

Package ‘vsePackage’

June 25, 2009

Version 1.0-1

Date 2008-11-20

Title Companion to the basic statistics course at FM VSE

Author Arnost Komarek <arnost.komarek@mff.cuni.cz>

Maintainer Arnost Komarek <arnost.komarek@mff.cuni.cz>

Depends R (>= 2.0.0)

Description Later

License GPL version 2 or newer

R topics documented:

asypm.mean.test	2
bhapkar.test	4
cars04	5
cube	6
estim.mean	8
estim.var	10
levene.var.test	11
lmbeta.test	13
lr.gof.test	15
lr.indep.test	17
masmooth.coef	18
num2string	20
odds.ratio.test	20
onesample.var.test	23
pearson.indep.test	25
pearson.test	27
plot.tsfit	29
prop.diff.test	31
prop.Z.test	34
rdiscrete	36

rel.risk.test	37
resplot	40
rsemistudent	42
stuart.test	44
symmetry.test	45
ts.explore	47
ts.fit.trends	48
ts.masmooth	51
ts.simpleseason.trend	53
ts.trend	58
tsAccident	60
tsAirpass	61
tsCPI	62
tsGNPdef	63
tukey.additivity.test	63
white.noise.test	65

Index **68**

asypm.mean.test *One-sample asymptotical test on the mean*

Description

Performs one-sample asymptotical test on the mean.

Usage

```
asypm.mean.test(x, mu=0, alternative=c("two.sided", "less", "greater"),
  conf.level=0.95)

## S3 method for class 'asypm.mean.test':
print(x, ...)
```

Arguments

x	a numeric vector, matrix or data frame of data values. If matrix or data frame is supplied, the test is applied separately on each column.
mu	a number indicating the true value μ_0 of the mean under the null hypothesis
alternative	a character string specifying the alternative hypothesis, must be one of “two.sided” (default), “greater” or “less”. You can specify just the initial letter.
conf.level	confidence level of the confidence interval for μ which is also constructed.
...	further arguments to be passed to or from methods.

Details

For a random sample X_1, \dots, X_n from the distribution of X with $\mu = E(X)$, it tests the null hypothesis

$$H_0 : \mu = \mu_0$$

against a two- or one-sided alternative using an asymptotic normality of the studentized sample mean

$$Z = \frac{\sqrt{n}(\bar{X} - \mu_0)}{S_x},$$

where \bar{X} denotes a sample mean and S_x a sample standard deviation. The function also provides the corresponding confidence interval for μ .

See Komárková, Komárek a Bína (2007, Sec. 8.10) for details.

Value

A list with class `asypm.mean.test` containing the following components:

<code>mean</code>	sample mean of the data.
<code>sd</code>	sample standard deviation of the data.
<code>lower</code>	lower limit of the confidence interval for μ .
<code>upper</code>	upper limit of the confidence interval for μ .
<code>mu</code>	the true mean μ_0 under the null hypothesis.
<code>p</code>	the P-value for the test.
<code>Z</code>	the value of the test statistic.
<code>sample.size</code>	sample size n .
<code>nx</code>	number of columns in the input data.
<code>names</code>	column names of the input data.
<code>conf.level</code>	confidence level of the confidence interval for μ .
<code>data.name</code>	name of the input data.
<code>type</code>	type of the alternative hypothesis.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárková, L., Komárek, A. a Bína, V. (2007).
Základy analýzy dat a statistického úsudku s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[estim.mean, t.test](#).

Examples

```
x <- rnorm(100, mean=8, sd=2)
asypm.mean.test(x, mu=7.5)
asypm.mean.test(x, mu=7.5, alter="greater")
asypm.mean.test(x, mu=7.5, alter="less")

y <- rnorm(100, mean=7, sd=1)
z <- data.frame(colOne=x, colTwo=y)
asypm.mean.test(z, mu=7.5)
```

<code>bhapkar.test</code>	<i>Bhapkar chi-squared test of marginal homogeneity in a squared table</i>
---------------------------	--

Description

Performs Bhapkar χ^2 test of marginal homogeneity in a squared table.

Usage

```
bhapkar.test(x)

## S3 method for class 'bhapkar.test':
print(x, ...)
```

Arguments

<code>x</code>	table of observed frequencies (in the matrix form).
<code>...</code>	further arguments to be passed to or from methods.

Details

See Bhapkar (1966) or Komárek a Komárková (2007, Sec. 7.5).

Value

A list with class `bhapkar.test` containing the following components:

<code>w</code>	the value of the test statistic.
<code>df</code>	degrees of freedom of the χ^2 distribution of the test statistic.
<code>p.val</code>	the P-value for the test.
<code>p.hat</code>	estimated marginal category probabilities.
<code>category.names</code>	names of the response categories.
<code>data.name</code>	name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Bhapkar, V. P. (1966). A note on the equivalence of two test criteria for hypotheses in categorical data. *Journal of the American Statistical Association*, **61**, 228–235.

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislosti s příklady v R.
Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[stuart.test](#), [mcnemar.test](#).

Examples

```
## Yule and Kendall (1950). Father's and son's eye color:
eye.color <- matrix(c(194, 83, 25, 56,
                    70, 124, 34, 36,
                    41, 41, 55, 43,
                    30, 36, 23, 109), ncol=4)
rownames(eye.color) <- colnames(eye.color) <- c("blue",
        "blue-green", "darkgrey", "darkbrown")
print(eye.color)
bhapkar.test(eye.color)

## Agresti (2002), Table 10.6
migration <- matrix(c(11607, 87, 172, 63,
                    100, 13677, 255, 176,
                    366, 515, 17819, 286,
                    124, 302, 270, 10192), ncol=4)
rownames(migration) <- colnames(migration) <- c("Northeast",
        "Midwest", "South", "West")
print(migration)
bhapkar.test(migration)
```

cars04

2004 new car and truck data

Description

Specifications are given for 428 new vehicles for the 2004 year. The variables recorded include price, measurements relating to the size of the vehicle, and fuel efficiency.

Usage

```
data(cars04)
```

Format

A data frame with 428 observations on the following 14 variables.

name name of the car.

type type of the car, factor with the following levels: *minivan*, *person* (personal), *pickup*, *sport*, *SUV* (sport utility vehicle), *wagon*.

drive factor specifying the drive of the car. It has the following levels: *all-wheel*, *front*, *rear*.

priceRetail suggested retail price, what the manufacturer thinks the vehicle is worth, including adequate profit for the automaker and the dealer (U.S. Dollars).

priceDealer dealer cost (or “invoice price”), what the dealership pays the manufacturer (U.S. Dollars).

engineSize engine size (liters).

nCylinders number of cylinders (value of -1 stands for rotary engine).

horsePower horse power.

lkmCity city fuel consumption (liters per 100 km).

lkmHighway highway fuel consumption (liters per 100 km).

weight weight (kg).

wheelBase wheel base (cm).

length length (cm).

width width (cm).

Source

Kiplinger's Personal Finance, December 2003, vol. **57**, no. 12, pp. 104-123.

<http://www.kiplinger.com>.

Journal of Statistical Education data archive.

http://www.amstat.org/publications/jse/jse_data_archive.html
04cars.dat

Examples

```
data(cars04)
summary(cars04)
```

cube

Cube plots

Description

Functions to draw cube plots to represent results of a 2^3 factorial experiment.

Usage

```
cube(lty=1, lwd=1, xaxt="n", yaxt="n", xlab="", ylab="", ...)
```

```
cubeY(x=c(1:4, -(1:4)), xname=c("A", "B", "C"), cex=2,
      lty=1, lwd=1, ...)
```

```
cubeA(x=c(1, 2, -1, -2), xname="A", cex=2, length=0.1,
      lty=c(1, 1), lwd=c(2, 1), col=c("red", "black"),
      xaxt="n", yaxt="n", xlab="", ylab="", ...)
```

```
cubeB(x=c(1, 2, -1, -2), xname="B", cex=2, length=0.1,
      lty=c(1, 1), lwd=c(2, 1), col=c("red", "black"),
      xaxt="n", yaxt="n", xlab="", ylab="", ...)
```

```
cubeC(x=c(1, 2, -1, -2), xname="C", cex=2, length=0.1,
      lty=c(1, 1), lwd=c(2, 1), col=c("red", "black"),
      xaxt="n", yaxt="n", xlab="", ylab="", ...)
```

Arguments

x	a vector representing the results of the factorial experiment. For function <code>cubeY</code> , it must have the length of 8, for functions <code>cubeA</code> , <code>cubeB</code> , <code>cubeC</code> , it must have the length of 4.
xname	names of the experimental effects.
length	argument passed to the function <code>arrows</code> .
lty	graphical argument.
lwd	graphical argument.
xaxt	graphical argument.
yaxt	graphical argument.
xlab	graphical argument.
ylab	graphical argument.
cex	graphical argument.
col	graphical argument.
...	further arguments to be passed to or from methods.

Details

See Box, Hunter and Hunter (2005, Chap. 5) for more details.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Box, G. E. P., Hunter, J. S. and Hunter, W. G. (2005). *Statistics for Experimenters: Design, Innovations, and Discovery, Second Edition*. New York: John Wiley and Sons.

Examples

```
cube(lwd=2)

oldPar <- par(mfrow=c(2, 2), bty="n", mar=c(1, 1, 1, 0)+0.1)
cubeY()
cubeA()
cubeB()
cubeC()
par(oldPar)
```

 estim.mean

Mean estimation

Description

Computes point estimate and two confidence intervals for the mean.

Usage

```
estim.mean(x, type=c("two.sided", "less", "greater"), conf.level=0.95)

## S3 method for class 'estim.mean':
print(x, ...)
```

Arguments

<code>x</code>	a numeric vector, matrix or data frame of data values. If matrix or data frame is supplied, the mean estimate is produced separately for each column.
<code>type</code>	a character string specifying the type of the confidence interval, must be one of “two.sided” (default), “greater” (left-sided interval) or “less” (right-sided interval). You can specify just the initial letter.
<code>conf.level</code>	confidence level of the confidence interval.
<code>...</code>	further arguments to be passed to or from methods.

Details

For a random sample X_1, \dots, X_n from the distribution of X with $\mu = E(X)$, it produces a point and two interval estimates of the mean μ .

The point estimate is equal to the sample mean

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i.$$

Further, let S_x be the sample standard deviation. The confidence intervals are based on the statistic

$$Z = \frac{\sqrt{n}(\bar{X} - \mu)}{S_x}$$

assuming either large sample size n and (asymptotic) normality of Z or assuming normality of X and a Student t_{n-1} distribution of Z .

See Komárková, Komárek a Bína (2007, Sec. 7.7) for details.

Value

A list with class `estim.mean` containing the following components:

<code>mean</code>	sample mean of the data.
<code>sd</code>	sample standard deviation of the data.
<code>lower</code>	lower limit of the confidence interval for μ based on the asymptotical normality of the Z statistic.
<code>upper</code>	upper limit of the confidence interval for μ based on the asymptotical normality of the Z statistic.
<code>lower.t</code>	lower limit of the confidence interval for μ based on the assumption of normality of X and Student t distribution of the Z statistic.
<code>upper.t</code>	upper limit of the confidence interval for μ based on the assumption of normality of X and Student t distribution of the Z statistic.
<code>sample.size</code>	sample size n .
<code>nx</code>	number of columns in the input data.
<code>names</code>	column names of the input data.
<code>conf.level</code>	confidence level of the confidence interval for μ .
<code>data.name</code>	name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárková, L., Komárek, A. a Bína, V. (2007).
Základy analýzy dat a statistického úsudku s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[asymp.mean.test](#), [t.test](#).

