB2007, SR2009), the distance covariance test statistic is the V-statistic $n \text{dCov}^2 = n \mathcal{V}_n^2$ (not dCov). Similarly, bdcor is bias-corrected, so we do not take the square root as with dCor.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Székely

References

Szekely, G.J. and Rizzo, M.L. (2014), Partial Distance Correlation with Methods for Dissimilarities, *Annals of Statistics*, Vol. 42 No. 6, 2382-2412.

Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.

[doi:10.1214/009053607000000505](https://doi.org/10.1214/009053607000000505)

Szekely, G.J. and Rizzo, M.L. (2009), Brownian Distance Covariance, *Annals of Applied Statistics*, Vol. 3, No. 4, 1236-1265.

[doi:10.1214/09AOAS312](https://doi.org/10.1214/09AOAS312)

Examples

```
x <- iris[1:50, 1:4]
y <- iris[51:100, 1:4]
Dx <- as.matrix(dist(x))
Dy <- as.matrix(dist(y))
dcovU_stats(Dx, Dy)
```

disco

distance components (DISCO)

Description

E-statistics DISTance COmponents and tests, analogous to variance components and anova.

Usage

```
disco(x, factors, distance, index=1.0, R, method=c("disco", "discoB", "discoF"))
disco.between(x, factors, distance, index=1.0, R)
```

Arguments

x	data matrix or distance matrix or dist object
factors	matrix or data frame of factor labels or integers (not design matrix)
distance	logical, TRUE if x is distance matrix
index	exponent on Euclidean distance in (0,2]
R	number of replicates for a permutation test
method	test statistic

Details

disco calculates the distance components decomposition of total dispersion and if $R > 0$ tests for significance using the test statistic disco "F" ratio (default method="disco"), or using the between component statistic (method="discoB"), each implemented by permutation test.

If x is a dist object, argument distance is ignored. If x is a distance matrix, set distance=TRUE.

In the current release disco computes the decomposition for one-way models only.

Value

When `method="discoF"`, `disco` returns a list similar to the return value from `anova.lm`, and the `print.disco` method is provided to format the output into a similar table. Details:

`disco` returns a class `disco` object, which is a list containing

<code>call</code>	<code>call</code>
<code>method</code>	<code>method</code>
<code>statistic</code>	vector of observed statistics
<code>p.value</code>	vector of p-values
<code>k</code>	number of factors
<code>N</code>	number of observations
<code>between</code>	between-sample distance components
<code>within</code>	one-way within-sample distance components
<code>within</code>	within-sample distance component
<code>total</code>	total dispersion
<code>Df.trt</code>	degrees of freedom for treatments
<code>Df.e</code>	degrees of freedom for error
<code>index</code>	index (exponent on distance)
<code>factor.names</code>	factor names
<code>factor.levels</code>	factor levels
<code>sample.sizes</code>	sample sizes
<code>stats</code>	matrix containing decomposition

When `method="discoB"`, `disco` passes the arguments to `disco.between`, which returns a class `htest` object.

`disco.between` returns a class `htest` object, where the test statistic is the between-sample statistic (proportional to the numerator of the F ratio of the `disco` test).

Note

The current version does all calculations via matrix arithmetic and boot function. Support for more general additive models and a formula interface is under development.

`disco` methods have been added to the cluster distance summary function `edist`, and energy tests for equality of distribution (see `eqdist.etest`).

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

M. L. Rizzo and G. J. Szekely (2010). DISCO Analysis: A Nonparametric Extension of Analysis of Variance, *Annals of Applied Statistics*, Vol. 4, No. 2, 1034-1055.

[doi:10.1214/09AOAS245](https://doi.org/10.1214/09AOAS245)

See Also

[edist](#) [eqdist.e](#) [eqdist.etest](#) [ksample.e](#)

Examples

```
## warpbreaks one-way decompositions
data(warpbreaks)
attach(warpbreaks)
disco(breaks, factors=wool, R=99)

## warpbreaks two-way wool+tension
disco(breaks, factors=data.frame(wool, tension), R=0)

## warpbreaks two-way wool*tension
disco(breaks, factors=data.frame(wool, tension, wool:tension), R=0)

## When index=2 for univariate data, we get ANOVA decomposition
disco(breaks, factors=tension, index=2.0, R=99)
aov(breaks ~ tension)

## Multivariate response
## Example on producing plastic film from Krzanowski (1998, p. 381)
tear <- c(6.5, 6.2, 5.8, 6.5, 6.5, 6.9, 7.2, 6.9, 6.1, 6.3,
          6.7, 6.6, 7.2, 7.1, 6.8, 7.1, 7.0, 7.2, 7.5, 7.6)
gloss <- c(9.5, 9.9, 9.6, 9.6, 9.2, 9.1, 10.0, 9.9, 9.5, 9.4,
           9.1, 9.3, 8.3, 8.4, 8.5, 9.2, 8.8, 9.7, 10.1, 9.2)
opacity <- c(4.4, 6.4, 3.0, 4.1, 0.8, 5.7, 2.0, 3.9, 1.9, 5.7,
             2.8, 4.1, 3.8, 1.6, 3.4, 8.4, 5.2, 6.9, 2.7, 1.9)
Y <- cbind(tear, gloss, opacity)
rate <- factor(gl(2,10), labels=c("Low", "High"))

## test for equal distributions by rate
disco(Y, factors=rate, R=99)
disco(Y, factors=rate, R=99, method="discoB")

## Just extract the decomposition table
disco(Y, factors=rate, R=0)$stats

## Compare eqdist.e methods for rate
## disco between stat is half of original when sample sizes equal
eqdist.e(Y, sizes=c(10, 10), method="original")
eqdist.e(Y, sizes=c(10, 10), method="discoB")

## The between-sample distance component
disco.between(Y, factors=rate, R=0)
```

Description

Computes distance covariance and distance correlation statistics, which are multivariate measures of dependence.

Usage

```
dcov(x, y, index = 1.0)
dcor(x, y, index = 1.0)
```

Arguments

x	data or distances of first sample
y	data or distances of second sample
index	exponent on Euclidean distance, in (0,2]

Details

dcov and dcor compute distance covariance and distance correlation statistics.

The sample sizes (number of rows) of the two samples must agree, and samples must not contain missing values.

The `index` is an optional exponent on Euclidean distance. Valid exponents for energy are in (0, 2) excluding 2.

Argument types supported are numeric data matrix, data.frame, or tibble, with observations in rows; numeric vector; ordered or unordered factors. In case of unordered factors a 0-1 distance matrix is computed.

Optionally pre-computed distances can be input as class "dist" objects or as distance matrices. For data types of arguments, distance matrices are computed internally.

Distance correlation is a new measure of dependence between random vectors introduced by Szekely, Rizzo, and Bakirov (2007). For all distributions with finite first moments, distance correlation \mathcal{R} generalizes the idea of correlation in two fundamental ways: (1) $\mathcal{R}(X, Y)$ is defined for X and Y in arbitrary dimension. (2) $\mathcal{R}(X, Y) = 0$ characterizes independence of X and Y .

Distance correlation satisfies $0 \leq \mathcal{R} \leq 1$, and $\mathcal{R} = 0$ only if X and Y are independent. Distance covariance \mathcal{V} provides a new approach to the problem of testing the joint independence of random vectors. The formal definitions of the population coefficients \mathcal{V} and \mathcal{R} are given in (SRB 2007). The definitions of the empirical coefficients are as follows.

