Package 'ddst'

October 19, 2026

Title Data Driven Smooth Tests

Version 1.6.12

Description
Smooth tests are data driven (alternative hypothesis is dynamically selected based on data).
In this package you will find two groups of smooth of test: goodness-of-fit tests and
nonparametric tests for comparing distributions.
Among goodness-of-fit tests there are tests for exponent, Gaussian, Gumbel
and uniform distribution.
Among nonparametric tests there are tests for stochastic dominance,
k-sample test, test with umbrella alternatives and test for change-point problems.

License GPL-2

Encoding UTF-8

Depends R (>= 2.7), polynom

Imports ggplot2, orthopolynom, evd

Suggests rmutil, knitr

URL https://pbiecek.github.io/ddst/

BugReports https://github.com/pbiecek/ddst/issues

VignetteBuilder knitr

RoxygenNote 7.3.3

NeedsCompilation no

Author Przemyslaw Biecek [aut, cre],
Grzegorz Wylupek [aut]

Maintainer Przemyslaw Biecek <przemyslaw.biecek@gmail.com>

Repository CRAN

Date/Publication 2026-10-19 16:50:02 UTC

Contents

ddst.againststochdom.test 2
ddst.evd.test 3

ddst.exp.test 5
ddst.forstochdom.test 7
ddst.ksample.test 8
ddst.normbounded.test 10
ddst.normubounded.test 11
ddst.twosample.test 13
ddst.umbrellaknownp.test 15
ddst.umbrellaunknownp.test 16
ddst.uniform.test 18
ddst.upwardtrend.test 19
plot.ddst.test 21

Index **22**

`ddst.againststochdom.test`

Data Driven Smooth Test Against Stochastic Dominance

Description

Performs data driven smooth non-parametric two-sample test against one-sided alternatives (stochastic dominance). Suppose that we have random samples from two distributions F and G. The null hypothesis is that F(x) < G(x) for some x while the alternative is that at F(x) >= G(x) for all x with strict inequality for at least one x. Detailed description of the test statistic is provided in Ledwina and Wylupek (2012).

Usage

```
ddst.againststochdom.test(
  x,
  y,
  k.N = 4,
  alpha = 0.05,
  t,
  nr = 1e+05,
  compute.cv = FALSE
)
```

Arguments

`x`	a (non-empty) numeric vector of data
`y`	a (non-empty) numeric vector of data
`k.N`	an integer specifying a level of complexity of the grid considered, only for advanced users
`alpha`	a significance level
`t`	an alpha-dependent tunning parameter in the penalty in the model selection rule

nr	an integer specifying the number of runs for a p-value and a critical value computation if any
compute.cv	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not

References

Two-sample test against one-sided alternatives. Ledwina and Wylupek (2012). https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9469.2011.00787.x

Examples

```
set.seed(7)
# H0 is true
x <- runif(80)
y <- runif(80)
t <- ddst.againststochdom.test(x, y, alpha = 0.05, t = 2.2, k.N = 4)
t
plot(t)

# H0 is false
# known fixed alternative
x <- runif(80)
y <- rbeta(80,4,2)
t <- ddst.againststochdom.test(x, y, alpha = 0.05, t = 2.2, k.N = 4)
t
plot(t)
```

---

ddst.evd.test *Data Driven Smooth Test for Extreme Value Distribution*

Description

Performs data driven smooth test for composite hypothesis of extreme value distribution. Null density is given by $f(z;\gamma) = 1/\gamma_2 \exp((z-\gamma_1)/\gamma_2 - \exp((z-\gamma_1)/\gamma_2))$, $z \in R$.