Examples

```
x <- rnorm(100, mean=8, sd=2)
estim.mean(x)
estim.mean(x, type="greater")
estim.mean(x, type="less")

y <- rnorm(100, mean=7, sd=1)
z <- data.frame(colOne=x, colTwo=y)
estim.mean(z)
```

 estim.var

Variance estimation

Description

Computes point estimates and confidence intervals for variance and standard deviation of a normal sample.

Usage

```
estim.var(x, type=c("two.sided", "less", "greater"), conf.level=0.95)

## S3 method for class 'estim.var':
print(x, ...)
```

Arguments

x	a numeric vector, matrix or data frame of data values. If matrix or data frame is supplied, the variance and standard deviation estimate is produced separately for each column.
type	a character string specifying the type of the confidence interval, must be one of “two.sided” (default), “greater” (left-sided interval) or “less” (right-sided interval). You can specify just the initial letter.
conf.level	confidence level of the confidence interval.
...	further arguments to be passed to or from methods.

Details

For a random sample X_1, \dots, X_n from the distribution of $X \sim N(\mu, \sigma^2)$, it produces a point and interval estimate of the variance σ^2 and standard deviation σ .

Let \bar{X} be the sample mean. The point estimate of the variance is equal to the sample variance

$$S_x^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2.$$

The confidence interval for the variance is based on the χ_{n-1}^2 distribution of the statistic

$$\frac{(n-1)S_x^2}{\sigma^2}.$$

See Komárková, Komárek a Bína (2007, Sec. 7.8) for details.

Value

A list with class `estim.var` containing the following components:

<code>var</code>	sample variance of the data.
<code>lower</code>	lower limit of the confidence interval for σ^2 .
<code>upper</code>	upper limit of the confidence interval for σ^2 .
<code>sample.size</code>	sample size n .
<code>nx</code>	number of columns in the input data.
<code>names</code>	column names of the input data.
<code>conf.level</code>	confidence level of the confidence interval for σ^2 .
<code>data.name</code>	name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárková, L., Komárek, A. a Bína, V. (2007).
Základy analýzy dat a statistického úsudku s příklady v R.
Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[onesample.var.test](#).

Examples

```
x <- rnorm(100, mean=8, sd=2)
estim.var(x)
estim.var(x, type="greater")
estim.var(x, type="less")

y <- rnorm(100, mean=7, sd=1)
z <- data.frame(colOne=x, colTwo=y)
estim.var(z)
```

`levene.var.test` *Modified Levene test of homoscedasticity*

Description

Performs modified Levene test of homoscedasticity, i.e. the test of the null hypothesis that the variances in each of the groups (samples) are the same.

Usage

```
levene.var.test(x, ...)

## Default S3 method:
levene.var.test(x, g, ...)

## S3 method for class 'formula':
levene.var.test(formula, data, subset, na.action, ...)
```

Arguments

<code>x</code>	a (non-empty) numeric vector of data values.
<code>g</code>	a factor variable of the same length as <code>x</code> which classifies <code>x</code> in the groups.
<code>formula</code>	a formula of the form <code>lhs ~ rhs</code> where <code>lhs</code> is a numeric variable giving the data values and <code>rhs</code> a factor with two or more levels giving the corresponding groups.
<code>data</code>	an optional matrix or data frame (or similar, see <code>model.frame</code>) containing the variables in the formula <code>formula</code> . By default, the variables are taken from <code>environment(formula)</code> .
<code>subset</code>	an optional vector specifying a subset of observations to be used.
<code>na.action</code>	a function which indicates what should happen when the data contain NAs. Defaults to <code>getOption("na.action")</code> .
<code>...</code>	further arguments to be passed to or from methods.

Details

See Neter et al. (1996, p. 766) or Komárek a Komárková (2007, Sec. 8.4) for details.

Value

A list with class `htest` containing the following components:

<code>statistic</code>	the value of the test statistic.
<code>parameter</code>	numerator and denominator degrees of freedom for the F-distribution of the test statistic under the null hypothesis.
<code>p.value</code>	the P-value for the test.
<code>data.name</code>	a character specifying the name(s) of the data.
<code>method</code>	a character string indicating what type of the test was performed.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

- Komárek, A. a Komárková, L. (2007).
Statistická analýza závislosti s příklady v R.
Jindřichův Hradec: Fakulta managementu VŠE.
- Neter, J., Kutner, M. H., Nachtsheim, C. J., and Wasserman, W. (1996). *Applied Linear Statistical Models, 4th Edition*. Boston: WCB/McGraw-Hill.

See Also

`bartlett.test`.

Examples

```
### Example from Neter, Kutner, Nachtsheim, Wasserman (1996), p. 767
yy <- c(14.87, 16.81, 15.83, 15.47, 13.60,
        14.76, 17.40, 14.62, 18.43, 18.76,
        20.12, 19.11, 19.81, 18.43, 17.16,
        16.40, 16.95, 12.28, 12.00, 13.18,
        14.99, 15.76, 19.35, 15.52, 8.59,
        10.90, 8.60, 10.13, 10.28, 9.98,
        9.41, 10.04, 11.55, 13.36, 13.64,
        12.16, 11.62, 12.39, 12.05, 11.95)
solder <- data.frame(yy=yy,
                    joint=gl(8, 1, 40, labels=1:8),
                    flux=gl(5, 8, 40, labels=1:5))

### Test of homoscedasticity of yy w.r.t. flux groups
levene.var.test(yy ~ flux, data=solder)
levene.var.test(solder$yy, solder$flux)
```

`lmbeta.test`

t-tests and confidence intervals for the regression coefficients

Description

Performs t-tests on the regression coefficients and compute confidence intervals for the regression coefficients in the normal linear regression model.

Usage

```
lmbeta.test(object, beta.null=0,
            alternative=c("two.sided", "less", "greater"), conf.level=0.95)
```

Arguments

<code>object</code>	an object with class <code>lm</code> .
<code>beta.null</code>	the value β^* of the regression coefficient under the null hypothesis.
<code>alternative</code>	a character string specifying the alternative hypothesis, must be one of “two.sided” (default), “greater” or “less”. You can specify just the initial letter.
<code>conf.level</code>	confidence level of the confidence interval for β s which is also constructed.

Details

In the linear model

$$Y_i = \beta_0 + \beta_1 x_{i,1} + \dots + \beta_p x_{i,p} + \varepsilon_i, \quad \varepsilon_i \sim \mathbf{N}(0, \sigma^2) \quad (i = 1, \dots, n)$$

tests (**separately for each k !!!**) the null hypotheses

$$H_{0,k} : \beta_k = \beta^*$$

against two- or one-sided alternatives using the t-tests. For each regression coefficient a corresponding two- or one-sided confidence interval is computed as well.

See Neter et al. (1996, Sec. 6.6) or Komárek a Komárková (2007, Sec. 10.8) for details.

Value

A data frame with the following columns:

<code>Estimate</code>	point estimates (least squares) of the β coefficients.
<code>Std. Error</code>	standard errors of estimated regression coefficients.
<code>Conf.</code>	confidence level of constructed confidence intervals.
<code>Alternative</code>	a character string specifying the type of the alternative hypothesis.
<code>Estim. Low</code>	lower limit of the confidence interval for each β coefficient.
<code>Estim. Up</code>	upper limit of the confidence interval for each β coefficient.
<code>Beta H0</code>	the value of the regression coefficient under the null hypothesis.
<code>t value</code>	the value of the test t-statistic.
<code>p value</code>	the P-value for the test.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

- Komárek, A. a Komárková, L. (2007). *Statistická analýza závislostí s příklady v R*. Jindřichův Hradec: Fakulta managementu VŠE.
- Neter, J., Kutner, M. H., Nachtsheim, C. J., and Wasserman, W. (1996). *Applied Linear Statistical Models, 4th Edition*. Boston: WCB/McGraw-Hill.

See Also

[summary.lm.](#)

Examples

```
data(cars04)
model <- lm(lkmCity~I(log(weight)), data=cars04)

### H0: beta1=11 against beta1 > 11
### * only the row labeled I(log(weight)) is relevant here!
lmbeta.test(model, beta.null=11, alternative="greater")
```

 lr.gof.test

Likelihood ratio goodness-of-fit test

Description

Performs likelihood ratio goodness-of-fit test for a multinomial distribution.

Usage

```
lr.gof.test(x, p)

## S3 method for class 'lr.gof.test':
print(x, ...)
```

Arguments

x vector of observed counts N_0, \dots, N_I .
p vector of category probabilities π_0^*, \dots, π_I^* under the null hypothesis.
... further arguments to be passed to or from methods.

Details

For a vector of counts

$$N = (N_0, \dots, N_I)'$$

sampled from a multinomial distribution $\text{Multin}_{I+1}(n; \pi_0, \dots, \pi_I)$, it tests the null hypothesis

$$H_0 : \pi_0 = \pi_0^*, \dots, \pi_I = \pi_I^*$$

using the likelihood ratio test.

Let $N = \sum_{i=0}^I N_i$ and let

$$\mu_i = N \pi_i^* \quad (i = 0, \dots, I)$$

be the expected counts under the null hypothesis. The likelihood ratio test statistic is given by

$$G^2 = 2 \sum_{i=0}^I N_i \log \left(\frac{N_i}{\mu_i} \right)$$

and follows under the null hypothesis χ_I^2 distribution.

Residuals for the residual analysis are given by

$$e_i = \text{sign}_i \sqrt{\left| 2 N_i \log \left(\frac{N_i}{\mu_i} \right) \right|} \quad (i = 0, \dots, I),$$

where sign_i is equal to 1 if $N_i > \mu_i$ and is equal to -1 otherwise.

See Komárek a Komárková (2007, Sec. 5.2) for details.

Value

A list with class `lr.gof.test` containing the following components:

<code>G2</code>	the value of the test statistic.
<code>df</code>	degrees of freedom of the null hypothesis χ^2 distribution of the test statistic.
<code>p.val</code>	the P-value for the test.
<code>G2.components</code>	components of the test statistic.
<code>tab</code>	data frame with columns
	<code>p</code> category probabilities under the null hypothesis
	<code>p.hat</code> estimated (empirical) category probabilities
	<code>observed</code> observed counts
	<code>expected</code> expected counts under the null hypothesis
	<code>residuals</code> vector of residuals
<code>data.name</code>	name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislostí s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[pearson.test](#).

Examples

```
## observed counts:
NN <- c(51, 150, 190, 251, 93)
names(NN) <- c("++", "+", "0", "-", "--")
```

```
## hypothetical probabilities:
pp0 <- c(0.05, 0.20, 0.25, 0.35, 0.15)

## test:
lr.gof.test(x=NN, p=pp0)
```

```
lr.indep.test          Likelihood ratio test of independence
```

Description

Performs likelihood ratio test of independence in the contingency table.

Usage

```
lr.indep.test(x)

## S3 method for class 'lr.indep.test':
print(x, ...)
```

Arguments

`x` a matrix of observed counts (contingency table).
`...` further arguments to be passed to or from methods.

Details

For a $(I + 1) \times (J + 1)$ contingency table $N = \{N_{i,j} : i = 0, \dots, I, j = 0, \dots, J\}$, it tests the null hypothesis of independence of underlying categorical responses.

Let $\mu_{i,j}$ ($i = 0, \dots, I, j = 0, \dots, J$) be the expected counts under the null hypothesis. The likelihood ratio test statistic is given by

$$G^2 = 2 \sum_{i=0}^I \sum_{j=0}^J N_{i,j} \log \left(\frac{N_{i,j}}{\mu_{i,j}} \right)$$

and follows under the null hypothesis $\chi_{I \cdot J}^2$ distribution.

See Komárek a Komárková (2007, Sec. 6.2) for details.