The empirical distance covariance $\mathcal{V}_n(\mathbf{X}, \mathbf{Y})$ with index 1 is the nonnegative number defined by

$$\mathcal{V}_n^2(\mathbf{X}, \mathbf{Y}) = \frac{1}{n^2} \sum_{k, l=1}^n A_{kl} B_{kl}$$

where A_{kl} and B_{kl} are

$$A_{kl} = a_{kl} - \bar{a}_{k.} - \bar{a}_{.l} + \bar{a}_{..}$$

$$B_{kl} = b_{kl} - \bar{b}_{k.} - \bar{b}_{.l} + \bar{b}_{..}$$

Here

$$a_{kl} = \|X_k - X_l\|_p, \quad b_{kl} = \|Y_k - Y_l\|_q, \quad k, l = 1, \dots, n,$$

and the subscript \cdot denotes that the mean is computed for the index that it replaces. Similarly, $\mathcal{V}_n(\mathbf{X})$ is the nonnegative number defined by

$$\mathcal{V}_n^2(\mathbf{X}) = \mathcal{V}_n^2(\mathbf{X}, \mathbf{X}) = \frac{1}{n^2} \sum_{k,l=1}^n A_{kl}^2.$$

The empirical distance correlation $\mathcal{R}_n(\mathbf{X}, \mathbf{Y})$ is the square root of

$$\mathcal{R}_n^2(\mathbf{X}, \mathbf{Y}) = \frac{\mathcal{V}_n^2(\mathbf{X}, \mathbf{Y})}{\sqrt{\mathcal{V}_n^2(\mathbf{X})\mathcal{V}_n^2(\mathbf{Y})}}.$$

See [dcov.test](#) for a test of multivariate independence based on the distance covariance statistic.

Value

dcov returns the sample distance covariance and dcor returns the sample distance correlation.

Note

Note that it is inefficient to compute dCor by:

square root of dcov(x, y)/sqrt(dcov(x, x)*dcov(y, y))

because the individual calls to dcov involve unnecessary repetition of calculations.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.

[doi:10.1214/009053607000000505](https://doi.org/10.1214/009053607000000505)

Szekely, G.J. and Rizzo, M.L. (2009), Brownian Distance Covariance, *Annals of Applied Statistics*, Vol. 3, No. 4, 1236-1265.

[doi:10.1214/09AOAS312](https://doi.org/10.1214/09AOAS312)

Szekely, G.J. and Rizzo, M.L. (2009), Rejoinder: Brownian Distance Covariance, *Annals of Applied Statistics*, Vol. 3, No. 4, 1303-1308.

See Also

[dcov2d](#) [dcor2d](#) [bcdcor](#) [dcovU](#) [pdcor](#) [dcov.test](#) [dcor.test](#) [pdcor.test](#)

Examples

```
x <- iris[1:50, 1:4]
y <- iris[51:100, 1:4]
dcov(x, y)
dcov(dist(x), dist(y)) #same thing
```

Description

Utilities for working with distance matrices. `is.dmatrix` is a utility that checks whether the argument is a distance or dissimilarity matrix; is it square symmetric, non-negative, with zero diagonal? `calc_dist` computes a distance matrix directly from a data matrix.

Usage

```
is.dmatrix(x, tol = 100 * .Machine$double.eps)
calc_dist(x)
```

Arguments

<code>x</code>	numeric matrix
<code>tol</code>	tolerance for checking required conditions

Details

Energy functions work with the distance matrices of samples. The `is.dmatrix` function is used internally when converting arguments to distance matrices. The default `tol` is the same as default tolerance of `isSymmetric`.

`calc_dist` is an exported Rcpp function that returns a Euclidean distance matrix from the input data matrix.

Value

`is.dmatrix` returns TRUE if (within tolerance) `x` is a distance/dissimilarity matrix; otherwise FALSE. It will return FALSE if `x` is a class `dist` object.

`calc_dist` returns the Euclidean distance matrix for the data matrix `x`, which has observations in rows.

Note

In practice, if `dist(x)` is not yet computed, `calc_dist(x)` will be faster than `as.matrix(dist(x))`.
On working with non-Euclidean dissimilarities, see the references.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu>

References

Szekely, G.J. and Rizzo, M.L. (2014), Partial Distance Correlation with Methods for Dissimilarities. *Annals of Statistics*, Vol. 42 No. 6, 2382-2412.

Examples

```
x <- matrix(rnorm(20), 10, 2)
D <- calc_dist(x)
is.dmatrix(D)
is.dmatrix(cov(x))
```

edist

E-distance

Description

Returns the E-distances (energy statistics) between clusters.

Usage

```
edist(x, sizes, distance = FALSE, ix = 1:sum(sizes), alpha = 1,
      method = c("cluster", "discoB"))
```

Arguments

x	data matrix of pooled sample or Euclidean distances
sizes	vector of sample sizes
distance	logical: if TRUE, x is a distance matrix
ix	a permutation of the row indices of x
alpha	distance exponent in (0,2]
method	how to weight the statistics

Details

A vector containing the pairwise two-sample multivariate \mathcal{E} -statistics for comparing clusters or samples is returned. The e-distance between clusters is computed from the original pooled data, stacked in matrix x where each row is a multivariate observation, or from the distance matrix x of the original data, or distance object returned by `dist`. The first `sizes[1]` rows of the original data matrix are the first sample, the next `sizes[2]` rows are the second sample, etc. The permutation vector `ix` may be used to obtain e-distances corresponding to a clustering solution at a given level in the hierarchy.

The default method `cluster` summarizes the e-distances between clusters in a table. The e-distance between two clusters C_i, C_j of size n_i, n_j proposed by Szekely and Rizzo (2005) is the e-distance $e(C_i, C_j)$, defined by

$$e(C_i, C_j) = \frac{n_i n_j}{n_i + n_j} [2M_{ij} - M_{ii} - M_{jj}],$$

where

$$M_{ij} = \frac{1}{n_i n_j} \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} \|X_{ip} - X_{jq}\|^\alpha,$$

$\| \cdot \|$ denotes Euclidean norm, $\alpha = \text{alpha}$, and X_{ip} denotes the p -th observation in the i -th cluster. The exponent α should be in the interval $(0,2]$.

The coefficient $\frac{n_i n_j}{n_i + n_j}$ is one-half of the harmonic mean of the sample sizes. The `discoB` method is related but with different ways of summarizing the pairwise differences between samples. The `disco` methods apply the coefficient $\frac{n_i n_j}{2N}$ where N is the total number of observations. This weights each (i,j) statistic by sample size relative to N . See the `disco` topic for more details.

Value

A object of class `dist` containing the lower triangle of the e-distance matrix of cluster distances corresponding to the permutation of indices `ix` is returned. The method attribute of the distance object is assigned a value of type, `index`.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G. J. and Rizzo, M. L. (2005) Hierarchical Clustering via Joint Between-Within Distances: Extending Ward's Minimum Variance Method, *Journal of Classification* 22(2) 151-183.

[doi:10.1007/s0035700500129](https://doi.org/10.1007/s0035700500129)

M. L. Rizzo and G. J. Szekely (2010). DISCO Analysis: A Nonparametric Extension of Analysis of Variance, *Annals of Applied Statistics*, Vol. 4, No. 2, 1034-1055.

[doi:10.1214/09AOAS245](https://doi.org/10.1214/09AOAS245)

Szekely, G. J. and Rizzo, M. L. (2004) Testing for Equal Distributions in High Dimension, *InterStat*, November (5).

Szekely, G. J. (2000) Technical Report 03-05, \mathcal{E} -statistics: Energy of Statistical Samples, Department of Mathematics and Statistics, Bowling Green State University.