Usage

```
ddst.evd.test(
  x,
  base = ddst.base.legendre,
  d.n = 10,
  c = 100,
  nr = 1e+05,
  compute.p = TRUE,
  alpha = 0.05,
  compute.cv = TRUE,
  ...
)
```

Arguments

x	a (non-empty) numeric vector of data values
base	a function which returns an orthonormal system, possible choice: ddst.base.legendre for the Legendre polynomials and ddst.base.cos for the cosine system
d.n	an integer specifying the maximum dimension considered, only for advanced users
c	a calibrating parameter in the penalty in the model selection rule
nr	an integer specifying the number of runs for a p-value and a critical value computation if any
compute.p	a logical value indicating whether to compute a p-value or not
alpha	a significance level
compute.cv	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not
...	further arguments

Details

We model alternatives similarly as in Kallenberg and Ledwina (1997) and Janic-Wroblewska (2004) using Legendre's polynomials or cosines. For more details see: http://www.biecek.pl/R/ddst/description.pdf.

Value

An object of class htest

statistic	the value of the test statistic.
parameter	the number of choosen coordinates (k).
method	a character string indicating the parameters of performed test.
data.name	a character string giving the name(s) of the data.
p.value	the p-value for the test, computed only if compute.p=TRUE.

References

Hosking, J.R.M., Wallis, J.R., Wood, E.F. (1985). Estimation of the generalized extreme-value distribution by the method of probability-weighted moments. *Technometrics* 27, 251–261.

Janic-Wroblewska, A. (2004). Data-driven smooth test for extreme value distribution. *Statistics* 38, 413–426.

Janic, A. and Ledwina, T. (2008). Data-driven tests for a location-scale family revisited. *J.Statist.Theory.Pract.Specialis*

Kallenberg, W.C.M., Ledwina, T. (1997). Data driven smooth tests for composite hypotheses: Comparison of powers. *J.Statist.Comput.Simul.* **59**, 101–121.

Examples

```
library(evd)
set.seed(7)

# for given vector of 19 numbers
z <- c(13.41, 6.04, 1.26, 3.67, -4.54, 2.92, 0.44, 12.93, 6.77, 10.09,
      4.10, 4.04, -1.97, 2.17, -5.38, -7.30, 4.75, 5.63, 8.84)
## Not run:
t <- ddst.evd.test(z, compute.p = TRUE, d.n = 10)
t
plot(t)

# H0 is true
x <- -qgumbel(runif(100),-1,1)
t <- ddst.evd.test (x, compute.p = TRUE, d.n = 10)
t
plot(t)

# H0 is false
x <- rexp(80,4)
t <- ddst.evd.test (x, compute.p = TRUE, d.n = 10)
t
plot(t)

## End(Not run)
```

`ddst.exp.test` *Data Driven Smooth Test for Exponentiality*

Description

Performs data driven smooth test for composite hypothesis of exponentiality. Null density is given by $f(z; gamma) = exp(-z/gamma)$ for z >= 0 and 0 otherwise. Modelling alternatives similarly as in Kallenberg and Ledwina (1997 a,b).

Usage

```
ddst.exp.test(
  x,
  base = ddst.base.legendre,
  d.n = 10,
  c = 100,
  nr = 1e+05,
  compute.p = TRUE,
  alpha = 0.05,
  compute.cv = TRUE,
  ...
)
```

Arguments

x	a (non-empty) numeric vector of data values
base	a function which returns an orthonormal system, possible choice: ddst.base.legendre for the Legendre polynomials and ddst.base.cos for the cosine system
d.n	an integer specifying the maximum dimension considered, only for advanced users
c	a calibrating parameter in the penalty in the model selection rule
nr	an integer specifying the number of runs for a p-value and a critical value computation if any
compute.p	a logical value indicating whether to compute a p-value or not
alpha	a significance level
compute.cv	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not
...	further arguments

Value

An object of class htest

statistic	the value of the test statistic.
parameter	the number of choosen coordinates (k).
method	a character string indicating the parameters of performed test.
data.name	a character string giving the name(s) of the data.
p.value	the p-value for the test, computed only if compute.p=T.

References

Kallenberg, W.C.M., Ledwina, T. (1997 a). Data driven smooth tests for composite hypotheses: Comparison of powers. $J.Statist.Comput.Simul.$ **59**, 101–121.

Kallenberg, W.C.M., Ledwina, T. (1997 b). Data driven smooth tests when the hypothesis is composite. $J.Amer.Statist.Assoc.$ **92**, 1094–1104.

