Package ‘Dogoftest’

October 12, 2026

Title Distributed Online Goodness-of-Fit Tests for Distributed Datasets

Version 0.4.2

Description Distributed Online Goodness-of-Fit Test can process the distributed datasets. The philosophy of the package is described in Guo G.(2024) <doi:10.1016/j.apm.2024.115709>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports stats

LazyData true

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Guangbao Guo [aut, cre] (ORCID: <https://orcid.org/0000-0002-4115-6218>), Di Chang [aut]

Maintainer Guangbao Guo <ggb11111111@163.com>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2026-10-12 11:50:02 UTC

Contents

AD2gof 2
ADgof 3
BSdata 4
CVM2gof 5
CVMgof 6
DAD 7
DAD2 8
DCVM 9

DCVM2 10
DKS 11
DKS2 12
DPECVM 13
DPECVM2 14
goats 15
KS2gof 16
KSgof 17
Kuiper2gof 18
PE2AD 19
PE2CVM 21
PE2KS 23
PECVM 25
PEKS 27
qCvMgof 29
run_custom_tests 29
run_FDP_sim 30
simpleCvMgof 31
Snow 32
strength 33
Wgof 34
WhiteWine 35

Index **36**

`AD2gof` *Two-Sample Anderson-Darling Permutation Test*

Description

Permutation two-sample Anderson-Darling test. AD statistic puts heavier weights on tail regions of distribution.

Usage

```
AD2gof(
  x,
  y,
  alternative = c("two.sided", "less", "greater"),
  nboots = 2000,
  keep.perms = FALSE
)
```

Arguments

`x`	Numeric vector, first sample.
`y`	Numeric vector, second sample.
`alternative`	Character string for alternative hypothesis: `"two.sided"`, `"less"`, `"greater"`. Default "two.sided".
`nboots`	Integer, number of permutation replicates, default 2000.
`keep.perms`	Logical; whether store permutation statistics, default FALSE.

Value

An `htest` list object:

statistic Observed Anderson-Darling statistic.

p.value Permutation p-value.

alternative Specified alternative.

method Test description.

data.name Sample names.

permutations Permutation statistics vector (if keep.perms=TRUE).

Examples

```
set.seed(123)
x <- rnorm(100)
y <- rnorm(100, mean=0.2)
out <- AD2gof(x, y, nboots = 200)
out$p.value
```

`ADgof` *Anderson-Darling Test for Normality*

Description

Performs the Anderson-Darling goodness-of-fit test for normality, where the mean and standard deviation of the normal distribution are estimated from the data. The p-value is computed using asymptotic approximation formulas adjusted for parameter estimation.

Usage

```
ADgof(x, eps = 1e-15)
```

Arguments

`x`	A numeric vector of sample observations.
`eps`	A small positive constant used to avoid numerical issues (e.g., `log(0)`) when evaluating tail probabilities (default: `1e-15`).

Details

This function tests the null hypothesis that the data come from a normal distribution, with unknown parameters replaced by their maximum likelihood estimates (sample mean and sample standard deviation).

The Anderson-Darling statistic is computed as:

$$A = -n - \frac{1}{n}\sum_{i=1}^{n}(2i-1)\left[\log F(x_{(i)}) + \log(1 - F(x_{(n+1-i)}))\right],$$

where F is the CDF of the fitted normal distribution. A small-sample adjustment $A^* = A \cdot (1 + 0.75/n + 2.25/n^2)$ is applied before computing the p-value, though the reported statistic is the unadjusted A.

The p-value is obtained via piecewise exponential-polynomial approximations derived for the case where parameters are estimated from the data (composite null hypothesis). These approximations are based on empirical studies (e.g., Stephens, 1986) and are valid for moderate to large sample sizes.

Note: Because the parameters are estimated from the data, this is not the standard Anderson-Darling test for a fully specified distribution. P-values are therefore approximate.

Value

An object of class `"htest"` with the following components:

statistic The Anderson-Darling test statistic (`AD`).

p.value The approximate p-value based on adjusted asymptotic formulas.

method A character string describing the test performed.

data.name The name of the input data vector.

parameter Named vector containing the estimated `mean` and `sd`.

Examples

```
set.seed(100)
x <- rnorm(60, mean = 0, sd = 5)
ADgof(x)
```

`BSdata` *Data set*

Description

`psi21k`, `psi26k`, and `psi31k` are from Birnbaum and Saunders (1969). The fatigue lifetimes of aluminum specimens exposed to a maximum stress of 21,000 psi, 26,000 psi, 31,000 psi, respectively.

`bearings` is from McCool (1974). The fatigue lifetimes (in hours) of ten bearings.

`fatigue` is from Brown and Miller (1978). The fatigue lifetimes of cylindrical specimens subjected to combined torsional and axial loads over constant-amplitude cycles until failure.

`repair` is from Hsieh (1990). This is a maintenance data set on active repair times (in hours) for an airborne communications transceiver.

Usage

```
data(BSdata)
```

References

Birnbaum, Z. W. and Saunders, S. C. (1969). A new family of life distributions. J. Appl. Probab. 6(2): 637-652.

McCool, J. I. (1974). Inferential techniques for Weibull populations. Aerospace Research Laboratories Report ARL T R74-0180, Wright-Patterson Air Force Base, Dayton, OH.

Rieck, J. R. and Nedelman, J. (1991). A Log-Linear Model for the Birnbaum-Saunders Distribution. Technometrics. 33, 51-60.

Brown, M. W. and Miller, K. J. (1978). Biaxial Fatigue Data. Report CEMR1/78. University of Sheffield, Dept. of Mechanical Engineering.

Hsieh, H. K. (1990). Estimating the Critical Time of Inverse Gaussian Hazard Rate. IEEE Transactions on Reliability, 39(10): 342-345.

Examples

```
# Attach data sets
data(BSdata)
```

CVM2gof — *Two-Sample Cramér-von Mises Permutation Test*

Description

Two-sample Cramér-von Mises goodness-of-fit test using permutation null distribution. Returns `htest` object.