See Also

[energy](#), [hclust](#), [eqdist](#), [etest](#), [ksample](#), [e](#), [disco](#)

Examples

```
## compute cluster e-distances for 3 samples of iris data
data(iris)
edist(iris[,1:4], c(50,50,50))

## pairwise disco statistics
edist(iris[,1:4], c(50,50,50), method="discoB")

## compute e-distances from a distance object
data(iris)
edist(dist(iris[,1:4]), c(50, 50, 50), distance=TRUE, alpha = 1)

## compute e-distances from a distance matrix
data(iris)
```

```
d <- as.matrix(dist(iris[,1:4]))
edist(d, c(50, 50, 50), distance=TRUE, alpha = 1)
```

energy-deprecated *Deprecated Functions*

Description

These deprecated functions have been replaced by revised functions and will be removed in future releases of the energy package.

Usage

```
DCOR(x, y, index=1.0)
```

Arguments

x	data or distances of first sample
y	data or distances of second sample
index	exponent on Euclidean distance in (0, 2)

Details

DCOR is an R version replaced by faster compiled code.

energy.hclust *Hierarchical Clustering by Minimum (Energy) E-distance*

Description

Performs hierarchical clustering by minimum (energy) E-distance method.

Usage

```
energy.hclust(dst, alpha = 1)
```

Arguments

dst	dist object
alpha	distance exponent

Details

Dissimilarities are $d(x, y) = \|x - y\|^\alpha$, where the exponent α is in the interval (0,2]. This function performs agglomerative hierarchical clustering. Initially, each of the n singletons is a cluster. At each of $n-1$ steps, the procedure merges the pair of clusters with minimum e-distance. The e-distance between two clusters C_i, C_j of sizes n_i, n_j is given by

$$e(C_i, C_j) = \frac{n_i n_j}{n_i + n_j} [2M_{ij} - M_{ii} - M_{jj}],$$

where

$$M_{ij} = \frac{1}{n_i n_j} \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} \|X_{ip} - X_{jq}\|^\alpha,$$

$\|\cdot\|$ denotes Euclidean norm, and X_{ip} denotes the p -th observation in the i -th cluster.

The return value is an object of class `hclust`, so `hclust` methods such as `print` or `plot` methods, `plclust`, and `cutree` are available. See the documentation for `hclust`.

The e-distance measures both the heterogeneity between clusters and the homogeneity within clusters. \mathcal{E} -clustering ($\alpha = 1$) is particularly effective in high dimension, and is more effective than some standard hierarchical methods when clusters have equal means (see example below). For other advantages see the references.

`edist` computes the energy distances for the result (or any partition) and returns the cluster distances in a `dist` object. See the `edist` examples.

Value

An object of class `hclust` which describes the tree produced by the clustering process. The object is a list with components:

<code>merge</code> :	an $n-1$ by 2 matrix, where row i of merge describes the merging of clusters at step i of the clustering. If an element j in the row is negative, then observation $-j$ was merged at this stage. If j is positive then the merge was with the cluster formed at the (earlier) stage j of the algorithm.
<code>height</code> :	the clustering height: a vector of $n-1$ non-decreasing real numbers (the e-distance between merging clusters)
<code>order</code> :	a vector giving a permutation of the indices of original observations suitable for plotting, in the sense that a cluster plot using this ordering and matrix merge will not have crossings of the branches.
<code>labels</code> :	labels for each of the objects being clustered.
<code>call</code> :	the call which produced the result.
<code>method</code> :	the cluster method that has been used (e-distance).
<code>dist.method</code> :	the distance that has been used to create <code>dst</code> .

Note

Currently `stats::hclust` implements Ward's method by `method="ward.D2"`, which applies the squared distances. That method was previously `"ward"`. Because both `hclust` and `energy` use the same type of Lance-Williams recursive formula to update cluster distances, now with the additional

option `method="ward.D"` in `hclust`, the energy distance method is easily implemented by `hclust`. (Some "Ward" algorithms do not use Lance-Williams, however). Energy clustering (with `alpha=1`) and "ward.D" now return the same result, except that the cluster heights of energy hierarchical clustering with `alpha=1` are two times the heights from `hclust`. In order to ensure compatibility with `hclust` methods, `energy.hclust` now passes arguments through to `hclust` after possibly applying the optional exponent to distance.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G. J. and Rizzo, M. L. (2005) Hierarchical Clustering via Joint Between-Within Distances: Extending Ward's Minimum Variance Method, *Journal of Classification* 22(2) 151-183.

[doi:10.1007/s0035700500129](https://doi.org/10.1007/s0035700500129)

Szekely, G. J. and Rizzo, M. L. (2004) Testing for Equal Distributions in High Dimension, *InterStat*, November (5).

Szekely, G. J. (2000) Technical Report 03-05: \mathcal{E} -statistics: Energy of Statistical Samples, Department of Mathematics and Statistics, Bowling Green State University.

See Also

[edist](#) [ksample.e](#) [eqdist](#) [etest](#) [hclust](#)

Examples

```
## Not run:
library(cluster)
data(animals)
plot(energy.hclust(dist(animals)))

data(USArrests)
ecl <- energy.hclust(dist(USArrests))
print(ecl)
plot(ecl)
cutree(ecl, k=3)
cutree(ecl, h=150)

## compare performance of e-clustering, Ward's method, group average method
## when sampled populations have equal means: n=200, d=5, two groups
z <- rbind(matrix(rnorm(1000), nrow=200), matrix(rnorm(1000, 0, 5), nrow=200))
g <- c(rep(1, 200), rep(2, 200))
d <- dist(z)
e <- energy.hclust(d)
a <- hclust(d, method="average")
w <- hclust(d^2, method="ward.D2")
list("E" = table(cutree(e, k=2) == g), "Ward" = table(cutree(w, k=2) == g),
     "Avg" = table(cutree(a, k=2) == g))

## End(Not run)
```

 eqdist.etest

Multisample E-statistic (Energy) Test of Equal Distributions

Description

Performs the nonparametric multisample E-statistic (energy) test for equality of multivariate distributions.

Usage

```
eqdist.etest(x, sizes, distance = FALSE,
             method=c("original", "discoB", "discoF"), R)
eqdist.e(x, sizes, distance = FALSE,
         method=c("original", "discoB", "discoF"))
ksample.e(x, sizes, distance = FALSE,
          method=c("original", "discoB", "discoF"), ix = 1:sum(sizes))
```

Arguments

x	data matrix of pooled sample
sizes	vector of sample sizes
distance	logical: if TRUE, first argument is a distance matrix
method	use original (default) or distance components (discoB, discoF)
R	number of bootstrap replicates
ix	a permutation of the row indices of x

Details

The k-sample multivariate \mathcal{E} -test of equal distributions is performed. The statistic is computed from the original pooled samples, stacked in matrix x where each row is a multivariate observation, or the corresponding distance matrix. The first $\text{sizes}[1]$ rows of x are the first sample, the next $\text{sizes}[2]$ rows of x are the second sample, etc.

The test is implemented by nonparametric bootstrap, an approximate permutation test with R replicates.

The function `eqdist.e` returns the test statistic only; it simply passes the arguments through to `eqdist.etest` with $R = 0$.

The k-sample multivariate \mathcal{E} -statistic for testing equal distributions is returned. The statistic is computed from the original pooled samples, stacked in matrix x where each row is a multivariate observation, or from the distance matrix x of the original data. The first $\text{sizes}[1]$ rows of x are the first sample, the next $\text{sizes}[2]$ rows of x are the second sample, etc.

The two-sample \mathcal{E} -statistic proposed by Szekely and Rizzo (2004) is the e-distance $e(S_i, S_j)$, defined for two samples S_i, S_j of size n_i, n_j by

$$e(S_i, S_j) = \frac{n_i n_j}{n_i + n_j} [2M_{ij} - M_{ii} - M_{jj}],$$

where

$$M_{ij} = \frac{1}{n_i n_j} \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} \|X_{ip} - X_{jq}\|,$$

$\|\cdot\|$ denotes Euclidean norm, and X_{ip} denotes the p -th observation in the i -th sample.