Examples

```
set.seed(7)
# H0 is true
z <- rexp(80,4)
## Not run:
t <- ddst.exp.test (z, compute.p = TRUE, d.n = 10)
t
plot(t)

# H0 is false
z = rchisq(80,4)
(t = ddst.exp.test (z, compute.p = TRUE, d.n = 10))
t$p.value
```

```
plot(t)

## End(Not run)
```

`ddst.forstochdom.test` *Data Driven Smooth Test for Stochastic Dominance in Two Samples*

Description

Performs the data driven smooth test for detection of the stochastic ordering, as described in detail in Ledwina and Wyłupek (2012). Suppose that we have random samples from two distributions F and G. The null hypothesis is that F(x) >= G(x) for all x while the alternative is that at F(x) < G(x) for some x. Detailed description of the test statistic is provided in Ledwina and Wylupek (2012).

Usage

```
ddst.forstochdom.test(
  x,
  y,
  K.N = floor(log(length(x) + length(y), 2)) - 1,
  alpha = 0.05,
  t,
  nr = 1e+05,
  compute.p = TRUE,
  compute.cv = TRUE
)
```

Arguments

`x`	a (non-empty) numeric vector of data
`y`	a (non-empty) numeric vector of data
`K.N`	an integer specifying a level of complexity of the grid considered, only for advanced users
`alpha`	a significance level
`t`	an alpha-dependent tunning parameter in the penalty in the model selection rule
`nr`	an integer specifying the number of runs for a p-value and a critical value computation if any
`compute.p`	a logical value indicating whether to compute a p-value or not
`compute.cv`	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not

References

Nonparametric tests for stochastic ordering. Ledwina and Wyłupek (2012) <doi:10.1007/s11749-011-0278-7>

Examples

```
set.seed(7)
library("rmutil", warn.conflicts = FALSE)
# 1. Pareto(1)/Pareto(1.5)
# H0 is false
x <- rpareto(50, 2, 2)
y <- rpareto(50, 1.5, 1.5)
t <- ddst.forstochdom.test(x, y, t = 2.2, K.N = 4)
t
plot(t)

# 2. Laplace(0,1)/Laplace(1,25)
# H0 is false
x <- rlaplace(50, 0, 1)
y <- rlaplace(50, 1, 25)
t <- ddst.forstochdom.test(x, y, t = 2.2, K.N = 4)
t
plot(t)

# 3. LN(0.85,0.6)/LN(1.2,0.2)
# H0 is true
x <- rlnorm(50, 0.85, 0.6)
y <- rlnorm(50, 1.2, 0.2)
t <- ddst.forstochdom.test(x, y, t = 2.2, K.N = 4)
t
plot(t)

## Not run:
# Generate distribution of test statistic
N <- 1000
samp <- replicate(N, {
   x <- runif(30)
   y <- runif(30)
   # statistics with Schwartz penalty
   ddst.forstochdom.test(x, y)$statistic
})
quantile(samp, 0.95)
plot(ecdf(samp))

## End(Not run)
```

`ddst.ksample.test` *Data Driven Smooth Test for k-Sample Problem*

Description

Performs data driven smooth test for the classical k-sample problem. Suppose that we have random samples from k distributions F_i where i = 1, ..., k. The null hypothesis is that F_1 = ... = F_k while the alternative is that at least two distributions are different. Detailed description of the test statistic is provided in Wylupek (2010).

Usage

```
ddst.ksample.test(
  x,
  d.N = 12,
  c = 2.3,
  nr = 1e+05,
  compute.p = TRUE,
  alpha = 0.05,
  compute.cv = TRUE
)
```

Arguments

x	a list of k (non-empty) numeric vectors of data
d.N	an integer specifying the maximum dimension considered, only for advanced users
c	a calibrating parameter in the penalty in the model selection rule
nr	an integer specifying the number of runs for a p-value and a critical value computation if any
compute.p	a logical value indicating whether to compute a p-value or not
alpha	a significance level
compute.cv	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not

References

Data-driven k-sample tests. Wylupek (2010) https://www.jstor.org/stable/40586684?seq=1