Usage

```
CVM2gof(
  x,
  y,
  alternative = c("two.sided", "less", "greater"),
  nboots = 2000,
  keep.boots = FALSE
)
```

Arguments

`x`	Numeric vector, first sample.
`y`	Numeric vector, second sample.
`alternative`	Character string specifying alternative hypothesis. One of `"two.sided"`, `"less"`, `"greater"`; default "two.sided".
`nboots`	Integer, number of permutation replicates, default 2000.
`keep.boots`	Logical; whether save all permutation statistics, default FALSE.

Value

An `htest` list:

statistic Observed CvM statistic.

p.value Permutation p-value.

alternative Alternative hypothesis.

method Test name.

data.name Sample names.

bootstraps Permutation statistics vector, only for keep.boots=TRUE.

Examples

```
set.seed(123)
x <- rnorm(100)
y <- rnorm(100, mean=0.2)
out <- CVM2gof(x, y, nboots = 200)
out$p.value
```

`CVMgof` *One-Sample Cramér–von Mises Test for Normality*

Description

Performs the one-sample Cramér–von Mises goodness-of-fit test for normality, where the mean and standard deviation of the normal distribution are estimated from the data. The p-value is computed using asymptotic approximations.

Usage

```
CVMgof(x, eps = 1e-15)
```

Arguments

`x`	A numeric vector of observations.
`eps`	A small positive value used to truncate extreme cumulative probabilities to avoid numerical instability (default: `1e-15`).

Details

This function tests the null hypothesis that the sample comes from a normal distribution, with unknown parameters estimated by the sample mean and standard deviation.

The Cramér–von Mises statistic is computed as:

$$W = \frac{1}{12n} + \sum_{i=1}^{n} \left(F(x_{(i)}) - \frac{2i-1}{2n} \right)^2,$$

where F is the CDF of the fitted normal distribution. A small-sample correction $(1 + 0.5/n) \cdot W$ is applied before computing the p-value.

The p-value is obtained via piecewise exponential-polynomial approximations based on the asymptotic distribution under the composite null hypothesis (i.e., when parameters are estimated from the data). These approximations are accurate for moderate to large sample sizes but remain approximate.

Note: Because the parameters are estimated from the data, this is not the classical Cramér–von Mises test (which assumes a fully specified distribution).

Value

An object of class `"htest"` with the following components:

statistic The corrected Cramér–von Mises test statistic (`W`).

p.value The approximated asymptotic p-value.

method A character string describing the test.

data.name The name of the input data vector.

parameter Named vector containing the estimated `mean` and `sd`.

Examples

```
set.seed(123)
x <- rnorm(50)
CVMgof(x)
```

DAD

DAD This function implements a resampling-based AD goodness-of-fit test. It draws K bootstrap subsamples, performs AD goodness-of-fit test on each subsample, returns the average AD statistic and the minimum p-value across all K subsamples.

Description

DAD This function implements a resampling-based AD goodness-of-fit test. It draws K bootstrap subsamples, performs AD goodness-of-fit test on each subsample, returns the average AD statistic and the minimum p-value across all K subsamples.

Usage

```
DAD(X, K)
```

Arguments

X	numeric vector. Observed sample for goodness-of-fit testing.
K	integer. Number of bootstrap subsamples.

Value

A list containing two elements:

stat scalar, average of the K AD test statistics.

pval scalar, minimum p-value from the K subsample AD tests.

See Also

ADgof

Examples

```
## Not run:
set.seed(123)
X <- rnorm(200)
DAD(X, K = 5)

## End(Not run)
```

DAD2 — *Distributed Two-Sample Anderson-Darling Test via Fisher Combination*

Description

Implements a distributed two-sample Anderson-Darling (AD) test: randomly draw subsets from samples into K partitions, run local bootstrap AD test on each subset pair, then combine partition p-values using Fisher's method.

Usage

```
DAD2(x, y, K, nboots = 50)
```

Arguments

x	Numeric vector, first sample.
y	Numeric vector, second sample.
K	Integer, number of random partitions.
nboots	Integer, number of bootstrap replicates for each local two-sample AD test; default 50.

Value

A list containing:

stat Mean of local AD statistics across all K partitions.

pval Combined p-value obtained from Fisher's combination method.

method Character string describing the test method.

partition_pvals Numeric vector of p-values from each partition.

partition_stats Numeric vector of AD statistics from each partition.

Examples

```
set.seed(123)
x <- rnorm(200)
y <- rnorm(200, mean = 0.3)
out <- DAD2(x, y, K = 3, nboots = 30)
out$pval
```

DCVM — *DCVM This function implements a resampling-based CVM goodness-of-fit test. It draws K bootstrap subsamples, performs CVM goodness-of-fit test on each subsample, returns the average CVM statistic and the minimum p-value across all K subsamples.*

Description

DCVM This function implements a resampling-based CVM goodness-of-fit test. It draws K bootstrap subsamples, performs CVM goodness-of-fit test on each subsample, returns the average CVM statistic and the minimum p-value across all K subsamples.

Usage

```
DCVM(X, K)
```

Arguments

X	numeric vector. Observed sample for goodness-of-fit testing.
K	integer. Number of bootstrap subsamples.

Value

A list containing two elements:

stat scalar, average of the K CVM test statistics.

pval scalar, minimum p-value from the K subsample CVM tests.

See Also

CVMgof

Examples

```
## Not run:
set.seed(123)
X <- rnorm(200)
DCVM(X, K = 5)

## End(Not run)
```

DCVM2 — *Distributed Two-Sample Cramér-von Mises Test via Fisher Combination*

Description

Implements a distributed two-sample Cramér-von Mises (CvM) test: randomly split samples into K subsets, run local bootstrap CvM test on each partition, then combine the partition p-values using Fisher's method.

Usage

```
DCVM2(x, y, K, nboots = 50)
```

Arguments

x	Numeric vector, first sample.
y	Numeric vector, second sample.
K	Integer, number of random partitions.
nboots	Integer, number of bootstrap replicates for each local two-sample CvM test; default 50.

Value

A list with components:

stat Mean of the local CvM statistics across all K partitions.

pval Combined p-value from Fisher's method.

method Character string describing the test method.

partition_pvals Numeric vector of p-values from each partition.

partition_stats Numeric vector of CvM statistics from each partition.