The original (default method) k -sample \mathcal{E} -statistic is defined by summing the pairwise e-distances over all $k(k-1)/2$ pairs of samples:

$$\mathcal{E} = \sum_{1 \leq i < j \leq k} e(S_i, S_j).$$

Large values of \mathcal{E} are significant.

The discoB method computes the between-sample disco statistic. For a one-way analysis, it is related to the original statistic as follows. In the above equation, the weights $\frac{n_i n_j}{n_i + n_j}$ are replaced with

$$\frac{n_i + n_j}{2N} \frac{n_i n_j}{n_i + n_j} = \frac{n_i n_j}{2N}$$

where N is the total number of observations: $N = n_1 + \dots + n_k$.

The discoF method is based on the disco F ratio, while the discoB method is based on the between sample component.

Also see `disco` and `disco.between` functions.

Value

A list with class `htest` containing

<code>method</code>	description of test
<code>statistic</code>	observed value of the test statistic
<code>p.value</code>	approximate p-value of the test
<code>data.name</code>	description of data

`eqdist.e` returns test statistic only.

Note

The pairwise e-distances between samples can be conveniently computed by the `edist` function, which returns a `dist` object.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G. J. and Rizzo, M. L. (2004) Testing for Equal Distributions in High Dimension, *InterStat*, November (5).

M. L. Rizzo and G. J. Szekely (2010). DISCO Analysis: A Nonparametric Extension of Analysis of Variance, *Annals of Applied Statistics*, Vol. 4, No. 2, 1034-1055.
doi:10.1214/09AOAS245

Szekely, G. J. (2000) Technical Report 03-05: \mathcal{E} -statistics: Energy of Statistical Samples, Department of Mathematics and Statistics, Bowling Green State University.

See Also

[ksample.e](#), [edist](#), [disco](#), [disco.between](#), [energy.hclust](#).

Examples

```
data(iris)

## test if the 3 varieties of iris data (d=4) have equal distributions
eqdist.etest(iris[,1:4], c(50,50,50), R = 199)

## example that uses method="disco"
x <- matrix(rnorm(100), nrow=20)
y <- matrix(rnorm(100), nrow=20)
X <- rbind(x, y)
d <- dist(X)

# should match edist default statistic
set.seed(1234)
eqdist.etest(d, sizes=c(20, 20), distance=TRUE, R = 199)

# comparison with edist
edist(d, sizes=c(20, 10), distance=TRUE)

# for comparison
g <- as.factor(rep(1:2, c(20, 20)))
set.seed(1234)
disco(d, factors=g, distance=TRUE, R=199)

# should match statistic in edist method="discoB", above
set.seed(1234)
disco.between(d, factors=g, distance=TRUE, R=199)
```

Description

Pre-computed eigenvalues corresponding to the asymptotic sampling distribution of the energy test statistic for univariate normality, under the null hypothesis. Four Cases are computed:

1. Simple hypothesis, known parameters.
2. Estimated mean, known variance.
3. Known mean, estimated variance.
4. Composite hypothesis, estimated parameters.

Case 4 eigenvalues are used in the test function `normal.test` when `method=="limit"`.

Usage

```
data(EVnormal)
```

Format

Numeric matrix with 125 rows and 5 columns; column 1 is the index, and columns 2-5 are the eigenvalues of Cases 1-4.

Source

Computed

References

Szekely, G. J. and Rizzo, M. L. (2005) A New Test for Multivariate Normality, *Journal of Multivariate Analysis*, 93/1, 58-80, doi:[10.1016/j.jmva.2003.12.002](https://doi.org/10.1016/j.jmva.2003.12.002).

indep.test

Energy-tests of Independence

Description

Computes a multivariate nonparametric test of independence. The default method implements the distance covariance test [dcov.test](#).

Usage

```
indep.test(x, y, method = c("dcov", "mvI"), index = 1, R)
```

Arguments

x	matrix: first sample, observations in rows
y	matrix: second sample, observations in rows
method	a character string giving the name of the test
index	exponent on Euclidean distances
R	number of replicates

Details

`indep.test` with the default `method = "dcov"` computes the distance covariance test of independence. `index` is an exponent on the Euclidean distances. Valid choices for `index` are in $(0,2]$, with default value 1 (Euclidean distance). The arguments are passed to the `dcov.test` function. See the help topic [dcov.test](#) for the description and documentation and also see the references below.

`indep.test` with `method = "mvI"` computes the coefficient \mathcal{I}_n and performs a nonparametric \mathcal{E} -test of independence. The arguments are passed to `mvI.test`. The `index` argument is ignored (`index = 1` is applied). See the help topic [mvI.test](#) and also see the reference (2006) below for details.

The test decision is obtained via bootstrap, with `R` replicates. The sample sizes (number of rows) of the two samples must agree, and samples must not contain missing values.

These energy tests of independence are based on related theoretical results, but different test statistics. The `dcov` method is faster than `mvI` method by approximately a factor of $O(n)$.

Value

`indep.test` returns a list with class `htest` containing

<code>method</code>	description of test
<code>statistic</code>	observed value of the test statistic $n\mathcal{V}_n^2$ or $n\mathcal{I}_n^2$
<code>estimate</code>	\mathcal{V}_n or \mathcal{I}_n
<code>estimates</code>	a vector [<code>dCov(x,y)</code> , <code>dCor(x,y)</code> , <code>dVar(x)</code> , <code>dVar(y)</code>] (method <code>dcov</code>)
<code>replicates</code>	replicates of the test statistic
<code>p.value</code>	approximate p-value of the test
<code>data.name</code>	description of data

Note

As of energy-1.1-0, `indep.etest` is deprecated and replaced by `indep.test`, which has methods for two different energy tests of independence. `indep.test` applies the distance covariance test (see `dcov.test`) by default (`method = "dcov"`). The original `indep.etest` applied the independence coefficient \mathcal{I}_n , which is now obtained by `method = "mvI"`.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G.J. and Rizzo, M.L. (2009), Brownian Distance Covariance, *Annals of Applied Statistics*, Vol. 3 No. 4, pp. 1236-1265. (Also see discussion and rejoinder.)

[doi:10.1214/09AOAS312](https://doi.org/10.1214/09AOAS312)

Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.

[doi:10.1214/009053607000000505](https://doi.org/10.1214/009053607000000505)

Bakirov, N.K., Rizzo, M.L., and Szekely, G.J. (2006), A Multivariate Nonparametric Test of Independence, *Journal of Multivariate Analysis* 93/1, 58-80,

[doi:10.1016/j.jmva.2005.10.005](https://doi.org/10.1016/j.jmva.2005.10.005)

See Also

[dcov.test](#) [mvl.test](#) [dcov](#) [mvl](#)

Examples

```
## independent multivariate data
x <- matrix(rnorm(60), nrow=20, ncol=3)
y <- matrix(rnorm(40), nrow=20, ncol=2)
indep.test(x, y, method = "dcov", R = 99)
indep.test(x, y, method = "mvl", R = 99)

## dependent multivariate data
if (require(MASS)) {
  Sigma <- matrix(c(1, .1, 0, 0, 1, 0, 0, .1, 1), 3, 3)
  x <- mvrnorm(30, c(0, 0, 0), diag(3))
  y <- mvrnorm(30, c(0, 0, 0), Sigma) * x
  indep.test(x, y, R = 99) #dcov method
  indep.test(x, y, method = "mvl", R = 99)
}
```

kgroups

K-Groups Clustering

Description

Perform k-groups clustering by energy distance.