Examples

```
set.seed(7)
# H0 is false
x <- runif(80)
y <- rexp(80, 1)
z <- runif(80)
t <- ddst.ksample.test(list(x, y, z))
t
plot(t)

# H0 is true
x <- runif(80)
y <- runif(80)
z <- runif(80)
t <- ddst.ksample.test(list(x, y, z))
t
plot(t)
```

`ddst.normbounded.test` *Data Driven Smooth Test for Normality; Bounded Basis Functions*

Description

Performs data driven smooth test for composite hypothesis of normality Null density is given by $f(z;\gamma) = 1/(\sqrt{2\pi}\gamma_2)\exp(-(z-\gamma_1)^2/(2\gamma_2^2))$ for $z \in R$. We model alternatives similarly as in Kallenberg and Ledwina (1997 a,b) using Legendre's polynomials or cosine basis.

Usage

```
ddst.normbounded.test(
  x,
  base = ddst.base.legendre,
  d.n = 10,
  c = 100,
  compute.p = TRUE,
  alpha = 0.05,
  compute.cv = TRUE,
  ...
)
```

Arguments

`x`	a (non-empty) numeric vector of data values
`base`	a function which returns an orthonormal system, possible choice: `ddst.base.legendre` for the Legendre polynomials and `ddst.base.cos` for the cosine system
`d.n`	an integer specifying the maximum dimension considered, only for advanced users
`c`	a calibrating parameter in the penalty in the model selection rule
`compute.p`	a logical value indicating whether to compute a p-value or not
`alpha`	a significance level
`compute.cv`	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not
`...`	further arguments

Value

An object of class `htest`

`statistic`	the value of the test statistic.
`parameter`	the number of choosen coordinates (k).
`method`	a character string indicating the parameters of performed test.
`data.name`	a character string giving the name(s) of the data.
`p.value`	the p-value for the test, computed only if `compute.p = TRUE`.

References

Chen, L., Shapiro, S.S. (1995). An alternative test for normality based on normalized spacings. *J. Statist. Comput. Simulation* 53, 269–288.

Inglot, T., Ledwina, T. (2006). Towards data driven selection of a penalty function for data driven Neyman tests. *Linear Algebra and its Appl.* **417**, 579–590.

Janic, A. and Ledwina, T. (2008). Data-driven tests for a location-scale family revisited. *J. Statist. Theory. Pract. Special issue on Modern Goodness of Fit Methods.*.

Kallenberg, W.C.M., Ledwina, T. (1997 a). Data driven smooth tests for composite hypotheses: Comparison of powers. *J. Statist. Comput. Simul.* **59**, 101–121.

Kallenberg, W.C.M., Ledwina, T. (1997 b). Data driven smooth tests when the hypothesis is composite. *J. Amer. Statist. Assoc.* **92**, 1094–1104.

Examples

```
set.seed(7)
# H0 is true
z <- rnorm(100)
# let's look on first 10 coordinates
d.n <- 10
t <- ddst.normbounded.test(z, compute.p = TRUE, d.n = d.n)
t
plot(t)

# H0 is false
z <- rexp(100, 1)
t <- ddst.normbounded.test(z, compute.p = TRUE, d.n = d.n)
t
plot(t)

# for Tephra data
z <- c(-1.748789, -1.75753, -1.740102, -1.740102, -1.731467, -1.765523,
      -1.761521, -1.72522, -1.80371, -1.745624, -1.872957, -1.729121,
      -1.81529, -1.888637, -1.887761, -1.881645, -1.91518, -1.849769,
      -1.755141, -1.665687, -1.764721, -1.736171, -1.736956, -1.737742,
      -1.687537, -1.804534, -1.790593, -1.808661, -1.784081, -1.729903,
      -1.711263, -1.748789, -1.772755, -1.72756, -1.71358, -1.821116,
      -1.839588, -1.839588, -1.830321, -1.807835, -1.747206, -1.788147,
      -1.759923, -1.786519, -1.726779, -1.738528, -1.754345, -1.781646,
      -1.641949, -1.755936, -1.775175, -1.736956, -1.705103, -1.743255,
      -1.82613, -1.826967, -1.780025, -1.684504, -1.751168)
t <- ddst.normbounded.test(z, compute.p = TRUE, d.n = d.n)
t
plot(t)
```

`ddst.normubounded.test`

Data Driven Smooth Test for Normality; Unbounded Basis Functions

---

Description

Performs data driven smooth test for composite hypothesis of normality. Null density is given by $f(z;\gamma) = 1/(\sqrt{2\pi}\gamma_2)\exp(-(z-\gamma_1)^2/(2\gamma_2^2))$ for $z \in R$.