Value

A list with class `lr.indep.test` containing the following components:

`G2` the value of the test statistic.
`df` degrees of freedom of the null hypothesis χ^2 distribution of the test statistic.
`p.val` the P-value for the test.
`G2.components` components of the test statistic.

observed a matrix of observed counts.
expected a matrix of expected counts under the null hypothesis.
data.name name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislostí s příklady v R.
Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[pearson.indep.test](#), [chisq.test](#).

Examples

```
### Opinion on the influence of movies to children
### (General Social Survey 1990)
inffilms <- matrix(c(51, 42, 150, 167, 190, 164, 251, 165, 93, 49), nrow=2)
rownames(inffilms) <- c("Female", "Male")
colnames(inffilms) <- c("++", "+", "0", "-", "--")
print(inffilms)

### Test of independence
lr.indep.test(inffilms)
```

masmooth.coef

Coefficients for moving averages smoothing

Description

Computes coefficients of the moving average.

Usage

```
masmooth.coef(order=3, length=5)
```

Arguments

order order of the moving average.
length length of the moving average.

Details

For odd `length` of $2M + 1$, the function computes coefficients of **simple** moving average of given `order`. For even `length` of $2M$, the function computes coefficients of **centered** moving average of length $2M + 1$, `order` is ignored in this case.

See Komárek a Komárková (2007, Sec. 13.2) for details.

Value

A list with the following components:

<code>coef</code>	a vector of coefficients to compute moving average inside.
<code>H</code>	hat matrix which can be used to compute simple moving average on the boundary.
<code>order</code>	order of the moving average.
<code>length</code>	length of the moving average.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislosti s příklady v R.
Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[ts.masmooth](#).

Examples

```
masmooth.coef(order=0, length=7)
masmooth.coef(order=1, length=7)
masmooth.coef(order=2, length=7)

masmooth.coef(order=0, length=12)
masmooth.coef(order=1, length=12)
masmooth.coef(order=2, length=12)
```

num2string *Conversion of a numeric value to LaTeX string*

Description

It converts a numeric value to the string which is formatted for the use in LaTeX.

Usage

```
num2string(x, ndec=2, add.dollar=TRUE, sep=".")
```

Arguments

x	numeric vector.
ndec	number of decimal places to round x.
add.dollar	logical, which indicates whether possible negative sign should be placed between the dollars (mathematics in LaTeX).
sep	character to be used to separate decimal places in the output.

Value

Character value or vector.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

Examples

```
num2string(c(3.67832, -0.96522, -0.7342))
num2string(c(3.67832, -0.96522, -0.7342), ndec=3, add.dollar=FALSE)
num2string(c(3.67832, -0.96522, -0.7342), ndec=3, add.dollar=TRUE, sep=",")
```

odds.ratio.test *Two-sample test and confidence interval for the odds ratio*

Description

Performs two-sample test on the odds ratio and computes the corresponding confidence interval for the odds ratio.

Usage

```
odds.ratio.test(x, n, or=1,
               alternative=c("two.sided", "less", "greater"), conf.level=0.95)

## S3 method for class 'odds.ratio.test':
print(x, ...)
```

Arguments

x a two-component vector of counts of successes in the two groups, i.e. the vector $(N_{0,1}, N_{1,1})'$ or a 2 by 2 matrix with the following entries

$$\begin{array}{ccc|ccc} & & & n_0 - N_{0,1} & N_{0,1} & \\ & & & n_1 - N_{1,1} & N_{1,1} & \end{array}$$

n a two-component vector of counts of trials in the two groups, i.e. the vector $(n_0, n_1)'$.
It is ignored if **x** is a matrix.

or the odds ratio under the null hypothesis (θ^*).

alternative a character string specifying the alternative hypothesis, must be one of “two.sided” (default), “greater” or “less”. You can specify just the initial letter.

conf.level confidence level of the confidence interval for the odds ratio θ which is also constructed.

... further arguments to be passed to or from methods.

Details

For counts $N_{0,1}$ and $N_{1,1}$ sampled independently from binomial distributions $\text{Binom}(n_0; \pi_0)$ and $\text{Binom}(n_1; \pi_1)$, respectively, the odds ratio is defined as

$$\theta = \frac{\frac{\pi_1}{1 - \pi_1}}{\frac{\pi_0}{1 - \pi_0}}.$$

This function tests the null hypothesis

$$H_0 : \theta = \theta^*$$

using the normal approximation to the corresponding Z statistic, i.e.

$$Z = \frac{\log(\hat{\theta}) - \log(\theta^*)}{\widehat{\text{se}}\{\log(\hat{\theta})\}},$$

where $\hat{\theta}$ is the observed odds ratio. That is,

$$\hat{\theta} = \frac{\frac{p_1}{1 - p_1}}{\frac{p_0}{1 - p_0}}, \quad p_1 = \frac{N_{1,1}}{n_1}, \quad p_0 = \frac{N_{0,1}}{n_0}.$$

The function also produces the confidence interval for the odds ratio θ based on the above Z statistic. See Komárek a Komárková (2007, Sec. 4.4) for details.

Value

A list with class `odds.ratio.test` containing the following components:

<code>Parameter</code>	estimate of the odds ratio θ .
<code>Log.parameter</code>	estimate of the log-odds ratio $\log(\theta)$.
<code>SE.log.parameter</code>	estimated standard error of $\log(\hat{\theta})$.
<code>lower</code>	lower limit of the confidence interval for θ based on the inversion of the Z statistic.
<code>upper</code>	upper limit of the confidence interval for θ based on the inversion of the Z statistic.
<code>log.lower</code>	lower limit of the confidence interval for $\log(\theta)$ based on the inversion of the Z statistic.
<code>log.upper</code>	upper limit of the confidence interval for $\log(\theta)$ based on the inversion of the Z statistic.
<code>p.success</code>	estimated probabilities of success in the two groups.
<code>n</code>	the counts of trials in the two groups.
<code>n.success</code>	the counts of successes in the two groups.
<code>Z</code>	the value of the test statistic.
<code>p.value</code>	the P-value for the test.
<code>or</code>	the odds ratio under the null hypothesis (θ^*).
<code>typ</code>	a number specifying the alternative hypothesis.
<code>alternative</code>	a character string specifying the alternative hypothesis.
<code>conf.level</code>	confidence level of the confidence interval for θ .
<code>data.name</code>	name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislostí s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[prop.test](#), [prop.diff.test](#), [rel.risk.test](#).

Examples

```

### Opinion over own alcoholism
### (General Social Survey 1993)
### Do you sometimes drink too much alcohol?
### Answers further cross-classified w.r.t. gender.
alco <- matrix(c(303, 188, 106, 134), nrow=2, ncol=2)
rownames(alco) <- c("female", "male")
colnames(alco) <- c("no", "yes")
print(alco)

### pi0 = P(drink too much | female)
### pi1 = P(drink too much | male)
### H0: (pi1/(1-pi1))/(pi0/(1-pi0)) = 1 against
### H1: (pi1/(1-pi1))/(pi0/(1-pi0)) <> 1
odds.ratio.test(alco)

## The same
n.yes <- alco[, "yes"]
n.total <- apply(alco, 1, sum)
odds.ratio.test(x=n.yes, n=n.total)

## Test of
### H0: (pi1/(1-pi1))/(pi0/(1-pi0)) = 1.5 against
### H1: (pi1/(1-pi1))/(pi0/(1-pi0)) <> 1.5
odds.ratio.test(alco, or=1.5, alternative="greater")

```

onesample.var.test *One-sample test on the variance*

Description

Performs one-sample test on the variance and standard deviation of normally distributed sample.

Usage

```

onesample.var.test(x, sd=1, var,
  alternative=c("two.sided", "less", "greater"), conf.level=0.95)

## S3 method for class 'onesample.var.test':
print(x, ...)

```

Arguments

x	a numeric vector, matrix or data frame of data values. If matrix or data frame is supplied, the mean estimate is produced separately for each column.
sd	the true standard deviation σ_0 under the null hypothesis.
var	the true variance σ_0^2 under the null hypothesis. It is sufficient to supply only either sd or var.

alternative	a character string specifying the alternative hypothesis, must be one of “two.sided” (default), “greater” or “less”. You can specify just the initial letter.
conf.level	confidence level of the confidence interval for σ^2 which is also constructed.
...	further arguments to be passed to or from methods.

Details

For a random sample X_1, \dots, X_n from the distribution of $X \sim N(\mu, \sigma^2)$, it tests the null hypothesis

$$H_0 : \sigma^2 = \sigma_0^2$$

against a two- or one-sided alternative using a χ_{n-1}^2 distribution of the statistic

$$SS = \frac{(n-1)S_x^2}{\sigma_0^2},$$

where S_x^2 is the sample variance. The function also provides the corresponding confidence interval for σ^2 and σ .

See Komárková, Komárek a Bína (2007, Sec. 8.13) for details.

Value

A list with class `estim.var` containing the following components:

var	sample variance of the data.
lower	lower limit of the confidence interval for σ^2 .
upper	upper limit of the confidence interval for σ^2 .
hyp.var	the true variance σ^2 under the null hypothesis.
p	the P-value of the test.
SS	the value of the test statistic.
sample.size	sample size n .
nx	number of columns in the input data.
names	column names of the input data.
conf.level	confidence level of the confidence interval for σ^2 .
data.name	name of the input data.
type	type of the alternative hypothesis.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárková, L., Komárek, A. a Bína, V. (2007). *Základy analýzy dat a statistického úsudku s příklady v R*. Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[onesample.var.test.](#)

Examples

```
x <- rnorm(100, mean=8, sd=2)
onesample.var.test(x, sd=1.8)
onesample.var.test(x, var=1.8^2)
onesample.var.test(x, sd=1.8, alter="greater")
onesample.var.test(x, sd=1.8, alter="less")

y <- rnorm(100, mean=7, sd=1)
z <- data.frame(colOne=x, colTwo=y)
onesample.var.test(z, sd=1)
```

pearson.indep.test *Pearson chi-squared test of independence*

Description

Performs Pearson chi-squared test of independence in the contingency table.

Usage

```
pearson.indep.test(x)

## S3 method for class 'pearson.indep.test':
print(x, ...)
```

Arguments

`x` a matrix of observed counts (contingency table).
`...` further arguments to be passed to or from methods.

Details

For a $(I + 1) \times (J + 1)$ contingency table $N = \{N_{i,j} : i = 0, \dots, I, j = 0, \dots, J\}$, it tests the null hypothesis of independence of underlying categorical responses.

Let $\mu_{i,j}$ ($i = 0, \dots, I, j = 0, \dots, J$) be the expected counts under the null hypothesis. The Pearson's test statistic is given by

$$X^2 = \sum_{i=0}^I \sum_{j=0}^J \frac{(N_{i,j} - \mu_{i,j})^2}{\mu_{i,j}}$$

and follows under the null hypothesis $\chi_{I \cdot J}^2$ distribution.

Pearson's residuals for the residual analysis are given by

$$e_{i,j} = \frac{N_{i,j} - \mu_{i,j}}{\sqrt{\mu_{i,j}}} \quad (i = 0, \dots, I, j = 0, \dots, J).$$

Adjusted Pearson's residuals are given by

$$e_{i,j} = \frac{N_{i,j} - \mu_{i,j}}{\sqrt{\mu_{i,j}(1 - p_{i+})(1 - p_{+j})}} \quad (i = 0, \dots, I, j = 0, \dots, J),$$

where p_{i+} ($i = 0, \dots, I$) are empirical (observed) row probabilities and p_{+j} ($j = 0, \dots, J$) are empirical (observed) column probabilities.

See Komárek a Komárková (2007, Sec. 6.2) for details.