Usage

```
kgroups(x, k, iter.max = 10, nstart = 1, cluster = NULL)
```

Arguments

x	Data frame or data matrix or distance object
k	number of clusters
iter.max	maximum number of iterations
nstart	number of restarts
cluster	initial clustering vector

Details

K-groups is based on the multisample energy distance for comparing distributions. Based on the disco decomposition of total dispersion (a Gini type mean distance) the objective function should either maximize the total between cluster energy distance, or equivalently, minimize the total within cluster energy distance. It is more computationally efficient to minimize within distances, and that

makes it possible to use a modified version of the Hartigan-Wong algorithm (1979) to implement K-groups clustering.

The within cluster Gini mean distance is

$$G(C_j) = \frac{1}{n_j^2} \sum_{i,m=1}^{n_j} |x_{i,j} - x_{m,j}|$$

and the K-groups within cluster distance is

$$W_j = \frac{n_j}{2} G(C_j) = \frac{1}{2n_j} \sum_{i,m=1}^{n_j} |x_{i,j} - x_{m,j}|.$$

If z is the data matrix for cluster C_j , then W_j could be computed as `sum(dist(z)) / nrow(z)`.

If cluster is not NULL, the clusters are initialized by this vector (can be a factor or integer vector). Otherwise clusters are initialized with random labels in k approximately equal size clusters.

If x is not a distance object (`class(x) == "dist"`) then x is converted to a data matrix for analysis.

Run up to `iter.max` complete passes through the data set until a local min is reached. If `nstart > 1`, on second and later starts, clusters are initialized at random, and the best result is returned.

Value

An object of class `kgroups` containing the components

<code>call</code>	the function call
<code>cluster</code>	vector of cluster indices
<code>sizes</code>	cluster sizes
<code>within</code>	vector of Gini within cluster distances
<code>W</code>	sum of within cluster distances
<code>count</code>	number of moves
<code>iterations</code>	number of iterations
<code>k</code>	number of clusters

`cluster` is a vector containing the group labels, 1 to k . `print.kgroups` prints some of the components of the `kgroups` object.

Expect that `count` is 0 if the algorithm converged to a local min (that is, 0 moves happened on the last iteration). If `iterations` equals `iter.max` and `count` is positive, then the algorithm did not converge to a local min.

Author(s)

Maria Rizzo and Songzi Li

References

- Li, Songzi (2015). "K-groups: A Generalization of K-means by Energy Distance." Ph.D. thesis, Bowling Green State University.
- Li, S. and Rizzo, M. L. (2017). "K-groups: A Generalization of K-means Clustering". ArXiv e-print 1711.04359. <https://arxiv.org/abs/1711.04359>
- Szekely, G. J., and M. L. Rizzo. "Testing for equal distributions in high dimension." *InterStat* 5, no. 16.10 (2004).
- Rizzo, M. L., and G. J. Szekely. "Disco analysis: A nonparametric extension of analysis of variance." *The Annals of Applied Statistics* (2010): 1034-1055.
- Hartigan, J. A. and Wong, M. A. (1979). "Algorithm AS 136: A K-means clustering algorithm." *Applied Statistics*, 28, 100-108. doi: 10.2307/2346830.

Examples

```
x <- as.matrix(iris[,1:4])
set.seed(123)
kg <- kgroups(x, k = 3, iter.max = 5, nstart = 2)
kg
fitted(kg)

d <- dist(x)
set.seed(123)
kg <- kgroups(d, k = 3, iter.max = 5, nstart = 2)
kg

kg$cluster

fitted(kg)
fitted(kg, method = "groups")
```

mutual independence *Energy Test of Mutual Independence*

Description

The test statistic is the sum of d-1 bias-corrected squared dcor statistics where the number of variables is d. Implementation is by permutation test.

Usage

```
mutualIndep.test(x, R)
```

Arguments

x	data matrix or data frame
R	number of permutation replicates

Details

A population coefficient for mutual independence of d random variables, $d \geq 2$, is

$$\sum_{k=1}^{d-1} \mathcal{R}^2(X_k, [X_{k+1}, \dots, X_d]).$$

which is non-negative and equals zero iff mutual independence holds. For example, if $d=4$ the population coefficient is

$$\mathcal{R}^2(X_1, [X_2, X_3, X_4]) + \mathcal{R}^2(X_2, [X_3, X_4]) + \mathcal{R}^2(X_3, X_4),$$

A permutation test is implemented based on the corresponding sample coefficient. To test mutual independence of

$$X_1, \dots, X_d$$

the test statistic is the sum of the $d-1$ statistics (bias-corrected $dcor^2$ statistics):

$$\sum_{k=1}^{d-1} \mathcal{R}_n^*(X_k, [X_{k+1}, \dots, X_d])$$

Value

`mutualIndep.test` returns an object of class `power.htest`.

Note

See Szekely and Rizzo (2014) for details on unbiased $dCov^2$ and bias-corrected $dCor^2$ (`bcdcor`) statistics.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.

[doi:10.1214/009053607000000505](https://doi.org/10.1214/009053607000000505)

Szekely, G.J. and Rizzo, M.L. (2014), Partial Distance Correlation with Methods for Dissimilarities. *Annals of Statistics*, Vol. 42 No. 6, 2382-2412.

See Also

[bcdcor](#), [dcovU_stats](#)

Examples

```
x <- matrix(rnorm(100), nrow=20, ncol=5)
mutualIndep.test(x, 199)
```

mvI.test

*Independence Coefficient and Test***Description**

Computes a type of multivariate nonparametric E-statistic and test of independence based on independence coefficient \mathcal{I}_n . This coefficient pre-dates and is different from distance covariance or distance correlation.

Usage

```
mvI.test(x, y, R)
mvI(x, y)
```

Arguments

x	matrix: first sample, observations in rows
y	matrix: second sample, observations in rows
R	number of replicates

Details

mvI computes the coefficient \mathcal{I}_n and mvI.test performs a nonparametric test of independence. The test decision is obtained via permutation bootstrap, with R replicates. The sample sizes (number of rows) of the two samples must agree, and samples must not contain missing values.

Historically this is the first energy test of independence. The distance covariance test [dcov.test](#), distance correlation [dcor](#), and related methods are more recent (2007, 2009).

The distance covariance test [dcov.test](#) and distance correlation test [dcor.test](#) are much faster and have different properties than mvI.test. All are based on a population independence coefficient that characterizes independence and of these tests are statistically consistent. However, dCor is scale invariant while \mathcal{I}_n is not. In applications [dcor.test](#) or [dcov.test](#) are the recommended tests.

Computing formula from Bakirov, Rizzo, and Szekely (2006), equation (2):

Suppose the two samples are $X_1, \dots, X_n \in R^p$ and $Y_1, \dots, Y_n \in R^q$. Define $Z_{kl} = (X_k, Y_l) \in R^{p+q}$.

The independence coefficient \mathcal{I}_n is defined

$$\mathcal{I}_n = \sqrt{\frac{2\bar{z} - z_d - z}{x + y - z}},$$

where

$$z_d = \frac{1}{n^2} \sum_{k,l=1}^n |Z_{kk} - Z_{ll}|_{p+q},$$

$$z = \frac{1}{n^4} \sum_{k,l=1}^n \sum_{i,j=1}^n |Z_{kl} - Z_{ij}|_{p+q},$$

$$\bar{z} = \frac{1}{n^3} \sum_{k=1}^n \sum_{i,j=1}^n |Z_{kk} - Z_{ij}|_{p+q},$$

$$x = \frac{1}{n^2} \sum_{k,l=1}^n |X_k - X_l|_p,$$

$$y = \frac{1}{n^2} \sum_{k,l=1}^n |Y_k - Y_l|_q.$$

Some properties:

- $0 \leq \mathcal{I}_n \leq 1$ (Theorem 1).
- Large values of $n\mathcal{I}_n^2$ (or \mathcal{I}_n) support the alternative hypothesis that the sampled random variables are dependent.
- \mathcal{I}_n is invariant to shifts and orthogonal transformations of X and Y.
- $\sqrt{n}\mathcal{I}_n$ determines a statistically consistent test of independence against all fixed dependent alternatives (Corollary 1).
- The population independence coefficient \mathcal{I} is a normalized distance between the joint characteristic function and the product of the marginal characteristic functions. \mathcal{I}_n converges almost surely to \mathcal{I} as $n \rightarrow \infty$. X and Y are independent if and only if $\mathcal{I}(X, Y) = 0$. See the reference below for more details.