Usage

```
ddst.normubounded.test(
  x,
  d.n = 20,
  e.0,
  v.0,
  r.alpha,
  s.n.alpha,
  alpha = 0.05,
  nr = 10000,
  compute.cv = TRUE
)

ddst.normunbounded.bias(n = 100, d.n = 20, nr = 10000)
```

Arguments

x	a (non-empty) numeric vector of data values
d.n	an integer specifying the maximum dimension considered, only for advanced users
e.0	a (non-empty) numeric vector being the Monte Carlo estimate of the mean of the vector (C2; ... ;Cd.n) calculated using the function ddst.normunbounded.bias()$e.0
v.0	a (non-empty) numeric vector being the Monte Carlo estimate of the variance of the vector (C2; ... ;Cd.n) calculated using the function ddst.normunbounded.bias()$e.0
r.alpha	a critical value of the alpha level R.n test
s.n.alpha	a penalty in the auxiliary model selection rule
alpha	a significance level
nr	an integer specifying the number of runs for a critical value computation
compute.cv	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not
n	sample size

References

Ledwina, T., Wyłupek, G. (2015) Detection of non-Gaussianity by Ledwina and Wyłupek *Journal of Statistical Computation and Simulation* 17, 3480-3497.

Examples

```
set.seed(7)
# H0 is true
z <- rnorm(100)
# let's look on first 20 coordinates
d.n <- 20

## Not run:
# calculate finite sample corrections
# see 6.2. Composite null hypothesis H in the appendix materials
e.v <- ddst.normunbounded.bias(n = length(z))
e.v

# simulated 1-alpha qunatiles, s(n, alpha) and  s.o(n, alpha)
# see Table 1 in the JSCS article
s <- 4.4
r.alpha <- 2.708

t <- ddst.normubounded.test(z, d.n, e.v$e.0, e.v$v.0, r.alpha, s)
t
plot(t)

# for Tephra data
z <- c(-1.748789, -1.75753, -1.740102, -1.740102, -1.731467, -1.765523,
       -1.761521, -1.72522, -1.80371, -1.745624, -1.872957, -1.729121,
       -1.81529, -1.888637, -1.887761, -1.881645, -1.91518, -1.849769,
       -1.755141, -1.665687, -1.764721, -1.736171, -1.736956, -1.737742,
       -1.687537, -1.804534, -1.790593, -1.808661, -1.784081, -1.729903,
       -1.711263, -1.748789, -1.772755, -1.72756, -1.71358, -1.821116,
       -1.839588, -1.839588, -1.830321, -1.807835, -1.747206, -1.788147,
       -1.759923, -1.786519, -1.726779, -1.738528, -1.754345, -1.781646,
       -1.641949, -1.755936, -1.775175, -1.736956, -1.705103, -1.743255,
       -1.82613, -1.826967, -1.780025, -1.684504, -1.751168)

# calculate finite sample corrections
e.v <- ddst.normunbounded.bias(n = length(z))
e.v

s <- 3.3
so <- 3.6
r.alpha <- 2.142

t <- ddst.normubounded.test(z, d.n, e.v$e.0, e.v$v.0, r.alpha, s)
t
plot(t)

## End(Not run)
```

`ddst.twosample.test` *Data Driven Smooth Test for Two-Sample Problem*

Description

Performs data driven smooth test for the classical two-sample problem. It is a special case of data driven test for k-samples. Detailed description of the test statistic is provided in Wylupek (2010).