Examples

```
set.seed(123)
x <- rnorm(200)
y <- rnorm(200, mean = 0.3)
out <- DCVM2(x, y, K = 3, nboots = 30)
out$pval
```

DKS — *DKS This function implements a resampling-based KS test. It draws K bootstrap subsamples, performs KS goodness-of-fit test on each subsample, returns the average KS statistic and the minimum p-value across all K subsamples.*

Description

DKS This function implements a resampling-based KS test. It draws K bootstrap subsamples, performs KS goodness-of-fit test on each subsample, returns the average KS statistic and the minimum p-value across all K subsamples.

Usage

```
DKS(X, K)
```

Arguments

X	numeric vector. Observed sample for goodness-of-fit testing.
K	integer. Number of bootstrap subsamples.

Value

A list containing two elements:

stat scalar, average of the K KS test statistics.

pval scalar, minimum p-value from the K subsample KS tests.

See Also

KSgof

Examples

```
## Not run:
set.seed(123)
X <- rnorm(200)
DKS(X, K = 5)

## End(Not run)
```

DKS2 *Distributed Two-Sample KS Test via Fisher Combination*

Description

Implements a distributed two-sample KS test: randomly split samples into K subsets, run bootstrap KS test on each partition, then combine partition p-values using Fisher's method.

Usage

```
DKS2(x, y, K, nboots = 50)
```

Arguments

x	Numeric vector, first sample.
y	Numeric vector, second sample.
K	Integer, number of random partitions.
nboots	Integer, number of bootstrap replicates for each local KS test, default 50.

Value

A list containing:

stat Mean of KS statistics across all K partitions.

pval Combined p-value from Fisher's method.

method Character string describing test method.

partition_pvals Vector of p-values from each partition.

partition_stats Vector of KS statistics from each partition.

Examples

```
set.seed(123)
x <- rnorm(200)
y <- rnorm(200, mean = 0.3)
out <- DKS2(x, y, K = 3, nboots = 30)
out$pval
```

DPECVM *DPECVM Performs the divided PECVM goodness-of-fit test. The original sample is split into K bootstrap subsamples, PECVM test is applied on each subsample. The final test statistic is the average of subsample statistics, and the p-value is taken as the minimum of subsample p-values.*

Description

DPECVM Performs the divided PECVM goodness-of-fit test. The original sample is split into K bootstrap subsamples, PECVM test is applied on each subsample. The final test statistic is the average of subsample statistics, and the p-value is taken as the minimum of subsample p-values.

Usage

```
DPECVM(X, K, F0, rF0)
```

Arguments

X	A numeric vector of observed sample data.
K	Integer, number of subsamples to split.
F0	Cumulative distribution function of the null distribution.
rF0	Random number generator function for the null distribution.

Value

A list containing two elements:

stat Numeric, averaged PECVM test statistic across K subsamples.

pval Numeric, minimum p-value from K subsample PECVM tests.

Examples

```
## Not run:
# Null: standard normal
set.seed(123)
X <- rnorm(200)
F0 <- pnorm
rF0 <- rnorm
out <- DPECVM(X, K = 5, F0 = F0, rF0 = rF0)
out$stat
out$pval

## End(Not run)
```

DPECVM2 *Distributed Power-Enhanced Two-Sample Cramer-von Mises Test*

Description

Implements a distributed power-enhanced two-sample Cramer-von Mises (PE-CvM) test. Randomly draw subsets from samples into K partitions, compute local power-enhanced CvM test for each partition via `PE2CVM`, then combine partition p-values using Fisher's combination method.

Usage

```
DPECVM2(
  x,
  y,
  K,
  lambda = 0.2,
  threshold = 0.99,
  B = 100,
  nboots = 50,
  na.rm = FALSE
)
```

Arguments

`x`	Numeric vector, first sample.
`y`	Numeric vector, second sample.
`K`	Integer, number of random partitions.
`lambda`	Numeric, power enhancement parameter, default 0.2.
`threshold`	Numeric, threshold parameter for power enhancement, default 0.99.
`B`	Integer or `NULL`, number of bootstrap replicates inside local test.
`nboots`	Integer, bootstrap replicates for local test, default 50.
`na.rm`	Logical. If `TRUE`, remove missing values; default `FALSE`.

Value

A list of class `htest` compatible structure containing:

statistic Named numeric vector with mean values across partitions: Enhanced, Original.CvM, PE.Component.

p.value Combined p-value obtained by Fisher's combination of K partition p-values.

method Character string describing the test name.

data.name Character string naming the input samples `x` and `y`.

parameters Named vector storing hyperparameters: K, lambda, threshold, B.

partition_pvals Numeric vector of p-values from each partition.

partition_stats Data frame containing three types of statistics for each partition.

Examples

```
set.seed(123)
x <- rnorm(300)
y <- rnorm(300, mean = 0.4)
out <- DPECVM2(x, y, K = 3, lambda = 0.2, threshold = 0.99, B = 30)
out$p.value
out$statistic
```

---

goats *Zoometric measurements of goats*

Description

Zoometric measurements of 27 week old creole goats collected by Dorantes-Coronado (2013).

Usage

```
data(goats)
```

Format

A data frame with 52 rows and 7 columns containing measurements (in kilograms and centimeters) on the following variables.

`body.weight`

`body.length`

`trunk.length`

`withers.height`

`thoracic.perimeter`

`hip.length`

`ear.length`

Source

Dorantes-Coronado (2013).

References

Dorantes-Coronado, E.J. (2013). Estudio preliminar para el establecimiento de un programa de mejoramiento genetico de cabras en el Estado de Mexico. Ph.D. Thesis. Colegio de Postgraduados, Mexico.

Examples

```
data(goats)
plot(goats)
```

`KS2gof` *Two-Sample Kolmogorov-Smirnov Test with Bootstrap/Permutation*

Description

Compute two-sample Kolmogorov-Smirnov test using permutation bootstrap to calculate p-value. Returns an object of class `htest`.

Usage

```
KS2gof(
  x,
  y,
  alternative = c("two.sided", "less", "greater"),
  nboots = 5000,
  keep.boots = FALSE
)
```

Arguments

`x`	Numeric vector, first sample.
`y`	Numeric vector, second sample.
`alternative`	Character string specifying alternative hypothesis. Must be one of `"two.sided"` (default), `"less"`, `"greater"`.
`nboots`	Integer, number of permutation bootstrap replicates, default 5000.
`keep.boots`	Logical; if TRUE, store all bootstrap test statistics in output, default FALSE.