Value

A list with class `pearson.indep.test` containing the following components:

<code>X2</code>	the value of the test statistic.
<code>df</code>	degrees of freedom of the null hypothesis χ^2 distribution of the test statistic.
<code>p.val</code>	the P-value for the test.
<code>X2.components</code>	components of the test statistic.
<code>observed</code>	a matrix of observed counts.
<code>expected</code>	a matrix of expected counts under the null hypothesis.
<code>residuals</code>	a matrix of Pearson's residuals.
<code>adjust.residuals</code>	a matrix of adjusted Pearson's residuals.
<code>data.name</code>	name of the input data.
<code>C</code>	Pearson's contingency coefficient.
<code>C.max</code>	maximal possible value of the Pearson's contingency coefficient.
<code>V</code>	Cramer's contingency coefficient.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislosti s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[lr.indep.test](#), [chisq.test](#).

Examples

```
### Opinion on the influence of movies to children
### (General Social Survey 1990)
inffilms <- matrix(c(51, 42, 150, 167, 190, 164, 251, 165, 93, 49), nrow=2)
rownames(inffilms) <- c("Female", "Male")
colnames(inffilms) <- c("++", "+", "0", "-", "--")
print(inffilms)

### Test of independence
pearson.indep.test(inffilms)

### The same result:
chisq.test(inffilms, correct=FALSE)
```

pearson.test	<i>Pearson chi-squared goodness-of-fit test</i>
--------------	---

Description

Performs Pearson chi-squared goodness-of-fit test for a multinomial distribution.

Usage

```
pearson.test(x, p)

## S3 method for class 'pearson.test':
print(x, ...)
```

Arguments

`x` vector of observed counts N_0, \dots, N_I .
`p` vector of category probabilities π_0^*, \dots, π_I^* under the null hypothesis.
`...` further arguments to be passed to or from methods.

Details

For a vector of counts

$$N = (N_0, \dots, N_I)'$$

sampled from a multinomial distribution $\text{Multin}_{I+1}(n; \pi_0, \dots, \pi_I)$, it tests the null hypothesis

$$H_0 : \pi_0 = \pi_0^*, \dots, \pi_I = \pi_I^*$$

using the Pearson's χ^2 test.

Let $N = \sum_{i=0}^I N_i$ and let

$$\mu_i = N \pi_i^* \quad (i = 0, \dots, I)$$

be the expected counts under the null hypothesis. The Pearson's test statistic is given by

$$X^2 = \sum_{i=0}^I \frac{(N_i - \mu_i)^2}{\mu_i}$$

and follows under the null hypothesis χ_I^2 distribution.

Pearson's residuals for the residual analysis are given by

$$e_i = \frac{N_i - \mu_i}{\sqrt{\mu_i}} \quad (i = 0, \dots, I).$$

See Komárek a Komárková (2007, Sec. 5.1) for details.

Value

A list with class `pearson.test` containing the following components:

<code>X2</code>	the value of the test statistic.
<code>df</code>	degrees of freedom of the null hypothesis χ^2 distribution of the test statistic.
<code>p.val</code>	the P-value for the test.
<code>X2.components</code>	components of the test statistic.
<code>tab</code>	data frame with columns
	<code>p</code> category probabilities under the null hypothesis
	<code>p.hat</code> estimated (empirical) category probabilities
	<code>observed</code> observed counts
	<code>expected</code> expected counts under the null hypothesis
	<code>residuals</code> vector of residuals
<code>data.name</code>	name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislosti s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[lr.gof.test](#).

Examples

```
## observed counts:
NN <- c(51, 150, 190, 251, 93)
names(NN) <- c("++", "+", "0", "-", "--")

## hypothetical probabilities:
pp0 <- c(0.05, 0.20, 0.25, 0.35, 0.15)

## test:
pearson.test(x=NN, p=pp0)
```

plot.tsfit

*Plot fitted time series models***Description**

For fitted time series model, it plots fitted trend, fitted values, predictive values etc.

Usage

```
## S3 method for class 'tsfit':
plot(x, step,
      type.ts="l", lty.ts=4, pch.ts=1, col.ts="black",
      type.fit="l", lty.fit=1, pch.fit=16, col.fit="red",
      type.trend="l", lty.trend=1, pch.trend=16, col.trend="blue",
      type.pred="l", lty.pred=2, pch.pred=pch.fit, col.pred=col.fit,
      add.point.ts=FALSE, add.point.fit=TRUE, add.point.pred=TRUE,
      pch.add=15, bty, xlab, ylab, xlim, ylim, ...)
```

Arguments

x	an object of class <code>tsfit</code> , typically obtained by one of the following functions: ts.trend , ts.masmooth , ts.simpleseason.trend .
step	a numeric vector indicating for how many steps in the future we want to predict and plot the prediction.
type.ts	character specifying the type of the plot to draw the original time series.
lty.ts	value which specifies the type of the line to draw the original time series if <code>type.ts</code> is "l".
pch.ts	value which specifies the plotting character to draw the original time series if <code>type.ts</code> is "p" or if <code>add.point.ts</code> is TRUE.
col.ts	character which specifies the color to draw the original time series.
type.fit	character specifying the type of the plot to draw the fitted trend.
lty.fit	value which specifies the type of the line to draw the fitted time series if <code>type.fit</code> is "l".

<code>pch.fit</code>	value which specifies the plotting character to draw the fitted time series if <code>type.fit</code> is “p” or if <code>add.point.fit</code> is TRUE.
<code>col.fit</code>	character which specifies the color to draw the fitted time series.
<code>type.trend</code>	character specifying the type of the plot to draw the fitted trend.
<code>lty.trend</code>	value which specifies the type of the line to draw the fitted trend if <code>type.trend</code> is “l”.
<code>pch.trend</code>	value which specifies the plotting character to draw the fitted trend if <code>type.trend</code> is “p”.
<code>col.trend</code>	character which specifies the color to draw the fitted trend.
<code>type.pred</code>	character specifying the type of the plot to draw the prediction.
<code>lty.pred</code>	value which specifies the type of the line to draw the prediction if <code>type.pred</code> is “l”.
<code>pch.pred</code>	value which specifies the plotting character to draw the prediction if <code>type.pred</code> is “p” or if <code>add.point.pred</code> is TRUE.
<code>col.pred</code>	character which specifies the color to draw the prediction.
<code>add.point.ts</code>	logical which indicates whether the original time series should (also) be indicated by the points.
<code>add.point.fit</code>	logical which indicates whether the fitted time series should (also) be indicated by the points.
<code>add.point.pred</code>	logical which indicates whether the prediction should (also) be indicated by the points.
<code>pch.add</code>	value indicating the plotting character used if either <code>add.point.ts</code> or <code>add.point.fit</code> is TRUE.
<code>bty</code>	graphical parameter passed to the <code>plot</code> function.
<code>xlab</code>	graphical parameter passed to the <code>plot</code> function.
<code>ylab</code>	graphical parameter passed to the <code>plot</code> function.
<code>xlim</code>	graphical parameter passed to the <code>plot</code> function.
<code>ylim</code>	graphical parameter passed to the <code>plot</code> function.
<code>...</code>	further arguments to be passed to or from methods.

Value

`invisible(x)`.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

See Also

`ts.trend`, `ts.masmooth`, `ts.simpleseason.trend`, `ts`.

Examples

```

### Example 1a
data(tsCPI)
fit <- ts.trend(tsCPI, trend="modified.exponential")
plot(fit, step=1:4, add.point.fit=FALSE, xlab="Year", ylab="CPI")

### Example 1b
fit7 <- ts.msmooth(tsCPI, length=7)
plot(fit7, step=1:4, add.point.fit=FALSE, add.point.pred=FALSE,
     col.pred="darkblue", xlab="Year", ylab="CPI")

### Example 1c
fit7c <- ts.msmooth(tsCPI, length=6)
plot(fit7c, add.point.fit=FALSE, xlab="Year", ylab="CPI")

### Example 2a
data(tsAccident)
fitAc1 <- ts.simpleseason.trend(tsAccident, decomposition="additive")
plot(fitAc1, xlab="Year", ylab="#deaths",
     main="Seasonally cleaned time series")

### Example 2b
fitAc3 <- ts.simpleseason.trend(tsAccident,
                               decomposition="additive", trend="quadratic")
plot(fitAc3, step=1:12, xlab="Year", ylab="#deaths")

### Example 3a
data(tsAirpass)
fitAirp1 <- ts.simpleseason.trend(tsAirpass,
                                  decomposition="multiplicative")
plot(fitAirp1, xlab="Year", ylab="#passangers (x1000)",
     main="Seasonally cleaned time series")

### Example 3b
fitAirp3 <- ts.simpleseason.trend(tsAirpass,
                                  decomposition="multiplicative", trend="gompertz")
plot(fitAirp3, step=1:12, xlab="Year", ylab="#passangers (x1000)")

```

prop.diff.test

Two-sample test and confidence interval for the difference of proportions

Description

Performs two-sample test on the difference of proportions and computes the corresponding confidence interval for the difference of proportions.

Usage

```
prop.diff.test(x, n, diff=0,
              alternative=c("two.sided", "less", "greater"), conf.level=0.95)

## S3 method for class 'prop.diff.test':
print(x, ...)
```

Arguments

x a two-component vector of counts of successes in the two groups, i.e. the vector $(N_{0,1}, N_{1,1})'$ or a 2 by 2 matrix with the following entries

$$\begin{array}{cc|cc} & & n_0 - N_{0,1} & N_{0,1} & | \\ & & n_1 - N_{1,1} & N_{1,1} & | \end{array}$$

n a two-component vector of counts of trials in the two groups, i.e. the vector $(n_0, n_1)'$.
It is ignored if **x** is a matrix.

diff the difference of the probabilities of success under the null hypothesis (θ^*).

alternative a character string specifying the alternative hypothesis, must be one of “two.sided” (default), “greater” or “less”. You can specify just the initial letter.

conf.level confidence level of the confidence interval for the probability of success π_1 which is also constructed.

... further arguments to be passed to or from methods.

Details

For counts $N_{0,1}$ and $N_{1,1}$ sampled independently from binomial distributions $\text{Binom}(n_0; \pi_0)$ and $\text{Binom}(n_1; \pi_1)$, respectively, it tests the null hypothesis

$$H_0 : \pi_1 - \pi_0 = \theta^*$$

using the normal approximation to the corresponding Z statistic, i.e.

$$Z = \frac{p_1 - p_0 - \theta^*}{\widehat{\text{se}}(p_1 - p_0)},$$

where

$$p_1 = \frac{N_{1,1}}{n_1}, \quad p_0 = \frac{N_{0,1}}{n_0}$$

are observed proportions of success.

The function also produces the confidence interval for the difference of the proportions $\theta = \pi_1 - \pi_0$ based on the above Z statistic.

See Komárek a Komárková (2007, Sec. 4.2) for details.

Value

A list with class `prop.diff.test` containing the following components:

<code>Parameter</code>	estimate of the difference in proportions $\theta = \pi_1 - \pi_0$.
<code>SE.parameter</code>	estimated standard error of $\hat{\theta}$.
<code>lower</code>	lower limit of the confidence interval for θ based on the inversion of the Z statistic.
<code>upper</code>	upper limit of the confidence interval for θ based on the inversion of the Z statistic.
<code>p.success</code>	estimated probabilities of success in the two groups.
<code>n</code>	the counts of trials in the two groups.
<code>n.success</code>	the counts of successes in the two groups.
<code>Z</code>	the value of the test statistic.
<code>p.value</code>	the P-value for the test.
<code>diff</code>	the difference in proportions of success under the null hypothesis (θ^*).
<code>typ</code>	a number specifying the alternative hypothesis.
<code>alternative</code>	a character string specifying the alternative hypothesis.
<code>conf.level</code>	confidence level of the confidence interval for π_1 .
<code>data.name</code>	name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislosti s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

`prop.test`, `rel.risk.test`, `odds.ratio.test`.