Usage

```
ddst.twosample.test(
  x,
  y,
  d.N = 12,
  c = 2,
  B = 1e+05,
  compute.p = TRUE,
  alpha = 0.05,
  compute.cv = TRUE
)
```

Arguments

x	a (non-empty) numeric vector of data
y	a (non-empty) numeric vector of data
d.N	an integer specifying the maximum dimension considered, only for advanced users
c	a calibrating parameter in the penalty in the model selection rule
B	an integer specifying the number of runs for a p-value and a critical value computation if any
compute.p	a logical value indicating whether to compute a p-value or not
alpha	a significance level
compute.cv	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not

References

Data-driven k-sample tests. Wylupek (2010) https://www.jstor.org/stable/40586684?seq=1

Examples

```
set.seed(7)
# H0 is true
x <- runif(80)
y <- runif(80)
t <- ddst.twosample.test(x, y)
t
plot(t)

# H0 is false
x <- runif(80)
y <- rexp(80, 1)
```

```
t <- ddst.twosample.test(x, y)
t
plot(t)
```

`ddst.umbrellaknownp.test` *Data Driven Smooth Test for Umbrella Alternatives; Known Peak*

Description

Performs data driven smooth test for so-called umbrella alternatives in k-sample problem. Suppose that we have random samples from k distributions F_i where i = 1, ..., k. The null hypothesis is that there is no umbrella pattern, i.e. F_1 >= ... >= F_p <= ... <= F_k and F_i != F_j for some i and j. The alternative is that there is an umbrella pattern i.e. F_1 >= ... >= F_p <= ... <= F_k and F_i != F_j for some i and j. Detailed description of the test statistic is provided in Wylupek (2016).

Usage

```
ddst.umbrellaknownp.test(
  x,
  p,
  r.N = rep(4, length(x) - 1),
  alpha = 0.05,
  t.p,
  t.n,
  nr = 1e+05,
  compute.cv = TRUE
)
```

Arguments

`x`	a list of k (non-empty) numeric vectors of data
`p`	a peak of the umbrella pattern
`r.N`	a (p(p-1)=2+(k-p)(k-p+1)/2)-dimensional vector specifying the levels of complexity of the grids considered, only for advanced users
`alpha`	a significance level
`t.p`	an alpha-dependent (p(p-1)=2+(k-p)(k-p+1)/2)-dimensional vector of the tunning parameters in the penalties in the model selection rules T.o
`t.n`	an alpha-dependent (k-1)-dimensional vector of the tunning parameters in the penalties in the model selection rules T.tilde
`nr`	an integer specifying the number of runs for a p-value and a critical value computation if any
`compute.cv`	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not

References

An automatic test for the umbrella alternatives. Wylupek (2016) https://onlinelibrary.wiley.com/doi/abs/10.1111/sjos.12231

Examples

```
set.seed(7)
# H0 is true
x = runif(80)
y = runif(80) + 0.2
z = runif(80)
t <- ddst.umbrellaknownp.test(list(x, y, z), p = 2, t.p = 2.2, t.n = 2.2)
t
plot(t)

# known fixed alternative
x1 = rnorm(80)
x2 = rnorm(80) + 2
x3 = rnorm(80) + 4
x4 = rnorm(80) + 3
x5 = rnorm(80) + 2
x6 = rnorm(80) + 1
x7 = rnorm(80)
t <- ddst.umbrellaknownp.test(list(x1, x2, x3, x4, x5, x6, x7), p = 3, t.p = 2.2, t.n = 2.2)
t
plot(t)

t <- ddst.umbrellaknownp.test(list(x1, x2, x3, x4, x5, x6, x7), p = 5, t.p = 2.2, t.n = 2.2)
t
plot(t)
```

---

ddst.umbrellaunknownp.test

Data Driven Smooth Test for Umbrella Alternatives; Unknown Peak

---

Description

Data Driven Smooth Test for Umbrella Alternatives; Unknown Peak

Usage

```
ddst.umbrellaunknownp.test(
  x,
  r.N = rep(4, length(x) - 1),
  alpha = 0.05,
  t.p.aux,
  t.p,
  t.n,
```

```
  nr = 1e+05,
  compute.cv = TRUE
)
```