Value

An object of class `htest`, a list containing:

statistic Observed KS statistic.

p.value Permutation p-value.

alternative Specified alternative hypothesis.

method Name of test method.

bootstraps Vector of bootstrap statistics, only present if `keep.boots = TRUE`.

Examples

```
set.seed(123)
x <- rnorm(100)
y <- rnorm(100, mean=0.2)
out <- KS2gof(x, y, alternative = "two.sided", nboots = 200)
out$p.value
```

KSgof — *One-sample Kolmogorov-Smirnov goodness-of-fit test*

Description

Performs the one-sample Kolmogorov-Smirnov test against the normal distribution with parameters estimated from the data.

Usage

```
KSgof(x, eps = 1e-15)
```

Arguments

`x`	Numeric vector of observations.
`eps`	Numeric lower and upper bound for tail probabilities to avoid numerical issues (default: `1e-15`).

Details

The test compares the empirical distribution function of `x` with the cumulative distribution function of a normal distribution (mean and standard deviation estimated from `x`) using the Kolmogorov-Smirnov statistic. For large sample sizes, a p-value approximation based on the asymptotic distribution is used.

A correction is applied when sample size exceeds 100, adjusting the test statistic to approximate a fixed sample size. For very small or very large statistics, piecewise polynomial approximations are used to compute the p-value.

Value

An object of class "htest" containing the test statistic, p-value, method description, and data name.

Examples

```
set.seed(123)
x <- rnorm(100)
KSgof(x)
```

Kuiper2gof — *Two-Sample Kuiper Test with Bootstrap*

Description

Performs a two-sample Kuiper test using bootstrap resampling to test whether two independent samples come from the same distribution.

Usage

```
Kuiper2gof(
  x,
  y,
  alternative = c("two.sided", "less", "greater"),
  nboots = 2000,
  keep.boots = FALSE
)
```

Arguments

x, y	Numeric vectors of data values for the two samples.
alternative	Character string indicating the alternative hypothesis. Must be one of "two.sided", "less", or "greater".
nboots	Integer. Number of bootstrap resamples to compute the empirical null distribution (default: 2000).
keep.boots	Logical. If TRUE, returns all bootstrap test statistics.

Details

The Kuiper test is a nonparametric test similar to the Kolmogorov–Smirnov test, but sensitive to discrepancies in both location and shape between two distributions. This implementation uses bootstrap resampling to estimate the p-value.

The two.sided test uses the sum of maximum positive and negative ECDF differences. The greater and less options use one-sided variations.

If the observed test statistic exceeds all bootstrap values, the p-value is set to 1 / (2 * nboots) to avoid zero.

Value

An object of class "htest" containing:

statistic The observed Kuiper statistic.

p.value The p-value computed from the bootstrap distribution.

alternative The specified alternative hypothesis.

method A character string describing the test.

bootstraps (If requested) A numeric vector of bootstrap statistics.

Examples

```
set.seed(123)
x <- rnorm(100, 0, 4)
y <- rnorm(100, 2, 4)
Kuiper2gof(x, y)
```

PE2AD *Power-Enhanced Two-Sample Anderson-Darling Test (PE2AD)*

Description

Performs a power-enhanced variant of the two-sample Anderson-Darling (AD) test. The proposed statistic augments the classical two-sample AD statistic by adding a penalty term constructed from extreme weighted deviations between empirical distribution functions. The proposed enhancement improves statistical power for detecting distributional differences concentrated in the tails. While the standard AD test already places higher weight on tail discrepancies, this method explicitly amplifies the contribution of extreme local deviations.

Usage

```
PE2AD(x, y, lambda = 1, threshold = 1, B = NULL)
```

Arguments

x	Numeric vector of observations from the first sample.
y	Numeric vector of observations from the second sample.
lambda	Non-negative numeric scalar controlling the magnitude of the power-enhancement penalty component (default = 1). Larger values increase sensitivity to extreme outlier-like local deviations.
threshold	Numeric scalar defining the cutoff for standardized marginal contributions to be treated as extreme (default = 1.0, corresponding to one standard deviation above the mean weighted deviation).
B	Integer specifying the number of permutation replicates for computing the permutation p-value. If NULL or B <= 0, permutation inference is skipped and p.value returns NA. Defaults to NULL.

Details

The classical two-sample Anderson-Darling statistic is defined as

$$A^2 = \frac{n_1 n_2}{(n_1+n_2)^2} \sum_{i=1}^{N} \frac{[F_{n_1}(z_i) - G_{n_2}(z_i)]^2}{H_N(z_i)\,[1 - H_N(z_i)]},$$

where

- z_i: pooled and sorted combined observations from the two samples,
- F_{n_1}, G_{n_2}: empirical cumulative distribution functions for sample x and sample y,
- H_N: pooled empirical CDF under the null hypothesis of identical distributions.

The denominator $H_N(1-H_N)$ assigns larger weights to tail deviations, which makes AD test more sensitive to tail differences than Kolmogorov-Smirnov or Cramer-von Mises tests.

The PE2AD statistic is constructed in four steps:

1. Compute pointwise weighted squared deviations (summands of the A^2 statistic).
2. Standardize these marginal contributions: $s_i = (m_i - \bar{m})/\mathrm{sd}(m)$.
3. Retain only points satisfying $s_i >$ `threshold`.
4. Compute the PE component as the mean of squared standardized selected terms, which ensures scale compatibility with the original AD statistic.

The final enhanced test statistic:

$$T = A^2 + \lambda \times \text{PE_comp}.$$

Under the null hypothesis that the two samples come from identical distributions, permutation inference randomly reassigns group labels while fixing the pooled ordered grid z. The permutation p-value is estimated by the fraction of permutation replicates whose simulated statistic is greater than or equal to the observed T.

Numerical stabilization: pooled CDF H_N is truncated to $[\epsilon, 1-\epsilon]$ where ϵ equals `.Machine$double.eps` to avoid division by zero near the extreme tails.