Value

mvI returns the statistic. mvI.test returns a list with class `htest` containing

method	description of test
statistic	observed value of the test statistic $n\mathcal{I}_n^2$
estimate	\mathcal{I}_n
replicates	permutation replicates
p.value	p-value of the test
data.name	description of data

Note

On scale invariance: Distance correlation ([dcor](#)) has the property that if we change the scale of X from e.g., meters to kilometers, and the scale of Y from e.g. grams to ounces, the statistic and the test are not changed. \mathcal{I}_n does not have this property; it is invariant only under a common rescaling of X and Y by the same constant. Thus, if the units of measurement change for either or both variables, `dCor` is invariant, but \mathcal{I}_n and possibly the `mvI.test` decision changes.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Bakirov, N.K., Rizzo, M.L., and Szekely, G.J. (2006), A Multivariate Nonparametric Test of Independence, *Journal of Multivariate Analysis* 93/1, 58-80.

Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.

Szekely, G.J. and Rizzo, M.L. (2009), Brownian Distance Covariance, *Annals of Applied Statistics*, Vol. 3, No. 4, 1236-1265.

See Also

[dcov.test](#) [dcov](#) [dcor.test](#) [dcor](#) [dcov2d](#) [dcor2d](#) [indep.test](#)

Examples

```
mvI(iris[1:25, 1], iris[1:25, 2])
```

```
mvI.test(iris[1:25, 1], iris[1:25, 2], R=99)
```

mvnorm.test

E-statistic (Energy) Test of Multivariate Normality

Description

Performs the E-statistic (energy) test of multivariate or univariate normality.

Usage

```
mvnorm.test(x, R)
mvnorm.etest(x, R)
mvnorm.e(x)
```

Arguments

x data matrix of multivariate sample, or univariate data vector
R number of bootstrap replicates

Details

If x is a matrix, each row is a multivariate observation. The data will be standardized to zero mean and identity covariance matrix using the sample mean vector and sample covariance matrix. If x is a vector, mvnorm.e returns the univariate statistic $\text{normal.e}(x)$. If the data contains missing values or the sample covariance matrix is singular, mvnorm.e returns NA.

The \mathcal{E} -test of multivariate normality was proposed and implemented by Szekely and Rizzo (2005). The test statistic for d-variate normality is given by

$$\mathcal{E} = n \left(\frac{2}{n} \sum_{i=1}^n E \|y_i - Z\| - E \|Z - Z'\| - \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n \|y_i - y_j\| \right),$$

where y_1, \dots, y_n is the standardized sample, Z, Z' are iid standard d-variate normal, and $\|\cdot\|$ denotes Euclidean norm.

The \mathcal{E} -test of multivariate (univariate) normality is implemented by parametric bootstrap with R replicates.

Value

The value of the \mathcal{E} -statistic for multivariate normality is returned by `mvnorm.e`.

`mvnorm.test` returns a list with class `htest` containing

<code>method</code>	description of test
<code>statistic</code>	observed value of the test statistic
<code>p.value</code>	approximate p-value of the test
<code>data.name</code>	description of data

`mvnorm.etest` is replaced by `mvnorm.test`.

Note

If the data is univariate, the test statistic is formally the same as the multivariate case, but a more efficient computational formula is applied in [normal.e](#).

[normal.test](#) also provides an optional method for the test based on the asymptotic sampling distribution of the test statistic.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

- Szekely, G. J. and Rizzo, M. L. (2005) A New Test for Multivariate Normality, *Journal of Multivariate Analysis*, 93/1, 58-80, doi:[10.1016/j.jmva.2003.12.002](https://doi.org/10.1016/j.jmva.2003.12.002).
- Mori, T. F., Szekely, G. J. and Rizzo, M. L. "On energy tests of normality." *Journal of Statistical Planning and Inference* 213 (2021): 1-15.
- Rizzo, M. L. (2002). A New Rotation Invariant Goodness-of-Fit Test, Ph.D. dissertation, Bowling Green State University.
- Szekely, G. J. (1989) Potential and Kinetic Energy in Statistics, Lecture Notes, Budapest Institute of Technology (Technical University).

See Also

[normal.test](#) for the energy test of univariate normality and [normal.e](#) for the statistic.

Examples

```
## compute normality test statistic for iris Setosa data
data(iris)
mvnorm.e(iris[1:50, 1:4])

## test if the iris Setosa data has multivariate normal distribution
mvnorm.test(iris[1:50,1:4], R = 199)
```

normal.test

Energy Test of Univariate Normality

Description

Performs the energy test of univariate normality for the composite hypothesis Case 4, estimated parameters.

Usage

```
normal.test(x, method=c("mc", "limit"), R)
normal.e(x)
```

Arguments

x	univariate data vector
method	method for p-value
R	number of replications if Monte Carlo method

Details

If method="mc" this test function applies the parametric bootstrap method implemented in [mvnorm.test](#).

If method="limit", the p-value of the test is computed from the asymptotic distribution of the test statistic under the null hypothesis. The asymptotic distribution is a quadratic form of centered Gaussian random variables, which has the form

$$\sum_{k=1}^{\infty} \lambda_k Z_k^2,$$

where λ_k are positive constants (eigenvalues) and Z_k are iid standard normal variables. Eigenvalues are pre-computed and stored internally. A p-value is computed using Imhof's method as implemented in the **CompQuadForm** package.

Note that the "limit" method is intended for moderately large samples because it applies the asymptotic distribution.

The energy test of normality was proposed and implemented by Szekely and Rizzo (2005). See [mvnorm.test](#) for more details.

Value

`normal.e` returns the energy goodness-of-fit statistic for a univariate sample.

`normal.test` returns a list with class `htest` containing

`statistic` observed value of the test statistic

`p.value` p-value of the test

`estimate` sample estimates: mean, sd

`data.name` description of data

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G. J. and Rizzo, M. L. (2005) A New Test for Multivariate Normality, *Journal of Multivariate Analysis*, 93/1, 58-80, doi:[10.1016/j.jmva.2003.12.002](https://doi.org/10.1016/j.jmva.2003.12.002).

Mori, T. F., Szekely, G. J. and Rizzo, M. L. "On energy tests of normality." *Journal of Statistical Planning and Inference* 213 (2021): 1-15.

Rizzo, M. L. (2002). A New Rotation Invariant Goodness-of-Fit Test, Ph.D. dissertation, Bowling Green State University.

J. P. Imhof (1961). Computing the Distribution of Quadratic Forms in Normal Variables, *Biometrika*, Volume 48, Issue 3/4, 419-426.

See Also

[mvnorm.test](#) and [mvnorm.e](#) for the energy test of multivariate normality and the test statistic for multivariate samples.

Examples

```
x <- iris[1:50, 1]
normal.e(x)
normal.test(x, R=199)
normal.test(x, method="limit")
```

pdcor

Partial distance correlation and covariance

Description

Partial distance correlation `pdcor`, `pdcov`, and tests.