Arguments

x	a list of k (non-empty) numeric vectors of data
r.N	a (p(p-1)=2+(k-p)(k-p+1)/2)-dimensional vector specifying the levels of complexity of the grids considered, only for advanced users
alpha	a significance level
t.p.aux	an auxiliary alpha-dependent k x (k - 1) matrix of the tunning parameters in the penalties in the model selection rules To aux employed for estimation of the peak
t.p	an alpha-dependent (p(p-1)=2+(k-p)(k-p+1)/2)-dimensional vector of the tunning parameters in the penalties in the model selection rules T.o
t.n	an alpha-dependent (k-1)-dimensional vector of the tunning parameters in the penalties in the model selection rules T.tilde
nr	an integer specifying the number of runs for a p-value and a critical value computation if any
compute.cv	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not

References

An automatic test for the umbrella alternatives. Wylupek (2016) https://onlinelibrary.wiley.com/doi/abs/10.1111/sjos.12231

Examples

```
set.seed(7)
# H0 is true
x = runif(80)
y = runif(80) + 0.2
z = runif(80)
t <- ddst.umbrellaknownp.test(list(x, y, z), p = 2, t.p = 2.2, t.n = 2.2)
t
plot(t)

# known fixed alternative
x1 = rnorm(80)
x2 = rnorm(80) + 2
x3 = rnorm(80) + 4
x4 = rnorm(80) + 3
x5 = rnorm(80) + 2
x6 = rnorm(80) + 1
x7 = rnorm(80)
t <- ddst.umbrellaknownp.test(list(x1, x2, x3, x4, x5, x6, x7), p = 3, t.p = 2.2, t.n = 2.2)
t
plot(t)
```

```
t <- ddst.umbrellaknownp.test(list(x1, x2, x3, x4, x5, x6, x7), p = 5, t.p = 2.2, t.n = 2.2)
t
plot(t)
```

`ddst.uniform.test` *Data Driven Smooth Test for Uniformity*

Description

Performs data driven smooth tests for simple hypothesis of uniformity on [0,1]. Embeding null model into the original exponential family introduced by Neyman (1937).

Usage

```
ddst.uniform.test(
  x,
  base = ddst.base.legendre,
  d.n = 10,
  c = 2.4,
  nr = 1e+05,
  compute.p = TRUE,
  alpha = 0.05,
  compute.cv = TRUE,
  ...
)
```

Arguments

`x`	a (non-empty) numeric vector of data
`base`	a function which returns an orthonormal system, possible choice: `ddst.base.legendre` for the Legendre polynomials and `ddst.base.cos` for the cosine system
`d.n`	an integer specifying the maximum dimension considered, only for advanced users
`c`	a calibrating parameter in the penalty in the model selection rule
`nr`	an integer specifying the number of runs for a p-value and a critical value computation if any
`compute.p`	a logical value indicating whether to compute a p-value or not
`alpha`	a significance level
`compute.cv`	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not
`...`	further arguments

Value

An object of class `htest`

`statistic`	the value of the test statistic.
`parameter`	the number of choosen coordinates (k).
`method`	a character string indicating the parameters of performed test.
`data.name`	a character string giving the name(s) of the data.
`p.value`	the p-value for the test, computed only if `compute.p=T`.

References

Inglot, T., Ledwina, T. (2006). Towards data driven selection of a penalty function for data driven Neyman tests. *Linear Algebra and its Appl.* **417**, 579–590.

Ledwina, T. (1994). Data driven version of Neyman's smooth test of fit. *J. Amer. Statist. Assoc.* **89** 1000-1005.

Neyman, J. (1937). 'Smooth test' for goodness of fit. *Skand. Aktuarietidskr.* **20**, 149-199.