Value

An object of S3 class `"htest"`, a list containing the following elements:

statistic Named numeric vector with three entries: `Enhanced` (final PE2AD test statistic), `Original.AD` (classical two-sample Anderson-Darling statistic), `PE.Component` (power-enhancement penalty term).

p.value Permutation-based p-value when `B > 0`; otherwise `NA`.

method Character string with the full test name.

data.name Character string naming the two input samples.

parameters Named numeric vector storing user-supplied `lambda`, `threshold`, and `B`.

Note

Permutation p-value computation can be computationally expensive for large B. The permutation scheme fixes the pooled ordered grid for efficiency. When all observations are identical across the pooled sample, marginal contributions become zero and the PE component returns zero.

References

Anderson, T. W., & Darling, D. A. (1952). Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes. *Annals of Mathematical Statistics*.

Examples

```
set.seed(2025)
x <- rnorm(60, mean = 0, sd = 1)      # light-tailed Gaussian sample
y <- rt(60, df = 3)                   # heavy-tailed t-distribution sample

# PE2AD with permutation p-value
PE2AD(x, y, lambda = 0.4, threshold = 1.0, B = 2000)

# Quick calculation without permutation p-value
PE2AD(x, y)
```

PE2CVM *Power-Enhanced Two-Sample Cramer-von Mises Test (PE2CVM)*

Description

Performs a power-enhanced variant of the two-sample Cramer-von Mises (CvM) test. The enhanced statistic combines the classical two-sample CvM statistic with a power-enhancement component constructed from large squared deviations between empirical distribution functions. This modification improves detection power for subtle distributional differences compared with the standard CvM test.

Usage

```
PE2CVM(x, y, lambda = 0.4, threshold = 0.9, B = NULL, na.rm = FALSE)
```

Arguments

x	Numeric vector of observations from the first sample.
y	Numeric vector of observations from the second sample.
lambda	Non-negative numeric scalar for the power enhancement coefficient (default = 0.4). Larger values amplify the contribution of the power-enhancement term.
threshold	Numeric scalar in $[0, 1]$ specifying the quantile threshold for selecting high-energy deviation points (default = 0.9).

B	Integer or NULL; number of permutation replicates for constructing the null distribution. If NULL or B <= 0, permutation inference is skipped and p.value returns NA.
na.rm	Logical. If TRUE, missing values are removed before computation; if FALSE, the presence of NA triggers an error (default = FALSE).

Details

The classical two-sample Cramer-von Mises statistic is defined as

$$W^2 = \frac{n_1 n_2}{(n_1+n_2)^2} \sum_{i=1}^{N} \left[F_{n_1}(z_i) - G_{n_2}(z_i)\right]^2,$$

where

- z_i: pooled sorted observations from the two samples,
- F_{n_1}, G_{n_2}: empirical cumulative distribution functions for sample x and sample y,
- n_1, n_2: sample sizes, $N = n_1 + n_2$.

The pointwise squared deviation $(F_{n_1}(z_i) - G_{n_2}(z_i))^2$ is treated as the local energy term. The energy values are standardized using median and MAD for robustness against outliers. Points whose standardized energy exceeds the quantile specified by threshold are selected as high-energy locations.

The raw power-enhancement term is the mean of squared standardized values among selected high-energy points, and then mapped via a sigmoid-like transformation:

$$\mathrm{PE}_{\text{scaled}} = \frac{2}{1+\exp(-\mathrm{PE}_{\text{raw}})} - 1.$$

The final enhanced test statistic is

$$T = W^2 + \lambda \times \mathrm{PE}_{\text{scaled}}.$$

Under the null hypothesis of identical distributions, permutation inference randomly reassigns group labels while keeping the pooled ordered grid z fixed. The permutation p-value is estimated as the proportion of permutation replicates with simulated statistic greater than or equal to the observed T.

Value

An object of S3 class "htest", a list containing:

statistic Named numeric vector: Enhanced (final combined test statistic), Original.CVM (classical two-sample Cramer-von Mises statistic), PE.Component (scaled power-enhancement component).

p.value Permutation-based p-value; NA when permutation is not performed.

method Character string with the full test name.

data.name Character string naming the two input samples.

parameters Named numeric vector storing hyperparameters lambda, threshold, and B.

Note

Permutation p-value computation is computationally costly when `B` is large. The quantile threshold `threshold` must lie between 0 and 1. If all local energy terms are identical, MAD equals zero and is replaced by 1 to avoid division by zero.

References

Cramer, H. (1928). On the composition of elementary errors. *Skandinavisk Aktuarietidskrift*. von Mises, R. (1928). Wahrscheinlichkeitsrechnung und ihre Anwendung in der Statistik und theoretischen Physik.

Examples

```
set.seed(123)
x <- rnorm(100)
y <- rnorm(100, mean = 0.2)
out <- PE2CVM(x, y, lambda = 0.4, threshold = 0.9, B = 200)
out$p.value
out$statistic

# Quick run without permutation p-value
PE2CVM(x, y, lambda = 0.4, threshold = 0.9)
```

PE2KS — *Power-Enhanced Two-Sample Kolmogorov–Smirnov Test (PE2KS)*

Description

Performs a power-enhanced version of the two-sample Kolmogorov–Smirnov (KS) test that augments the classical KS statistic with a penalty term based on unusually large deviations across the entire empirical distribution function (EDF). This enhancement improves sensitivity to differences in shape or tail behavior that may not produce a large maximum deviation.

Usage

```
PE2KS(x, y, lambda = 1, threshold = 1, B = NULL)
```

Arguments

`x`	A numeric vector of observations from the first sample.
`y`	A numeric vector of observations from the second sample.
`lambda`	Numeric scalar ≥ 0 controlling the weight of the power-enhancement component (default: 1). Larger values increase emphasis on widespread moderate discrepancies.
`threshold`	Numeric scalar defining the standardized deviation threshold above which points are considered "extreme" and contribute to the enhancement term (default: 1.0, i.e., 1 standard deviation above the mean deviation).

B Integer specifying the number of permutation replicates for p-value computation. If NULL or B <= 0, no p-value is computed (default: NULL).

Details

The classical two-sample KS test uses the statistic

$$D_{\text{KS}} = \sup_t |F_n(t) - G_m(t)|,$$

which is sensitive primarily to the largest pointwise difference between empirical CDFs.