Usage

```
pdcov.test(x, y, z, R)
pdcor.test(x, y, z, R)
pdcor(x, y, z)
pdcov(x, y, z)
```

Arguments

x	data or dist object of first sample
y	data or dist object of second sample
z	data or dist object of third sample
R	replicates for permutation test

Details

pdcor(x, y, z) and pdcov(x, y, z) compute the partial distance correlation and partial distance covariance, respectively, of x and y removing z.

A test for zero partial distance correlation (or zero partial distance covariance) is implemented in pdcor.test, and pdcov.test.

Argument types supported are numeric data matrix, data.frame, tibble, numeric vector, class "dist" object, or factor. For unordered factors a 0-1 distance matrix is computed.

Value

Each test returns an object of class htest.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G.J. and Rizzo, M.L. (2014), Partial Distance Correlation with Methods for Dissimilarities. *Annals of Statistics*, Vol. 42 No. 6, 2382-2412.

Examples

```
n = 30
R <- 199

## mutually independent standard normal vectors
x <- rnorm(n)
y <- rnorm(n)
z <- rnorm(n)

pdcor(x, y, z)
pdcov(x, y, z)
set.seed(1)
pdcov.test(x, y, z, R=R)
```

```
set.seed(1)
pdcov.test(x, y, z, R=R)

if (require(MASS)) {
  p = 4
  mu <- rep(0, p)
  Sigma <- diag(p)

  ## linear dependence
  y <- mvrnorm(n, mu, Sigma) + x
  print(pdcov.test(x, y, z, R=R))

  ## non-linear dependence
  y <- mvrnorm(n, mu, Sigma) * x
  print(pdcov.test(x, y, z, R=R))
}
```

Description

Performs the mean distance goodness-of-fit test and the energy goodness-of-fit test of Poisson distribution with unknown parameter.

Usage

```
poisson.e(x)
poisson.m(x)
poisson.etest(x, R)
poisson.mtest(x, R)
poisson.tests(x, R, test="all")
```

Arguments

x	vector of nonnegative integers, the sample data
R	number of bootstrap replicates
test	name of test(s)

Details

Two distance-based tests of Poissonity are applied in `poisson.tests`, "M" and "E". The default is to do all tests and return results in a data frame. Valid choices for `test` are "M", "E", or "all" with default "all".

If "all" tests, all tests are performed by a single parametric bootstrap computing all test statistics on each sample.

The "M" choice is two tests, one based on a Cramer-von Mises distance and the other an Anderson-Darling distance. The "E" choice is the energy goodness-of-fit test.

R must be a positive integer for a test. If R is missing or 0, a warning is printed but test statistics are computed (without testing).

The mean distance test of Poissonity (M-test) is based on the result that the sequence of expected values $E|X-j|$, $j=0,1,2,\dots$ characterizes the distribution of the random variable X. As an application of this characterization one can get an estimator $\hat{F}(j)$ of the CDF. The test statistic (see [poisson.m](#)) is a Cramer-von Mises type of distance, with M-estimates replacing the usual EDF estimates of the CDF:

$$M_n = n \sum_{j=0}^{\infty} (\hat{F}(j) - F(j; \hat{\lambda}))^2 f(j; \hat{\lambda}).$$

In `poisson.tests`, an Anderson-Darling type of weight is also applied when `test="M"` or `test="all"`.

The tests are implemented by parametric bootstrap with R replicates.

An energy goodness-of-fit test (E) is based on the test statistic

$$Q_n = n \left(\frac{2}{n} \sum_{i=1}^n E|x_i - X| - E|X - X'| - \frac{1}{n^2} \sum_{i,j=1}^n |x_i - x_j| \right),$$

where X and X' are iid with the hypothesized null distribution. For a test of $H: X \sim \text{Poisson}(\lambda)$, we can express $E|X-X'|$ in terms of Bessel functions, and $E|x_i - X|$ in terms of the CDF of $\text{Poisson}(\lambda)$.

If `test="all"` or not specified, all tests are run with a single parametric bootstrap. `poisson.mtest` implements only the Poisson M-test with Cramer-von Mises type distance. `poisson.etest` implements only the Poisson energy test.

Value

The functions `poisson.m` and `poisson.e` return the test statistics. The function `poisson.mtest` or `poisson.etest` return an `htest` object containing

<code>method</code>	Description of test
<code>statistic</code>	observed value of the test statistic
<code>p.value</code>	approximate p-value of the test
<code>data.name</code>	replicates R
<code>estimate</code>	sample mean

`poisson.tests` returns "M-CvM test", "M-AD test" and "Energy test" results in a data frame with columns

<code>estimate</code>	sample mean
<code>statistic</code>	observed value of the test statistic
<code>p.value</code>	approximate p-value of the test
<code>method</code>	Description of test

which can be coerced to a tibble.

Note

The running time of the M test is much faster than the E-test.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G. J. and Rizzo, M. L. (2004) Mean Distance Test of Poisson Distribution, *Statistics and Probability Letters*, 67/3, 241-247. doi:10.1016/j.spl.2004.01.005.

Szekely, G. J. and Rizzo, M. L. (2005) A New Test for Multivariate Normality, *Journal of Multivariate Analysis*, 93/1, 58-80, doi:10.1016/j.jmva.2003.12.002.

Examples

```
x <- rpois(50, 2)
poisson.m(x)
poisson.e(x)

poisson.etest(x, R=199)
poisson.mtest(x, R=199)
poisson.tests(x, R=199)
```

sortrank

Sort, order and rank a vector

Description

A utility that returns a list with the components equivalent to `sort(x)`, `order(x)`, `rank(x, ties.method = "first")`.

Usage

```
sortrank(x)
```

Arguments

x vector compatible with `sort(x)`

Details

This utility exists to save a little time on large vectors when two or all three of the `sort()`, `order()`, `rank()` results are required. In case of ties, the ranks component matches `rank(x, ties.method = "first")`.

Value

A list with components

x	the sorted input vector x
ix	the permutation = order(x) which rearranges x into ascending order
r	the ranks of x

Note

This function was benchmarked faster than the combined calls to sort and rank.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu>

References

See [sort](#).

Examples

```
sortrank(rnorm(5))
```

Unbiased distance covariance

Unbiased dcov and bias-corrected dcor statistics

Description

These functions compute unbiased estimators of squared distance covariance and a bias-corrected estimator of (squared) distance correlation.

Usage

```
bcdcor(x, y)  
dcovU(x, y)
```

Arguments

x	data or dist object of first sample
y	data or dist object of second sample

Details

The unbiased (squared) dcov is inner product definition of dCov, in the Hilbert space of U-centered distance matrices.

The sample sizes (number of rows) of the two samples must agree, and samples must not contain missing values.

Argument types supported are numeric data matrix, data.frame, or tibble, with observations in rows; numeric vector; ordered or unordered factors. In case of unordered factors a 0-1 distance matrix is computed.

Value

dcovU returns the unbiased estimator of squared dcov. bcdcor returns a bias-corrected estimator of squared dcor.

Note

Unbiased distance covariance (SR2014) corresponds to the biased (original) dCov². Since dcovU is an unbiased statistic, it is signed and we do not take the square root. For the original distance covariance test of independence (SRB2007, SR2009), the distance covariance test statistic is the V-statistic $n \text{dCov}^2 = n \mathcal{V}_n^2$ (not dCov). Similarly, bcdcor is bias-corrected, so we do not take the square root as with dCor.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G.J. and Rizzo, M.L. (2014), Partial Distance Correlation with Methods for Dissimilarities. *Annals of Statistics*, Vol. 42 No. 6, 2382-2412.