Examples

```
### Opinion over own alcoholism
### (General Social Survey 1993)
### Do you sometimes drink too much alcohol?
### Answers further cross-classified w.r.t. gender.
alco <- matrix(c(303, 188, 106, 134), nrow=2, ncol=2)
rownames(alco) <- c("female", "male")
colnames(alco) <- c("no", "yes")
print(alco)

### pi0 = P(drink too much | female)
```

```

### pi1 = P(drink too much | male)
### H0: pi1 - pi0 = 0 against H1: pi1 - pi0 <> 0
prop.diff.test(alco)

## The same
n.yes <- alco[, "yes"]
n.total <- apply(alco, 1, sum)
prop.diff.test(x=n.yes, n=n.total)

## The same using standard function prop.test
## (the confidence interval is for pi0 - pi1
## and not for pi1 - pi0)
prop.test(x=n.yes, n=n.total, correct=FALSE)

## Test of
## H0: pi1 - pi0 = 0.15 against pi1 - pi0 > 0.15
prop.diff.test(alco, diff=0.15, alternative="greater")

```

prop.Z.test

One-sample test and confidence interval for the proportion

Description

Performs one-sample test on the proportion and computes the corresponding confidence interval for the proportion.

Usage

```

prop.Z.test(x, n, p=0.5,
            alternative=c("two.sided", "less", "greater"), conf.level=0.95)

## S3 method for class 'prop.Z.test':
print(x, ...)

```

Arguments

x	a count of successes.
n	a count of trials.
p	probability of success under the null hypothesis (π_1^*).
alternative	a character string specifying the alternative hypothesis, must be one of “two.sided” (default), “greater” or “less”. You can specify just the initial letter.
conf.level	confidence level of the confidence interval for the probability of success π_1 which is also constructed.
...	further arguments to be passed to or from methods.

Details

For a count N_1 sampled from a binomial distribution $\text{Binom}(n; \pi_1)$, it tests the null hypothesis

$$H_0 : \pi_1 = \pi_1^*$$

using the normal approximation to the Z statistic, i.e.

$$Z = \frac{p_1 - \pi_1^*}{\text{se}(p_1)},$$

where

$$p_1 = \frac{N_1}{n}$$

and

$$\text{se}(p_1) = \sqrt{\frac{\pi_1^*(1 - \pi_1^*)}{n}}.$$

The function also produces the confidence interval for π_1 based either on the Z statistic with $\text{se}(p_1)$ replaced by its estimate

$$\widehat{\text{se}}(p_1) = \sqrt{\frac{p_1(1 - p_1)}{n}}$$

or on the Z statistic itself.

See Komárek a Komárková (2007, Chap. 2) for details.

Value

A list with class `prop.Z.test` containing the following components:

<code>Parameter</code>	estimate of the probability of success π_1 .
<code>SE.parameter</code>	estimated standard error of $\hat{\pi}_1$.
<code>lower</code>	lower limit of the confidence interval for π_1 based on the inversion of the Z statistic with estimated standard error.
<code>upper</code>	upper limit of the confidence interval for π_1 based on the inversion of the Z statistic with estimated standard error.
<code>lower.inv</code>	lower limit of the confidence interval for π_1 based on the inversion of the Z statistic with exact standard error. By default, the <code>print</code> method shows this interval.
<code>upper.inv</code>	upper limit of the confidence interval for π_1 based on the inversion of the Z statistic with exact standard error. By default, the <code>print</code> method shows this interval.
<code>p.success</code>	estimate of the probability of success π_1 .
<code>n</code>	the count of trials
<code>n.success</code>	the count of successes
<code>Z</code>	the value of the test statistic.
<code>p.value</code>	the P-value for the test.
<code>p</code>	probability of success under the null hypothesis (π_1^*).
<code>typ</code>	a number specifying the alternative hypothesis.
<code>alternative</code>	a character string specifying the alternative hypothesis.
<code>conf.level</code>	confidence level of the confidence interval for θ .

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislostí s příklady v R.
Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[prop.test](#).

Examples

```
## 166 successes out of 170 trials observed
## H0: pi1 = 0.97 against H1: pi1 > 0.97
prop.z.test(x=166, n=170, alt="greater")

## The same using standard function prop.test
prop.test(x=166, n=170, alt="greater", correct=FALSE)
```

rdiscrete

Pseudo-random numbers from a discrete distribution

Description

It produces pseudo-random numbers from a discrete distribution specified by its probability function.

Usage

```
rdiscrete(n, probs, labels)
```

Arguments

n	number of pseudo-random numbers to generate.
probs	vector of probabilities specifying the distribution we want to sample from. Probabilities can be specified up to a proportionality constant.
labels	optional vector of labels for categories from which we sample from.

Value

Vector of sampled values.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

See Also

[sample](#).

Examples

```
z1 <- rdiscrete(1000, probs=c(0.1, 0.3, 0.25, 0.35))
prop.table(table(z1))

z2 <- rdiscrete(1000, probs=c(10, 30, 25, 35))
prop.table(table(z2))

z3 <- rdiscrete(1000, probs=c(10, 30, 25, 35),
  labels=c("red", "orange", "yellow", "green"))
prop.table(table(z3))
```

rel.risk.test	<i>Two-sample test and confidence interval for the relative risk</i>
---------------	--

Description

Performs a two-sample test on the relative risk (ratio of proportions) and computes corresponding confidence interval for the relative risk.

Usage

```
rel.risk.test(x, n, rr=1,
  alternative=c("two.sided", "less", "greater"), conf.level=0.95)

## S3 method for class 'rel.risk.test':
print(x, ...)
```

Arguments

x	a two-component vector of counts of successes in the two groups, i.e. the vector $(N_{0,1}, N_{1,1})'$ or a 2 by 2 matrix with the following entries <div style="text-align: center; margin: 10px 0;"> $\begin{array}{ccc ccc} & & & n_0 - N_{0,1} & N_{0,1} & \\ & & & n_1 - N_{1,1} & N_{1,1} & \end{array}$ </div>
n	a two-component vector of counts of trials in the two groups, i.e. the vector $(n_0, n_1)'$. It is ignored if x is a matrix.
rr	the relative risk under the null hypothesis (θ^*).
alternative	a character string specifying the alternative hypothesis, must be one of “two.sided” (default), “greater” or “less”. You can specify just the initial letter.
conf.level	confidence level of the confidence interval for the relative risk π_1/π_0 which is also constructed.

... further arguments to be passed to or from methods.

Details

For counts $N_{0,1}$ and $N_{1,1}$ sampled independently from binomial distributions $\text{Binom}(n_0; \pi_0)$ and $\text{Binom}(n_1; \pi_1)$, respectively, it tests the null hypothesis

$$H_0 : \frac{\pi_1}{\pi_0} = \theta^*$$

using the normal approximation to the corresponding Z statistic, i.e.

$$Z = \frac{\log(\hat{\theta}) - \log(\theta^*)}{\widehat{\text{se}}\{\log(\hat{\theta})\}},$$

where $\hat{\theta}$ is the observed relative risk. That is,

$$\hat{\theta} = \frac{p_1}{p_0}, \quad p_1 = \frac{N_{1,1}}{n_1}, \quad p_0 = \frac{N_{0,1}}{n_0}.$$

The function also produces the confidence interval for the relative risk $\theta = \pi_1/\pi_0$ based on the above Z statistic.

See Komárek a Komárková (2007, Sec. 4.3) for details.

Value

A list with class `rel.risk.test` containing the following components:

<code>Parameter</code>	estimate of the relative risk $\theta = \pi_1/\pi_0$.
<code>Log.parameter</code>	estimate of the log-relative risk $\log(\theta)$.
<code>SE.log.parameter</code>	estimated standard error of $\log(\hat{\theta})$.
<code>lower</code>	lower limit of the confidence interval for θ based on the inversion of the Z statistic.
<code>upper</code>	upper limit of the confidence interval for θ based on the inversion of the Z statistic.
<code>log.lower</code>	lower limit of the confidence interval for $\log(\theta)$ based on the inversion of the Z statistic.
<code>log.upper</code>	upper limit of the confidence interval for $\log(\theta)$ based on the inversion of the Z statistic.
<code>p.success</code>	estimated probabilities of success in the two groups.
<code>n</code>	the counts of trials in the two groups.
<code>n.success</code>	the counts of successes in the two groups.
<code>Z</code>	the value of the test statistic.
<code>p.value</code>	the P-value for the test.
<code>rr</code>	the relative risk under the null hypothesis (θ^*).

`typ` a number specifying the alternative hypothesis.
`alternative` a character string specifying the alternative hypothesis.
`conf.level` confidence level of the confidence interval for θ .
`data.name` name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislosti s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[prop.test](#), [prop.diff.test](#), [odds.ratio.test](#).

Examples

```

### Opinion over own alcoholism
### (General Social Survey 1993)
### Do you sometimes drink too much alcohol?
### Answers further cross-classified w.r.t. gender.
alco <- matrix(c(303, 188, 106, 134), nrow=2, ncol=2)
rownames(alco) <- c("female", "male")
colnames(alco) <- c("no", "yes")
print(alco)

### pi0 = P(drink too much | female)
### pi1 = P(drink too much | male)
### H0: pi1/pi0 = 1 against H1: pi1/pi0 <> 1
rel.risk.test(alco)

## The same
n.yes <- alco[, "yes"]
n.total <- apply(alco, 1, sum)
rel.risk.test(x=n.yes, n=n.total)

## Test of
## H0: pi1/pi0 = 1.5 against pi1/pi0 > 1.5
rel.risk.test(alco, rr=1.5, alternative="greater")

```

resplot

*Residual plots***Description**

Draws several residual plots for linear models and time series models.

Usage

```
resplot(object, ...)

## Default S3 method:
resplot(object, ...)

## S3 method for class 'lm':
resplot(object, type=c("e-yhat", "e-x", "e-time"), xterm,
        restype=c("raw", "standard"), lowess=TRUE, hline=FALSE,
        col="black", lcol="black", pch, lty, lwd, xlab, ylab, main, ...)

## S3 method for class 'tsfit':
resplot(object, lowess=TRUE, hline=FALSE, type="l", pch, lty, lwd,
        xlab, ylab, main, col="black", lcol="black", ...)
```

Arguments

object	an object of class <code>lm</code> or of class <code>tsfit</code> .
type	<p>If object has a class <code>lm</code>: type of the residual plot to draw:</p> <p>e-yhat residuals versus fitted values. e-x residuals versus predictor. The predictor is further specified by the argument <code>xterm</code>. Note, that if the plot of residuals versus predictor is required the object must be obtained by a call to the <code>lm</code> function with its argument <code>x</code> set to <code>TRUE</code>. e-time residuals versus the sequence $1, \dots, n$, where n is the number of observations.</p> <p>If object has a class <code>tsfit</code>: a character string specifying the graphical type of the plot (“l” for lines, “p” for points etc.).</p>
xterm	a character string which specifies the predictor if <code>type</code> is “e-x”. See examples below for more details.
restype	<p>a character string specifying the type of residuals to draw in the plot:</p> <p>raw raw residuals obtained with <code>residuals.lm</code>. standard standardized residuals obtained with <code>rstandard.lm</code>.</p>

lowess	logical value indicating whether a lowess curve should be added to the residual plot.
hline	logical value indicating whether a horizontal line going through zero should be added to the residual plot.
col	character string specifying the color for the symbols in the residual plot.
lcol	character string specifying the color for added lines (lowess and horizontal line going through zero).
pch	graphical parameter passed to the plot function.
lty	graphical parameter passed to the plot function.
lwd	graphical parameter passed to the plot function.
xlab	graphical parameter passed to the plot function.
ylab	graphical parameter passed to the plot function.
main	graphical parameter passed to the plot function.
...	further arguments to be passed to or from methods.