The PE2KS test enhances this by adding a term that captures **systematic or tail deviations**:

1. Compute the absolute deviation vector $d_i = |F_n(z_i) - G_m(z_i)|$ over all pooled ordered points z_i.
2. Standardize these deviations: $s_i = (d_i - \bar{d})/\text{sd}(d)$.
3. Select points where $s_i >$ threshold.
4. Define the power-enhancement (PE) component as the **mean squared standardized deviation** over selected points (or 0 if none are selected).

The final test statistic is:

$$T = D_{\text{KS}} + \lambda \times \text{PE_comp}.$$

A permutation procedure (if B > 0) randomly reassigns group labels to approximate the null distribution of T, yielding a p-value as the proportion of permuted statistics exceeding the observed value.

Missing values are not explicitly handled; users should ensure inputs are clean.

Value

An object of class "htest" containing:

statistic A named vector with three components: Enhanced (the final test statistic), Original.KS (classical KS statistic), and PE.Component (the power-enhancement term).

p.value Permutation-based p-value if B > 0; otherwise NA.

method A character string describing the test.

data.name Names of the input data vectors.

parameters Named vector of user-specified parameters (lambda, threshold, B).

Note

- The enhancement term is scale-normalized (via standardization and averaging) to be comparable in magnitude to D_{KS} (which lies in [0,1]). - Setting lambda = 0 recovers the standard KS test. - Very high lambda may reduce specificity; recommended values are in [0.1, 2].

Examples

```
set.seed(42)
x <- rnorm(60, mean = 0, sd = 1)
y <- rnorm(60, mean = 0.25, sd = 1)

# Basic usage (no p-value)
PE2KS(x, y)

# With permutation p-value
PE2KS(x, y, lambda = 0.5, threshold = 1.0, B = 2000)
```

PECVM *Power-Enhanced Cramer-von Mises Goodness-of-Fit Test*

Description

Performs the Power-Enhanced Cramer-von Mises (PECVM) goodness-of-fit test, which improves testing power by standardizing a Cramer-von Mises-type statistic with an estimate of its standard deviation under the null hypothesis. The null distribution must be fully specified via a cumulative distribution function (CDF).

Performs the Power-Enhanced Cramér–von Mises (PECVM) goodness-of-fit test, which improves testing power by standardizing a Cramér–von Mises-type statistic with an estimate of its standard deviation under the null hypothesis. The null distribution must be fully specified via a cumulative distribution function (CDF).

Usage

```
PECVM(x, F0, alpha = 0.05, B = 500, boot = TRUE)

PECVM(x, F0, alpha = 0.05, B = 500, boot = TRUE)
```

Arguments

`x`	A numeric vector of observed data.
`F0`	A function representing the theoretical cumulative distribution function (CDF) under the null hypothesis. It must accept a numeric vector and return probabilities in `[0, 1]` (e.g., `function(x) pnorm(x)` for standard normal).
`alpha`	Significance level for critical value estimation (default: `0.05`).
`B`	Number of bootstrap replicates used to compute the p-value and critical value (default: `500`).
`boot`	Logical. If `TRUE` (default), uses bootstrap to compute the p-value and critical value; if `FALSE`, returns `NA` for these (not implemented via asymptotics).

Details

The base statistic is defined as:

$$T_n = n \cdot \frac{1}{n} \sum_{i=1}^{n} \left(\hat{F}_n(x_i) - F_0(x_i) \right)^2 = \sum_{i=1}^{n} \left(\hat{F}_n(x_i) - F_0(x_i) \right)^2,$$

where $\hat{F}_n$ is the empirical CDF and F_0 is the user-specified null CDF.

The PECVM test enhances power by standardizing T_n with $\hat{\sigma}_{02}$, the bootstrap-estimated standard deviation of T_n under the null. The standardized statistic is $T_n^{PE} = T_n/\hat{\sigma}_{02}$.

Bootstrap inference proceeds as follows:

1. Estimate $\hat{\sigma}_{02}$ using 500 resamples from the original data.
2. Generate `B` bootstrap replicates of $T^*/\hat{\sigma}_{02}$ (using the same $\hat{\sigma}_{02}$).
3. Compute the p-value as the proportion of bootstrap statistics exceeding the observed T_n^{PE}.

This test assumes a simple null hypothesis: the CDF `F0` must be completely known and must not depend on parameters estimated from `x`. Using a data-dependent `F0` (e.g., plugging in sample mean) invalidates the test.

The base statistic is defined as:

$$T_n = n \cdot \frac{1}{n} \sum_{i=1}^{n} \left(\hat{F}_n(x_i) - F_0(x_i) \right)^2 = \sum_{i=1}^{n} \left(\hat{F}_n(x_i) - F_0(x_i) \right)^2,$$

where $\hat{F}_n$ is the empirical CDF and F_0 is the user-specified null CDF.

The PECVM test enhances power by standardizing T_n with $\hat{\sigma}_{02}$, the bootstrap-estimated standard deviation of T_n under the null. The standardized statistic is $T_n^{PE} = T_n/\hat{\sigma}_{02}$.

Bootstrap inference proceeds as follows:

1. Estimate $\hat{\sigma}_{02}$ using 500 resamples from the original data.
2. Generate `B` bootstrap replicates of $T^*/\hat{\sigma}_{02}$ (using the same $\hat{\sigma}_{02}$).
3. Compute the p-value as the proportion of bootstrap statistics exceeding the observed T_n^{PE}.

This test assumes a **simple null hypothesis**: the CDF `F0` must be completely known and must not depend on parameters estimated from `x`. Using a data-dependent `F0` (e.g., plugging in sample mean) invalidates the test.

Value

A list with class `"htest"` containing:

statistic The power-enhanced Cramer-von Mises statistic $T_n^{PE} = T_n/\hat{\sigma}_{02}$.

p.value Bootstrap p-value (or `NA` if `boot = FALSE`).

critical.value `(1 - alpha)` quantile of the bootstrap distribution (or `NA`).

sigma Estimated standard deviation $\hat{\sigma}_{02}$ of the base CVM-type statistic.

n Sample size.

method Description of the test performed.

data.name Name of the input data vector.

A list with class "htest" containing:

statistic The power-enhanced Cramér–von Mises statistic $T_n^{PE} = T_n/\hat{\sigma}_{02}$.

p.value Bootstrap p-value (or NA if boot = FALSE).

critical.value (1 - alpha) quantile of the bootstrap distribution (or NA).

sigma Estimated standard deviation $\hat{\sigma}_{02}$ of the base CVM-type statistic.

n Sample size.

method Description of the test performed.

data.name Name of the input data vector.