Examples

```
set.seed(7)
# H0 is true
z <- runif(80)
## Not run:
t <- ddst.uniform.test(z, compute.p = TRUE, d.n = 10)
t
plot(t)

# known fixed alternative
z <- rnorm(80,10,16)
t <- ddst.uniform.test(pnorm(z, 10, 16), compute.p = TRUE, d.n = 10)
t
plot(t)

# H0 is false
z <- rbeta(80,4,2)
(t <- ddst.uniform.test(z, compute.p = TRUE, d.n = 10))
t$p.value
plot(t)

## End(Not run)
```

`ddst.upwardtrend.test` *Data Driven Smooth Test for Upward Trend Alternatives*

Description

Performs data driven smooth test for upward trend in k-sample problem. Suppose that we have random samples from k distributions F_i where i = 1, ..., k. The null hypothesis is that there is lack of trend, i.e. F_1 >= ... >= F_k and F_i != F_j for some i and j. The alternative is that there is a trend i.e. F_1 >= ... >= F_k and F_i != F_j for some i and j. This test is implemented as a special case of an umbrella test.

Usage

```
ddst.upwardtrend.test(
  x,
  r.N = rep(4, length(x) - 1),
  alpha = 0.05,
  t.p,
  t.n,
  nr = 1e+05,
  compute.cv = TRUE
)
```

Arguments

x	a list of k (non-empty) numeric vectors of data
r.N	a (k-1)-dimensional vector specifying the levels of complexity of the grids considered, only for advanced users
alpha	a significance level
t.p	an alpha-dependent (k-1)-dimensional vector of the tunning parameters in the penalties in the model selection rules T.o
t.n	an alpha-dependent (k-1)-dimensional vector of the tunning parameters in the penalties in the model selection rules T.tilde
nr	an integer specifying the number of runs for a p-value and a critical value computation if any
compute.cv	a logical value indicating whether to compute a critical value corresponding to the significance level alpha or not
t	an alpha-dependent tunning parameter in the penalty in the model selection rule

References

An automatic test for the umbrella alternatives. Wylupek (2016) https://onlinelibrary.wiley.com/doi/abs/10.1111/sjos.12231

Examples

```
set.seed(7)
# H0 is true
x = runif(80)
y = runif(80) + 0.2
z = runif(80) + 0.4
```

```
t <- ddst.upwardtrend.test(list(x, y, z), t.p = 2.2, t.n = 2.2)
t
plot(t)

# H0 is false
# known fixed alternative
x1 = rnorm(80)
x2 = rnorm(80) + 2
x3 = rnorm(80) + 4
x4 = rnorm(80) + 3
t <- ddst.upwardtrend.test(list(x1, x2, x3, x4), t.p = 2.2, t.n = 2.2)
t
plot(t)
```

`plot.ddst.test` *Plot Function fo Data Driven Tests*

Description

Plots coordinates for selected test statistics

Usage

```
## S3 method for class 'ddst.test'
plot(x, ...)
```

Arguments

`x`	result from ddst test function
`...`	currently not used

Examples

```
# H0 is true
x <- runif(80)
y <- runif(80)
z <- runif(80)
t <- ddst.ksample.test(list(x, y, z))
plot(t)
```

Index

∗ **htest**
ddst.againststochdom.test, 2
ddst.evd.test, 3
ddst.exp.test, 5
ddst.forstochdom.test, 7
ddst.ksample.test, 8
ddst.normbounded.test, 10
ddst.normubounded.test, 11
ddst.twosample.test, 13
ddst.umbrellaknownp.test, 15
ddst.umbrellaunknownp.test, 16
ddst.uniform.test, 18
ddst.upwardtrend.test, 19

ddst.againststochdom.test, 2
ddst.evd.test, 3
ddst.exp.Nk *(*ddst.exp.test*)*, 5
ddst.exp.test, 5
ddst.extr.Nk *(*ddst.evd.test*)*, 3
ddst.forstochdom.test, 7
ddst.ksample.test, 8
ddst.norm.Nk *(*ddst.normbounded.test*)*, 10
ddst.normbounded.test, 10
ddst.normubounded.test, 11
ddst.normunbounded.bias
*(*ddst.normubounded.test*)*, 11
ddst.twosample.test, 13
ddst.umbrellaknownp.test, 15
ddst.umbrellaunknownp.test, 16
ddst.uniform.Nk *(*ddst.uniform.test*)*, 18
ddst.uniform.test, 18
ddst.upwardtrend.test, 19

plot.ddst.test, 21