Details

The function draws residual plots for models fitted using functions returning objects of class `lm` or returning objects of class `tsfit` (functions [ts.trend](#), [ts.masMOOTH](#), [ts.simpleseason.trend](#)).

Value

`invisible(object)`.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

See Also

[lm](#), [ts.trend](#), [ts.masMOOTH](#), [ts.simpleseason.trend](#), [residuals.lm](#), [rstandard.lm](#), [residuals.ts.trend](#), [residuals.ts.masMOOTH](#), [residuals.ts.simpleseason.trend](#).

Examples

```
##### Examples for an object of class lm:
##### =====
data(cars04)
lmodel <- lm(lkmCity~weight+I(log(horsePower)), data=cars04, x=TRUE)

## Raw residuals
oldPar <- par(mfrow=c(2, 2), bty="n")
resplot(lmodel, type="e-yhat", col="red")
resplot(lmodel, type="e-x", xterm="weight", col="red")
resplot(lmodel, type="e-x", xterm="I(log(horsePower))", col="red")
resplot(lmodel, type="e-time", col="red")
par(oldPar)
```

```

## Standardized residuals
oldPar <- par(mfrow=c(2, 2), bty="n")
resplot(lmodel, type="e-yhat", restype="standard", col="red")
resplot(lmodel, type="e-x", xterm="weight",
        restype="standard", col="red")
resplot(lmodel, type="e-x", xterm="I(log(horsePower))",
        restype="standard", col="red")
resplot(lmodel, type="e-time", restype="standard", col="red")
par(oldPar)

##### Examples for an object of class tsfit:
##### =====
### Example 1a
data(tsCPI)
fit <- ts.trend(tsCPI, trend="modified.exponential")
resplot(fit, col="darkblue", bty="n", xlab="Year")
resplot(fit, type="p", col="red", bty="n", xlab="Year")

### Example 1b
fit7 <- ts.msmooth(tsCPI, length=7)
resplot(fit7, col="darkblue", bty="n", xlab="Year")
resplot(fit7, type="p", col="red", bty="n", xlab="Year")

### Example 1c
fit7c <- ts.msmooth(tsCPI, length=6)
resplot(fit7c, lowess=FALSE, col="darkblue", bty="n", xlab="Year")
resplot(fit7c, lowess=FALSE, type="p", col="red", bty="n", xlab="Year")

### Example 2
data(tsAccident)
fitAc3 <- ts.simpleseason.trend(tsAccident,
    decomposition="additive", trend="quadratic")
resplot(fitAc3, col="darkblue", bty="n", xlab="Year")
resplot(fitAc3, type="p", col="red", bty="n", xlab="Year")

### Example 3
data(tsAirpass)
fitAirt3 <- ts.simpleseason.trend(tsAirpass,
    decomposition="multiplicative", trend="gompertz")
resplot(fitAirt3, col="darkblue", bty="n", xlab="Year")
resplot(fitAirt3, type="p", col="red", bty="n", xlab="Year")

```

Description

Computes semistudentized residuals for the linear model.

Usage

```
rsemistudent(model)

## Default S3 method:
rsemistudent(model)

## S3 method for class 'lm':
rsemistudent(model)
```

Arguments

model an object with class `lm`.

Details

Let e_i ($i = 1, \dots, n$) be the residuals in the linear model and let MSE denote the mean squared error of the model. The semistudentized residuals are defined as

$$e_i^* = \frac{e_i}{\sqrt{MSE}} \quad (i = 1, \dots, n).$$

See Neter et al. (1996, p. 98, 369) for details.

Value

A numeric vector with computed semistudentized residuals.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Neter, J., Kutner, M. H., Nachtsheim, C. J., and Wasserman, W. (1996). *Applied Linear Statistical Models, 4th Edition*. Boston: WCB/McGraw-Hill.

See Also

[rstandard](#), [rstudent](#).

Examples

```
data(cars04)
model <- lm(lkmCity~I(log(weight)), data=cars04)

semistRes <- rsemistudent(model)
oldPar <- par(mfrow=c(1, 2), bty="n")
boxplot(semistRes, ylab="Semistudentized residual", col="seagreen2")
hist(semistRes, xlab="Semistudentized residual", ylab="Density",
     prob=TRUE, col="seagreen2", main="")
par(oldPar)
```

`stuart.test`*Stuart chi-squared test of marginal homogeneity in a squared table*

Description

Performs Stuart chi-squared test of marginal homogeneity in a squared table.

Usage

```
stuart.test(x)

## S3 method for class 'stuart.test':
print(x, ...)
```

Arguments

<code>x</code>	table of observed frequencies (in the matrix form).
<code>...</code>	further arguments to be passed to or from methods.

Details

Stuart χ^2 test of marginal homogeneity in a squared table. For 2 by 2 table, it is the same as McNemar's test.

See Stuart (1955) or Komárek a Komárková (2007, Sec. 7.5).

Value

A list with class `stuart.test` containing the following components:

<code>w</code>	the value of the test statistic.
<code>df</code>	degrees of freedom of the χ^2 distribution of the test statistic.
<code>p.val</code>	the P-value for the test.
<code>p.hat</code>	estimated marginal category probabilities.
<code>category.names</code>	names of the response categories.
<code>data.name</code>	name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Stuart, A. (1955). A test for homogeneity of the marginal distributions in a two-way classification. *Biometrika*, **42**, 412-416.

Komárek, A. a Komárková, L. (2007). *Statistická analýza závislosti s příklady v R*. Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[bhapkar.test](#), [mcnemar.test](#).

Examples

```
## Yule and Kendall (1950). Father's and son's eye color:
eye.color <- matrix(c(194, 83, 25, 56,
                    70, 124, 34, 36,
                    41, 41, 55, 43,
                    30, 36, 23, 109), ncol=4)
rownames(eye.color) <- colnames(eye.color) <- c("blue",
        "blue-green", "darkgrey", "darkbrown")
print(eye.color)
stuart.test(eye.color)

## Agresti (2002), Table 10.6
migration <- matrix(c(11607, 87, 172, 63,
                    100, 13677, 255, 176,
                    366, 515, 17819, 286,
                    124, 302, 270, 10192), ncol=4)
rownames(migration) <- colnames(migration) <- c("Northeast",
        "Midwest", "South", "West")
print(migration)
stuart.test(migration)
```

symmetry.test	<i>Bowker chi-squared test of symmetry in a squared table</i>
---------------	---

Description

Performs Bowker chi-squared test of symmetry in a squared table.

Usage

```
symmetry.test(x)

## S3 method for class 'symmetry.test':
print(x, ...)
```

Arguments

x	table of observed frequencies (in the matrix form).
...	further arguments to be passed to or from methods.

Details

Bowker χ^2 test of symmetry in a squared table. The function adds 1/2 to the off-diagonal elements if some of them are equal to zero. For 2 by 2 table, it is the same as McNemar's test.

See Bowker (1948) or Komárek a Komárková (2007, Sec. 7.6).

Value

A list with class `symmetry.test` containing the following components:

<code>x2</code>	the value of the test statistic.
<code>df</code>	degrees of freedom of the χ^2 distribution of the test statistic.
<code>p.val</code>	the P-value for the test.
<code>observed</code>	the matrix of observed counts.
<code>expected</code>	the matrix of expected counts under the null hypothesis.
<code>residuals</code>	Pearson residuals.
<code>data.name</code>	name of the input data.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Bowker, A. H. (1948). A test for symmetry in contingency tables. *Journal of the American Statistical Association*, **43**, 572-574.

Komárek, A. a Komárková, L. (2007). *Statistická analýza závislosti s příklady v R*. Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[mcnemar.test](#).

Examples

```
## Agresti (2002), Table 10.6
migration <- matrix(c(11607, 87, 172, 63,
                    100, 13677, 255, 176,
                    366, 515, 17819, 286,
                    124, 302, 270, 10192), ncol=4)
rownames(migration) <- colnames(migration) <- c("Northeast",
        "Midwest", "South", "West")
print(migration)
symmetry.test(migration)

## The same
mcnemar.test(migration)
```

ts.explore

Exploratory plots for non-seasonal time series

Description

Creates several exploratory plots for non-seasonal time series.

Usage

```
ts.explore(x, lty.lowess=1, col.lowess="red",
           type="l", lty=1, pch=1, bty="n", xlab="Time", ...)
```

Arguments

x	a numeric vector or an object of class <code>ts</code> holding the time series.
lty.lowess	type of the line for lowess smoothing of the plot of differences or indeces.
col.lowess	color of the line for lowess smoothing of the plot of differences or indeces.
type	graphical parameter passed to the <code>plot</code> function.
lty	graphical parameter passed to the <code>plot</code> function.
pch	graphical parameter passed to the <code>plot</code> function.
bty	graphical parameter passed to the <code>plot</code> function.
xlab	graphical parameter passed to the <code>plot</code> function.
...	further arguments to be passed to or from methods.

Details

For a time series x_t ($t = 1, \dots, n$) the function produces six plots of differences or indices which, if (approximately) constant, indicate the trend in the time series. Namely, the following differences and indeces are plotted, whose constancy indicates the following trends:

linear:	$x_{t+1} - x_t$
quadratic:	$x_{t+2} - 2x_{t+1} + x_t$
exponential:	$\frac{x_{t+1}}{x_t}$
modified exponential:	$\frac{x_{t+2} - x_{t+1}}{x_{t+1} - x_t}$
logistic:	$\frac{1/x_{t+2} - 1/x_{t+1}}{1/x_{t+1} - 1/x_t}$
Gompertz curve:	$\frac{\log(x_{t+2}) - \log(x_{t+1})}{\log(x_{t+1}) - \log(x_t)}$

Created plots are further smoothed using the `lowess` method.

See Komárek a Komárková (2007, Sec. 13.1) for details.

Value

Besides producing the plots function returns `invisible(x)`.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislostí s příklady v R.
Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[ts.trend](#), [ts.fit.trends](#), [ts](#).

Examples

```
data(tsCPI)
ts.explore(tsCPI)
ts.explore(tsCPI, type="p")
```

`ts.fit.trends`

Estimate several trends in the non-seasonal time series

Description

Fits several trends to the non-seasonal time series.

Usage

```
ts.fit.trends(x)

## S3 method for class 'ts.fit.trends':
print(x, ...)

## S3 method for class 'ts.fit.trends':
plot(x, type.ts="l", lty.ts=4, pch.ts=1, col.ts="black",
      type.fit="l", lty.fit=1, pch.fit=16, col.fit="red",
      add.point.ts=FALSE, add.point.fit=FALSE, pch.add=15,
      digits=4, main, sub, bty, xlab, ylab, xlim, ylim, ...)
```

Arguments

<code>x</code>	a numeric vector or an object of class <code>ts</code> holding the time series. For methods <code>print</code> and <code>plot</code> , <code>x</code> is an object with class <code>ts.fit.trends</code> .
<code>type.ts</code>	character specifying the type of the plot to draw the original time series.
<code>lty.ts</code>	value which specifies the type of the line to draw the original time series if <code>type.ts</code> is "l".
<code>pch.ts</code>	value which specifies the plotting character to draw the original time series if <code>type.ts</code> is "p" or if <code>add.point.ts</code> is TRUE.
<code>col.ts</code>	character which specifies the color to draw the original time series.
<code>type.fit</code>	character specifying the type of the plot to draw the fitted trend.
<code>lty.fit</code>	value which specifies the type of the line to draw the fitted trend if <code>type.fit</code> is "l".
<code>pch.fit</code>	value which specifies the plotting character to draw the fitted trend if <code>type.fit</code> is "p" or if <code>add.point.fit</code> is TRUE.
<code>col.fit</code>	character which specifies the color to draw the fitted trend.
<code>add.point.ts</code>	logical which indicates whether the original time series should (also) be indicated by the points.
<code>add.point.fit</code>	logical which indicates whether the fitted trend should (also) be indicated by the points.
<code>pch.add</code>	value indicating the plotting character used if either <code>add.point.ts</code> or <code>add.point.fit</code> is TRUE.
<code>digits</code>	a value by which the MSE and MAE are rounded to produce subtitle (if not supplied by the user).
<code>main</code>	graphical parameter passed to the <code>plot</code> function.
<code>sub</code>	graphical parameter passed to the <code>plot</code> function.
<code>bty</code>	graphical parameter passed to the <code>plot</code> function.
<code>xlab</code>	graphical parameter passed to the <code>plot</code> function.
<code>ylab</code>	graphical parameter passed to the <code>plot</code> function.
<code>xlim</code>	graphical parameter passed to the <code>plot</code> function.
<code>ylim</code>	graphical parameter passed to the <code>plot</code> function.
<code>...</code>	further arguments to be passed to or from methods.