Examples

```
set.seed(123)
x <- rnorm(100)               # Data generated under N(0,1)
F0 <- function(x) pnorm(x)    # Null hypothesis: standard normal CDF
res <- PECVM(x, F0 = F0, B = 999)
print(res)

set.seed(123)
x <- rnorm(100)               # Data generated under N(0,1)
F0 <- function(x) pnorm(x)    # Null hypothesis: standard normal CDF
res <- PECVM(x, F0 = F0, B = 999)
print(res)
```

PEKS *Power-Enhanced Kolmogorov–Smirnov Goodness-of-Fit Test*

Description

Performs the Power-Enhanced Kolmogorov–Smirnov (PEKS) goodness-of-fit test, which improves the power of the classical KS test by standardizing the test statistic with an estimate of its standard deviation under the null hypothesis. The null distribution must be fully specified via a cumulative distribution function (CDF).

Usage

```
PEKS(x, F0, alpha = 0.05, B = 500, boot = TRUE)
```

Arguments

x	A numeric vector of observed data.
F0	A function representing the theoretical cumulative distribution function (CDF) under the null hypothesis. It must accept a numeric vector and return probabilities in [0, 1] (e.g., function(x) pnorm(x) for standard normal).
alpha	Significance level for critical value estimation (default: 0.05).

B	Number of bootstrap replicates used to estimate the standard deviation of the KS statistic and to compute the p-value and critical value (default: 500).
boot	Logical. If TRUE (default), uses bootstrap to compute the p-value and critical value; if FALSE, returns NA for these (not yet implemented).

Details

The PEKS test modifies the classical Kolmogorov–Smirnov statistic D_n by dividing it by an estimate of its standard deviation $\hat{\sigma}_{01}$, obtained via nonparametric bootstrap from the original sample. This standardization enhances the test's power, particularly against local alternatives.

The bootstrap procedure involves:

1. Computing the original KS statistic D_n.
2. Estimating $\hat{\sigma}_{01}$ using 500 bootstrap resamples.
3. Generating B bootstrap replicates of the standardized statistic $D^*/\hat{\sigma}_{01}$.

The p-value is the proportion of bootstrap statistics exceeding the observed D_n^{PE}. Note that the null hypothesis must be **simple** (i.e., the CDF F0 is fully known); this test does not support composite hypotheses where parameters are estimated from the data.

Value

A list with class "htest" containing:

statistic The power-enhanced KS statistic $D_n^{PE} = D_n/\hat{\sigma}_{01}$.

p.value Bootstrap p-value (or NA if boot = FALSE).

critical.value (1 - alpha) quantile of the bootstrap distribution (or NA).

sigma Estimated standard deviation $\hat{\sigma}_{01}$ of the KS statistic.

n Sample size.

method Description of the test performed.

data.name Name of the input data vector.

Examples

```
set.seed(123)
x <- rnorm(100)               # Data from N(0,1)
F0 <- function(x) pnorm(x)    # Null: standard normal CDF
res <- PEKS(x, F0 = F0, B = 999)
print(res)
```

qCvMgof — *Calculate the Quantile of the Cramer-von Mises Goodness-of-Fit Statistic*

Description

This function calculates the quantile of the Cramer-von Mises goodness-of-fit statistic using the ‘uniroot‘ function to find the root of the given function.

Usage

```
qCvMgof(X, p)
```

Arguments

X	A numeric vector containing the sample data.
p	A numeric value representing the desired quantile probability.

Value

root	The quantile value corresponding to the given probability.

Examples

```
# Example usage:
set.seed(123)
X <- rnorm(100)  # Generate a sample from a normal distribution
p <- 0.95        # Desired quantile probability
result <- qCvMgof(X, p)
print(result)
```

run_custom_tests — *Run four goodness-of-fit tests for normality*

Description

Perform Anderson-Darling, Cramer-von Mises, Lilliefors (KS), and PECVM tests on a single sample. Return 0/1 indicators indicating whether the null hypothesis is rejected at level `alpha`.

Usage

```
run_custom_tests(X, F0 = pnorm, rF0 = rnorm, B_boot = 500, alpha = 0.05)
```

Arguments

X	A numeric vector of observations to be tested.
F0	Function, the cumulative distribution function (CDF) of the null distribution. Default pnorm (standard normal).
rF0	Function, the random number generator for the null distribution. Default rnorm (standard normal).
B_boot	Integer, number of bootstrap replicates used in the internal PECVM test. Default 500.
alpha	Numeric, significance level for the rejection decisions. Default 0.05.

Value

A named numeric vector of 0/1 indicators for AD, CVM, KS and PECVM.

See Also

ADgof, CVMgof, KSgof, PECVM

Examples

```
## Not run:
set.seed(123)
x <- rt(50, df = 5)
run_custom_tests(x)

## End(Not run)
```

---

run_FDP_sim — *Compute False Discovery Proportion (FDP) via Monte Carlo simulation for DKS, DCVM, DAD and DPECVM tests.*

Description

Compute False Discovery Proportion (FDP) via Monte Carlo simulation for DKS, DCVM, DAD and DPECVM tests.

Usage

```
run_FDP_sim(rgen, n_values, M, K, B, F0, rF0, alpha = 0.05, seed = 123)
```

Arguments

`rgen`	function, random sample generator, input n and return numeric vector.
`n_values`	numeric vector, candidate sample sizes.
`M`	integer, number of Monte Carlo replications.
`K`	integer, number of split subsets for D-type tests.
`B`	integer, bootstrap times for PECVM test.
`F0`	function, CDF of the null distribution.
`rF0`	function, random generator under null distribution.
`alpha`	numeric, significance level, default 0.05.
`seed`	integer, random seed for reproducibility, default 123.