Szekely, G.J., Rizzo, M.L., and Bakirov, N.K. (2007), Measuring and Testing Dependence by Correlation of Distances, *Annals of Statistics*, Vol. 35 No. 6, pp. 2769-2794.

[doi:10.1214/009053607000000505](https://doi.org/10.1214/009053607000000505)

Szekely, G.J. and Rizzo, M.L. (2009), Brownian Distance Covariance, *Annals of Applied Statistics*, Vol. 3, No. 4, 1236-1265.

[doi:10.1214/09AOAS312](https://doi.org/10.1214/09AOAS312)

Examples

```
x <- iris[1:50, 1:4]
y <- iris[51:100, 1:4]
dcovU(x, y)
bcdcor(x, y)
```

U_product

Inner product in the Hilbert space of U-centered distance matrices

Description

Stand-alone function to compute the inner product in the Hilbert space of U-centered distance matrices, as in the definition of partial distance covariance.

Usage

```
U_product(U, V)
```

Arguments

U	U-centered distance matrix
V	U-centered distance matrix

Details

Note that `pdcor`, etc. functions include the centering and projection operations, so that these stand alone versions are not needed except in case one wants to check the internal computations.

Exported from `U_product.cpp`.

Value

`U_product` returns the inner product, a scalar.

Author(s)

Maria L. Rizzo <mrizzo@bgsu.edu> and Gabor J. Szekely

References

Szekely, G.J. and Rizzo, M.L. (2014), Partial Distance Correlation with Methods for Dissimilarities, *Annals of Statistics*, Vol. 42, No. 6, pp. 2382-2412.

Examples

```
x <- iris[1:10, 1:4]
y <- iris[11:20, 1:4]
M1 <- as.matrix(dist(x))
M2 <- as.matrix(dist(y))
U <- U_center(M1)
V <- U_center(M2)

U_product(U, V)
dcovU_stats(M1, M2)
```

Index

- * **cluster**
 - edist, 18
 - energy.hclust, 20
 - kgroups, 28
- * **distance correlation**
 - centering distance matrices, 3
 - dcorT, 5
 - dcov.test, 7
 - dcov2d, 9
 - dcovU_stats, 11
 - distance correlation, 14
 - energy-deprecated, 20
 - mutual independence, 30
 - pdcor, 37
 - U_product, 44
 - Unbiased distance covariance, 42
- * **distance covariance**
 - centering distance matrices, 3
 - dcorT, 5
 - dcov.test, 7
 - dcov2d, 9
 - dcovU_stats, 11
 - distance correlation, 14
 - energy-deprecated, 20
 - mutual independence, 30
 - pdcor, 37
 - U_product, 44
 - Unbiased distance covariance, 42
- * **energy statistics**
 - centering distance matrices, 3
 - dcor.ttest, 4
 - dcorT, 5
 - dcov.test, 7
 - dcov2d, 9
 - dcovU_stats, 11
 - distance correlation, 14
 - edist, 18
 - energy-deprecated, 20
 - energy.hclust, 20
 - eqdist.etest, 23
 - indep.test, 26
 - mutual independence, 30
 - mvI.test, 32
 - mvnorm.test, 34
 - normal.test, 36
 - pdcor, 37
 - U_product, 44
 - Unbiased distance covariance, 42
- * **energy**
 - Poisson Tests, 39
- * **goodness-of-fit**
 - mvnorm.test, 34
 - normal.test, 36
- * **htest**
 - dcor.ttest, 4
 - dcorT, 5
 - dcov.test, 7
 - dcov2d, 9
 - disco, 12
 - eqdist.etest, 23
 - indep.test, 26
 - mvI.test, 32
 - mvnorm.test, 34
 - normal.test, 36
 - pdcor, 37
 - Poisson Tests, 39
- * **independence**
 - dcorT, 5
 - dcov.test, 7
 - dcov2d, 9
 - dcovU_stats, 11
 - distance correlation, 14
 - energy-deprecated, 20
 - indep.test, 26
 - mutual independence, 30
 - mvI.test, 32
 - pdcor, 37
 - Unbiased distance covariance, 42

- * **multivariate**
 - centering distance matrices, 3
 - dcor.ttest, 4
 - dcorT, 5
 - dcov.test, 7
 - dcovU_stats, 11
 - disco, 12
 - distance correlation, 14
 - edist, 18
 - energy-deprecated, 20
 - energy-package, 2
 - energy.hclust, 20
 - eqdist.etest, 23
 - indep.test, 26
 - kgroups, 28
 - mutual independence, 30
 - mvI.test, 32
 - mvnorm.test, 34
 - pdcor, 37
 - U_product, 44
 - Unbiased distance covariance, 42
- * **nonparametric**
 - dcorT, 5
 - dcov.test, 7
 - dcov2d, 9
 - dcovU_stats, 11
 - edist, 18
 - energy-deprecated, 20
 - eqdist.etest, 23
 - indep.test, 26
 - mvI.test, 32
 - pdcor, 37
 - Unbiased distance covariance, 42
- * **normal distribution**
 - mvnorm.test, 34
 - normal.test, 36
- * **package**
 - energy-package, 2
- bcdcor, 6, 16, 31
- bcdcor (Unbiased distance covariance), 42
- calc_dist (Distance Matrix), 17
- centering distance matrices, 3
- D_center (centering distance matrices), 3
- Dcenter (centering distance matrices), 3
- DCOR, 6
- DCOR (energy-deprecated), 20
- dcor, 6, 9, 10, 32–34
- dcor (distance correlation), 14
- dcor.t (dcor.ttest), 4
- dcor.test, 10, 16, 32, 34
- dcor.test (dcov.test), 7
- dcor.ttest, 4, 9
- dcor2d, 16, 34
- dcor2d (dcov2d), 9
- dcorT, 5, 5
- dcov, 8–10, 28, 34
- dcov (distance correlation), 14
- dcov.test, 6, 7, 10, 16, 26–28, 32, 34
- dcov2d, 9, 16, 34
- dcovU, 16
- dcovU (Unbiased distance covariance), 42
- dcovU_stats, 11, 31
- disco, 12, 19, 25
- disco.between, 25
- distance correlation, 14
- distance covariance (dcov.test), 7
- Distance Matrix, 17
- edist, 14, 18, 22, 25
- eigenvalues (EVnormal), 25
- energy (energy-package), 2
- energy-deprecated, 20
- energy-package, 2
- energy.hclust, 19, 20, 25
- eqdist.e, 14
- eqdist.e (eqdist.etest), 23
- eqdist.etest, 14, 19, 22, 23
- EVnormal, 25
- indep.test, 26, 34
- is.dmatrix (Distance Matrix), 17
- kgroups, 28
- ksample.e, 14, 19, 22, 25
- ksample.e (eqdist.etest), 23
- mutual independence, 30
- mutualIndep.test (mutual independence), 30
- mvI, 28
- mvI (mvI.test), 32
- mvI.test, 27, 28, 32
- mvnorm.e, 37

mvnorm.e (mvnorm.test), 34
mvnorm.etest (mvnorm.test), 34
mvnorm.test, 34, 36, 37

normal.e, 35
normal.e (normal.test), 36
normal.test, 35, 36

pdcor, 16, 37
pdcor.test, 9, 16
pdcov (pdcor), 37
pdcov.test, 9
Poisson Tests, 39
poisson.e (Poisson Tests), 39
poisson.etest (Poisson Tests), 39
poisson.m, 40
poisson.m (Poisson Tests), 39
poisson.mtest (Poisson Tests), 39
poisson.tests (Poisson Tests), 39
print.disco (disco), 12

sort, 42
sortrank, 41

U_center (centering distance matrices),
3
U_product, 44
Ucenter (centering distance matrices), 3
Unbiased distance covariance, 42