Details

For a time series y_t ($t = 1, \dots, n$) the following decomposition is assumed:

$$y_t = T_t + \varepsilon_t \quad (t = 1, \dots, n),$$

where T_t is the trend and ε_t is the white noise. The function fits the following six trends using the least squares:

Linear: $T_t = \alpha + \beta t$ ($t = 1, \dots, n$).

Quadratic: $T_t = \alpha + \beta t + \gamma t^2$ ($t = 1, \dots, n$).

Exponential: $T_t = \alpha \cdot \beta^t$ ($t = 1, \dots, n$).

Modified exponential: $T_t = \gamma + \alpha \cdot \beta^t$ ($t = 1, \dots, n$).

Logistic: $T_t = \frac{\gamma}{1 + \alpha \cdot \beta^t}$ ($t = 1, \dots, n$).

Gompertz curve: $T_t = \gamma \cdot \alpha^{\beta^t}$ ($t = 1, \dots, n$).

The `plot` method produces a figure with the original time series overlaid by the fitted trends.

See Komárek a Komárková (2007, Sec. 13.1) for details.

Value

A data frame with the class `ts.fit.trends` having the following columns:

<code>mse</code>	mean squared errors (MSE) for the fitted trends.
<code>mae</code>	mean absolute errors (MAE) for the fitted trends.
<code>me</code>	mean errors (ME) for the fitted trends.
<code>mpe</code>	mean percentage errors (MPE) for the fitted trends.
<code>mape</code>	mean absolute percentage errors (MAPE) for the fitted trends.

The object has further attributes `fit`, `trend`, `ntrend` holding more detailed information on the fitted trends.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislosti s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[ts.trend](#), [ts](#).

Examples

```

data(tsCPI)
fitT <- ts.fit.trends(tsCPI)
print(fitT)
plot(fitT, xlab="Year", ylab="CPI", bty="n")
plot(fitT, type.ts="p", xlab="Year", ylab="CPI", bty="n")

```

ts.masmooth

*Smoothing of the time series using the moving average***Description**

It smooths the time series using the moving average.

Usage

```

ts.masmooth(x, order=0, length)

## S3 method for class 'ts.masmooth':
print(x, print.fitted=TRUE, round, ...)

## S3 method for class 'ts.masmooth':
predict(object, step=1, interval="none", level=0.95, ...)

## S3 method for class 'ts.masmooth':
residuals(object, ...)

## S3 method for class 'ts.masmooth':
fitted(object, ...)

```

Arguments

x	a numeric vector or an object of class <code>ts</code> holding the time series. For method <code>print</code> , x is an object with class <code>ts.masmooth</code> .
order	order of the moving average.
length	length of the moving average.
print.fitted	logical indicating whether fitted values should be printed.
round	a numeric value indicating possible rounding of the fitted values when printing.
object	an object with class <code>ts.trend</code> .
step	a numeric vector indicating for how many steps in the future we want to predict.
interval	type of the confidence interval. Currently, only “none” is implemented.
level	confidence level of the confidence interval.
...	further arguments to be passed to or from methods.

Details

For odd `length` of $2M + 1$, **simple** moving average of given `order` is used. For even `length` of $2M$, the **centered** moving average of length $2M + 1$ is used and `order` is ignored in this case.

Among others, there are functions implemented to predict (for simple moving averages only), compute residuals and fitted values and to plot the smoothed (fitted) time series.

See Komárek a Komárková (2007, Sec. 13.2) for details.

Value

A list with the classes `ts.masmooth` and `tsfit` having the following components:

<code>fitted</code>	a numeric vector with fitted values.
<code>residuals</code>	a numeric vector with residuals.
<code>sse</code>	sum of squared errors (SSE) of the fitted trend.
<code>mse</code>	mean squared error (MSE) of the fitted trend.
<code>mae</code>	mean absolute error (MAE) of the fitted trend.
<code>me</code>	mean error (ME) of the fitted trend.
<code>mpe</code>	mean percentage error (MPE) of the fitted trend.
<code>mape</code>	mean absolute percentage error (MAPE) of the fitted trend.
<code>order</code>	order of the moving average.
<code>length</code>	length of the moving average.
<code>H</code>	hat matrix which can be used to compute simple moving average on the boundary.
<code>centered</code>	logical which indicates whether centered or simple moving averages were used to compute fitted values.

The object has further the attribute `ts` which holds the original time series.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislostí s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[plot.tsfit](#), [resplot.tsfit](#), [masmooth.coef](#), [white.noise.test](#), [ts](#).

Examples

```

data(tsCPI)

##### Smoothing with simple moving average of order 0 and length 3
##### =====
fit3 <- ts.masmooth(tsCPI, length=3)

print(fit3)
predict(fit3, step=1:5)

### Plot of the time series with fitted values and
### plot of residuals
oldPar <- par(mfrow=c(1, 2), bty="n")
plot(tsCPI, xlab="Year", ylab="CPI")
lines(as.numeric(time(tsCPI)), fitted(fit3), col="red")
plot(as.numeric(time(tsCPI)), residuals(fit3),
     type="l", xlab="Year", ylab="Residual", col="darkgreen")
par(oldPar)

### plot method for the fitted model
plot(fit3, add.point.fit=FALSE, xlab="Year", ylab="CPI")

### resplot method for the fitted model
resplot(fit3, xlab="Year", ylab="Residual", col="darkgreen", bty="n")

##### Smoothing with centered moving average of length 3
##### =====
fit3c <- ts.masmooth(tsCPI, length=2)

print(fit3c)

### Plot of the time series with fitted values and
### plot of residuals
oldPar <- par(mfrow=c(1, 2), bty="n")
plot(tsCPI, xlab="Year", ylab="CPI")
lines(as.numeric(time(tsCPI)), fitted(fit3c), col="red")
plot(as.numeric(time(tsCPI)), residuals(fit3c),
     type="l", xlab="Year", ylab="Residual", col="darkgreen")
par(oldPar)

### plot method for the fitted model
plot(fit3c, add.point.fit=FALSE, xlab="Year", ylab="CPI")

### resplot method for the fitted model
resplot(fit3c, xlab="Year", ylab="Residual", col="darkgreen", bty="n",
       lowess=FALSE)

```

ts.simpleseason.trend

Simple analysis of the seasonal time series

Description

Performs a simple analysis of the seasonal time series.

Usage

```
ts.simpleseason.trend(x, decomposition="additive", trend=NULL)

## S3 method for class 'ts.simpleseason.trend':
print(x, print.fitted=TRUE, round, ...)

## S3 method for class 'ts.simpleseason.trend':
predict(object, step=1, interval="none", level=0.95, ...)

## S3 method for class 'ts.simpleseason.trend':
residuals(object, ...)

## S3 method for class 'ts.simpleseason.trend':
fitted(object, ...)
```

Arguments

x	an object of class <code>ts</code> holding the time series.
decomposition	a character string which specifies the type of the decomposition. It can be one of the following: “additive” or “multiplicative”.
trend	a character string which specifies the type of the trend. It can be one of the following: “linear”, “quadratic”, “exponential”, “modified.exponential”, “logistic”, “gompertz”. If <code>trend</code> is equal to <code>NULL</code> , only the first part of the analysis is performed. That is, a provisional trend is estimated using the moving average, seasonal factors are computed and the seasonal effect is removed from the time series.
print.fitted	logical indicating whether fitted values should be printed.
round	a numeric value indicating possible rounding of the fitted values when printing.
object	an object with class <code>ts.trend</code> .
step	a numeric vector indicating for how many steps in the future we want to predict.
interval	type of the confidence interval. Currently, only “none” is implemented.
level	confidence level of the confidence interval.
...	further arguments to be passed to or from methods.

Details

For a seasonal time series $y_{k,j}$ ($k = 1, \dots, s$, $j = 1, \dots, r$), where r is the number of periods within a season, one of the following models is assumed:

Additive decomposition:

$$y_{k,j} = T_{k,j} + S_j + \varepsilon_{k,j} \quad (k = 1, \dots, s, j = 1, \dots, r),$$

where $T_{k,j}$ is the trend, S_j are the seasonal components satisfying $\sum_{j=1}^r S_j = 0$, and $\varepsilon_{k,j}$ is the white noise.

Multiplicative decomposition:

$$y_{k,j} = T_{k,j} \cdot S_j + \varepsilon_{k,j} \quad (k = 1, \dots, s, j = 1, \dots, r),$$

where $T_{k,j}$ is the trend, S_j are the seasonal components satisfying $\sum_{j=1}^r S_j = r$, and $\varepsilon_{k,j}$ is the white noise.

Estimation proceeds in the following steps:

1. The trend is provisionally estimated by smoothing with the moving average of the length corresponding to the length of the season.
2. Provisional trend is removed from the time series and the seasonal components are estimated using the method of mean seasonal indices.
3. The original time series is cleaned from the seasonal component and final trend is chosen and estimated using the least squares.

Among others, there are functions implemented to predict, compute residuals and fitted values and to plot the smoothed (fitted) time series.

See Komárek a Komárková (2007, Chap. 14) for details.

Value

A list with the classes `ts.simpleseason.trend` and `tsfit` whose structure depends on whether the argument `trend` is `NULL` or not.

The list has further the attributes `no.trend`, `decomposition`, `season`, `ts`.

Value if trend argument is NULL

A list with the following components:

result a data frame with the following columns:

time a vector indicating the principal time (season).

period a vector indicating the secondary time (periods within season).

time.series original time series.

season fitted seasonal component.

d original time series cleaned from the seasonal effect.

Value if trend argument is a character string

A list with the following components:

result a data frame with the following columns:

time a vector indicating the principal time (season).

period a vector indicating the secondary time (periods within season).

time.series original time series.

fitted fitted values of the time series.
season fitted seasonal component.
trend fitted trend.
residuals residuals based on the fitted model.
sse sum of squared errors (SSE) of the fitted time series.
mse mean squared error (MSE) of the fitted time series.
mae mean absolute error (MAE) of the fitted time series.
me mean error (ME) of the fitted time series.
mpe mean percentage error (MPE) of the fitted time series.
mape mean absolute percentage error (MAPE) of the fitted time series.
coef a numeric vector with estimated coefficients of the trend.
trend character string indicating the fitted trend.
ctrend character string indicating the mathematical expression of the fitted trend.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislostí s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[plot.tsfit](#), [resplot.tsfit](#), [ts.trend](#), [ts.fit.trends](#), [ts.masmooth](#), [white.noise.test](#),
[ts](#).