Value

A data frame with columns:

n sample size

DKS average FDP of DKS test

DCVM average FDP of DCVM test

DAD average FDP of DAD test

DPECVM average FDP of DPECVM test

Examples

```
## Not run:
F0  <- function(x) pnorm(x,0,1)
rF0 <- function(n) rnorm(n,0,1)
rgen_t6 <- function(n) rt(n, df=6)
res <- run_FDP_sim(rgen = rgen_t6,
                   n_values = c(200,300),
                   M = 10, K = 2, B = 50,
                   F0 = F0, rF0 = rF0, seed=123)
print(res)

## End(Not run)
```

`simpleCvMgof` *Perform a Simple Cramer-von Mises Goodness-of-Fit Test*

Description

This function performs a simple Cramer-von Mises goodness-of-fit test. It assumes the input `X` are probability integral transformed values under the null distribution (uniform on [0,1]). The test statistic and an asymptotic approximate p-value are computed.

Usage

```
simpleCvMgof(X)
```

Arguments

X	A numeric vector containing probability integral transformed sample data (should lie in [0,1]).

Value

A list with class "htest" containing:

statistic The value of the Cramer-von Mises test statistic.

p.value Asymptotic approximate p-value for the test.

method Description of the test performed.

data.name Name of input data vector.

Examples

```
# Example usage: null uniform data
set.seed(123)
X <- runif(100)
result <- simpleCvMgof(X)
print(result)
```

Snow *Snow Dataset*

Description

Snowfall dataset

Format

vector of values

Details

This file contains observations of the annual snowfall amounts in Buffalo, New York. 63 as observed from 1910/11 to 1972/73 as listed in *The autoregressive method: a method of approximating and estimating positive functions. Carmichael, Jean-Pierre. DTIC Document. 1976*

`strength` *Compressive strength of maize seeds*

Description

Compressive strength and strain of maize seeds.

Usage

```
data("strength")
```

Format

A data frame with 90 observations on the following 2 variables.

`strain` a numeric vector giving the relative change in length under compression stress in millimeters.

`cstrength` a numeric vector giving the compressive strength in Newtons.

Details

These data correspond to maize seeds with floury endosperm and 8% of moisture.

Source

Mancera-Rico, A. (2014).

References

Mancera-Rico, A. (2014). Contenido de humedad y tipo de endospermo en la resistencia a compresion en semillas de maiz. Ph.D. Thesis. Colegio de Postgraduados, Mexico.

Examples

```
data(strength)
plot(strength)     # plot of "strain" versus "cstrength"
```

`Wgof` *Watson goodness-of-fit test Performs the Watson test for goodness-of-fit to a specified distribution.*

Description

Watson goodness-of-fit test Performs the Watson test for goodness-of-fit to a specified distribution.

Usage

```
Wgof(x, dist = c("norm", "exp", "unif", "lnorm", "gamma"), ..., eps = 1e-15)
```

Arguments

`x`	Numeric vector of observations.
`dist`	Character string specifying the distribution to test against. One of `"norm"`, `"exp"`, `"unif"`, `"lnorm"`, or `"gamma"`.
`...`	Additional parameters passed to the distribution's cumulative distribution function (CDF). For example, `mean` and `sd` for the normal distribution.
`eps`	Numeric tolerance for probability bounds to avoid extremes (default: 1e-15).

Details

The Watson test is a modification of the Cramér–von Mises test, adjusting for mean deviations. It measures the squared distance between the empirical distribution function of the data and the specified theoretical cumulative distribution function, with a correction for location.

Value

An object of class `"htest"` containing the test statistic, p-value, method description, data name, and any distribution parameters used.

Examples

```
set.seed(123)
x_norm <- rnorm(1000, mean = 5, sd = 2)
Wgof(x_norm, dist = "norm", mean = 5, sd = 2)

x_exp <- rexp(500, rate = 0.5)
Wgof(x_exp, dist = "exp", rate = 0.5)

x_unif <- runif(300, min = 0, max = 10)
Wgof(x_unif, dist = "unif", min = 0, max = 10)

x_lnorm <- rlnorm(200, meanlog = 0, sdlog = 1)
Wgof(x_lnorm, dist = "lnorm", meanlog = 0, sdlog = 1)

x_gamma <- rgamma(400, shape = 1, rate = 1)
```

```
Wgof(x_gamma, dist = "gamma", shape = 1, rate = 1)
```

WhiteWine *White Wine Quality Data*

Description

Physicochemical measurements and sensory quality ratings for white wine samples of the Portuguese Vinho Verde wine. Quality was evaluated by wine experts and is recorded as an ordered factor from 3 (poor) to 9 (excellent).

Usage

```
WhiteWine
```

Format

A data frame with 4898 rows and 12 variables:

quality Sensory quality score (ordered factor), levels 3 to 9; median of at least 3 expert evaluations.

fixed.acidity Fixed acidity, g(tartaric acid)/dm^3.

volatile.acidity Volatile acidity, g(acetic acid)/dm^3.

citric.acid Citric acid, g/dm^3.

residual.sugar Residual sugar, g/dm^3.

chlorides Chlorides, g(sodium chloride)/dm^3.

free.sulfur.dioxide Free sulfur dioxide, mg/dm^3.

total.sulfur.dioxide Total sulfur dioxide, mg/dm^3.

density Density, g/cm^3.

pH pH value.

sulphates Sulphates, g(potassium sulphate)/dm^3.

alcohol Alcohol content, percent by volume.

Source

UCI Machine Learning Repository, https://archive.ics.uci.edu/ml/datasets/wine+quality
P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis (2009). Modeling wine preferences by data mining from physicochemical properties. Decision Support Systems, 47(4):547-553.

Index

∗ **datasets**
- BSdata, 4
- goats, 15
- Snow, 32
- strength, 33
- WhiteWine, 35

AD2gof, 2
ADgof, 3, *8*, *30*

BSdata, 4

CVM2gof, 5
CVMgof, 6, *10*, *30*

DAD, 7
DAD2, 8
DCVM, 9
DCVM2, 10
DKS, 11
DKS2, 12
DPECVM, 13
DPECVM2, 14

goats, 15

KS2gof, 16
KSgof, *12*, 17, *30*
Kuiper2gof, 18

PE2AD, 19
PE2CVM, 21
PE2KS, 23
PECVM, 25, *30*
PEKS, 27

qCvMgof, 29

run_custom_tests, 29
run_FDP_sim, 30

simpleCvMgof, 31
Snow, 32
snow *(*Snow*)*, 32
strength, 33

Wgof, 34
WhiteWine, 35