Examples

```

##### Analysis of the time series with ADDITIVE decomposition
##### =====
data(tsAccident)
plot(tsAccident, bty="n", col="darkgreen", xlab="Year", ylab="#deaths")

## Elimination of the seasonal component
fitAcl <- ts.simpleseason.trend(tsAccident, decomposition="additive")
print(fitAcl, print.fitted=FALSE)
print(fitAcl)

## Plot seasonally cleaned time series
## (together with the original time series)
plot(fitAcl, xlab="Year", ylab="#deaths")

## Fit different trends to seasonally cleaned time series
## to see which one leads to minimal MSE

```

```

fitAc2 <- ts.fit.trends(fitAc1$result$d)
print(fitAc2)

## Fit the model with quadratic trend
fitAc3 <- ts.simpleseason.trend(tsAccident,
  decomposition="additive", trend="quadratic")
print(fitAc3, print.fitted=FALSE)
print(fitAc3)

## Prediction for the following 12 months
predict(fitAc3, step=1:12)

## Plot the fitted time series, fitted trend,
## original time series and prediction for 12 months
plot(fitAc3, step=1:12, xlab="Year", ylab="#deaths")

## Plot residuals with LOWESS smoother
resplot(fitAc3, xlab="Year", ylab="Residual", col="darkblue")

##### Analysis of the time series with MULTIPLICATIVE decomposition
##### =====
data(tsAirpass)
plot(tsAirpass, bty="n", col="darkgreen",
  xlab="Year", ylab="#passangers (x1000)")

## Elimination of the seasonal component
fitAirp1 <- ts.simpleseason.trend(tsAirpass,
  decomposition="multiplicative")
print(fitAirp1, print.fitted=FALSE)
print(fitAirp1)

## Plot seasonally cleaned time series
## (together with the original time series)
plot(fitAirp1, xlab="Year", ylab="#passangers (x1000)")

## Fit different trends to seasonally cleaned time series
## to see which one leads to minimal (or reasonable) MSE
fitAirp2 <- ts.fit.trends(fitAirp1$result$d)
print(fitAirp2)

## Fit the model with Gompertz curve as trend
fitAirp3 <- ts.simpleseason.trend(tsAirpass,
  decomposition="multiplicative", trend="gompertz")
print(fitAirp3, print.fitted=FALSE)
print(fitAirp3)

## Prediction for the following 12 months
predict(fitAirp3, step=1:12)

## Plot the fitted time series, fitted trend,
## original time series and prediction for 12 months
plot(fitAirp3, step=1:12, xlab="Year", ylab="#passangers (x1000)")

```

```
## Plot residuals with LOWESS smoother
resplot(fitAirp3, xlab="Year", ylab="Residual", col="darkblue")
```

ts.trend

Estimate the trend in the non-seasonal time series

Description

Fits a trend to the non-seasonal time series.

Usage

```
ts.trend(x, trend="linear")

## S3 method for class 'ts.trend':
print(x, print.fitted=TRUE, round, ...)

## S3 method for class 'ts.trend':
predict(object, step=1, interval="none", level=0.95, ...)

## S3 method for class 'ts.trend':
residuals(object, ...)

## S3 method for class 'ts.trend':
fitted(object, ...)
```

Arguments

x	a numeric vector or an object of class <code>ts</code> holding the time series. For method <code>print</code> , x is an object with class <code>ts.trend</code> .
trend	a character indicating the type of the trend we want to fit. It can be one of the following: “linear”, “quadratic”, “exponential”, “modified.exponential”, “logistic”, “gompertz”.
print.fitted	logical indicating whether fitted values should be printed.
round	a numeric value indicating possible rounding of the fitted values when printing.
object	an object with class <code>ts.trend</code> .
step	a numeric vector indicating for how many steps in the future we want to predict.
interval	type of the confidence interval. Currently, only “none” is implemented.
level	confidence level of the confidence interval.
...	further arguments to be passed to or from methods.

Details

For a time series y_t ($t = 1, \dots, n$) the following decomposition is assumed:

$$y_t = T_t + \varepsilon_t \quad (t = 1, \dots, n),$$

where T_t is the trend and ε_t is the white noise. The function fits one of the following trends using the least squares:

Linear: $T_t = \alpha + \beta t \quad (t = 1, \dots, n).$

Quadratic: $T_t = \alpha + \beta t + \gamma t^2 \quad (t = 1, \dots, n).$

Exponential: $T_t = \alpha \cdot \beta^t \quad (t = 1, \dots, n).$

Modified exponential: $T_t = \gamma + \alpha \cdot \beta^t \quad (t = 1, \dots, n).$

Logistic: $T_t = \frac{\gamma}{1 + \alpha \cdot \beta^t} \quad (t = 1, \dots, n).$

Gompertz curve: $T_t = \gamma \cdot \alpha^{\beta^t} \quad (t = 1, \dots, n).$

Among others, there are functions implemented to predict, compute residuals and fitted values and to plot the fitted trend.

See Komárek a Komárková (2007, Sec. 13.1) for details.

Value

A list with the classes `ts.trend` and `tsfit` having the following components:

<code>fitted</code>	a numeric vector with fitted values.
<code>residuals</code>	a numeric vector with residuals.
<code>coef</code>	a numeric vector with estimated coefficients of the trend.
<code>sse</code>	sum of squared errors (SSE) of the fitted trend.
<code>mse</code>	mean squared error (MSE) of the fitted trend.
<code>mae</code>	mean absolute error (MAE) of the fitted trend.
<code>me</code>	mean error (ME) of the fitted trend.
<code>mpe</code>	mean percentage error (MPE) of the fitted trend.
<code>mape</code>	mean absolute percentage error (MAPE) of the fitted trend.
<code>init.coef</code>	initial values of the trend coefficients used by the optimization routine.
<code>init.sse</code>	initial value of the SSE.
<code>init.mse</code>	initial value of the MSE.
<code>init.mae</code>	initial value of the MAE.
<code>init.me</code>	initial value of the ME.
<code>init.mpe</code>	initial value of the MPE.
<code>init.mape</code>	initial value of the MAPE.
<code>trend</code>	character string indicating the fitted trend.
<code>ctrend</code>	character string indicating the mathematical expression of the fitted trend.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislosti s příklady v R.
 Jindřichův Hradec: Fakulta managementu VŠE.

See Also

[plot.tsfit](#), [resplot.tsfit](#), [ts.fit.trends](#), [white.noise.test](#), [ts](#).

Examples

```
data(tsCPI)
fit <- ts.trend(tsCPI, trend="modified.exponential")

print(fit)
predict(fit, step=1:5)

### Plot of the time series with fitted values and
### plot of residuals
oldPar <- par(mfrow=c(1, 2), bty="n")
plot(tsCPI, xlab="Year", ylab="CPI")
lines(as.numeric(time(tsCPI)), fitted(fit), col="red")
plot(as.numeric(time(tsCPI)), residuals(fit),
     type="l", xlab="Year", ylab="Residual", col="darkgreen")
par(oldPar)

### plot method for the fitted model
plot(fit, add.point.fit=FALSE, xlab="Year", ylab="CPI")

### resplot method for the fitted model
resplot(fit, xlab="Year", ylab="Residual", col="darkgreen", bty="n")
```

tsAccident

Accidental deaths in the U.S.

Description

A seasonal time series with monthly observations of accidental deaths in the U.S. from January 1973 till June 1979.

Usage

```
data(tsAccident)
```

Format

An object of class `ts`.

Source

Brockwell, P. J. and Davis, R. A. (1991). *Time Series: Theory and Methods*. New York: Springer.

References

Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S, 4th Edition*. New York: Springer.

See Also

`accdeaths`.

Examples

```
data(tsAccident)
summary(tsAccident)
```

`tsAirpass`*International airline passangers data*

Description

A seasonal time series with mothly observations of numbers (times 1000) of international airline passangers carried in January 1949-December 1960.

Usage

```
data(tsAirpass)
```

Format

An object of class `ts`.

Source

Box, G. E. P. and Jenkins, G. M. (1976). *Time Series Analysis: Forecasting and Control, Revised Edition*. San Francisco: Holden-Day.

References

Brockwell, P. J. and Davis, R. A. (1991). *Time Series: Theory and Methods, Second Edition*. Sec. 9.2. New York: Springer.

Examples

```
data(tsAirpass)
summary(tsAirpass)
```

tsCPI	<i>Consumer price index</i>
-------	-----------------------------

Description

This is a part of the Nelson-Plosser macroeconomic time series data giving the U.S. consumer price index from 1860-1988.

Usage

```
data(tsCPI)
```

Format

A time series (object of class `ts`) with yearly observations of the U.S. consumer price index.

Source

Journal of Business and Economic Statistics data archive.
<http://www.amstat.org/publications/jbes/>

References

Koop, G. and Steel, M. F. J. (1994). A decision-theoretic analysis of the unit-root hypothesis using mixtures of elliptical models. *Journal of Business and Economic Statistics*, **12**, 95-107.

See Also

`cpi`.

Examples

```
data(tsCPI)
summary(tsCPI)
```

`tsGNPdef`*GNP deflator*

Description

This is a part of the Nelson-Plosser macroeconomic time series data giving the U.S. GNP deflator from 1889-1988.

Usage

```
data(tsGNPdef)
```

Format

A time series (object of class `ts`) with yearly observations of the U.S. GNP deflator.

Source

Journal of Business and Economic Statistics data archive.
<http://www.amstat.org/publications/jbes/>

References

Koop, G. and Steel, M. F. J. (1994). A decision-theoretic analysis of the unit-root hypothesis using mixtures of elliptical models. *Journal of Business and Economic Statistics*, **12**, 95-107.

See Also

[gnp.def](#).

Examples

```
data(tsGNPdef)
summary(tsGNPdef)
```

`tukey.additivity.test`*Tukey one degree of freedom test of additivity*

Description

Performs Tukey one degree of freedom test.

Usage

```
tukey.additivity.test(x, data)

## S3 method for class 'tukey.additivity.test':
print(x, ...)
```

Arguments

`x` an object of class `avov` or `lm`.
`data` data frame which was used by functions `avov` or `lm` to produce `x`.
`...` further arguments to be passed to or from methods.

Details

Performs Tukey one degree of freedom test, i.e. the test for presence of a multiplicative interaction in the two-way ANOVA model **without** replications. That is, the model is

$$Y_{i,j} = \mu + \alpha_i + \beta_j + D\alpha_i\beta_j + \varepsilon_{i,j} \quad (i = 1, \dots, I, j = 1, \dots, J)$$

with $\varepsilon_{i,j} \sim N(0, \sigma^2)$. We test the null hypothesis $H_0 : D = 0$ against the alternative $H_1 : D \neq 0$.

See Tukey (1949) or Neter et al. (1996, Sec. 21.2) or Komárek a Komárková (2007, Sec. 9.4) for details.

Value

A list with class `tukey.additivity.test` containing the following components:

<code>F</code>	the value of the test statistic.
<code>num.df</code>	numerator degrees of freedom for the F-distribution of the test statistic under the null hypothesis (it is always equal to 1).
<code>den.df</code>	denominator degrees of freedom for the F-distribution of the test statistic under the null hypothesis
<code>p.value</code>	the P-value for the test.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

- Komárek, A. a Komárková, L. (2007). *Statistická analýza závislosti s příklady v R*. Jindřichův Hradec: Fakulta managementu VŠE.
- Neter, J., Kutner, M. H., Nachtsheim, C. J., and Wasserman, W. (1996). *Applied Linear Statistical Models, 4th Edition*. Boston: WCB/McGraw-Hill.
- Tukey, J. W. (1949). One degree of freedom for non-additivity. *Biometrics*, **5**, 232-242.

Examples

```

### Example from Neter, Kutner, Nachtsheim, Wasserman (1996), p. 884
premium <- data.frame(y=c(140, 210, 220, 100, 180, 200),
                      bl=gl(3, 1, 6, labels=c("0small", "1medium", "2large")),
                      reg=gl(2, 3, 6, labels=c("east", "west")))
print(premium)

## All the following produces the same results:
modell1 <- aov(y ~ reg + bl, data=premium)
tukey.additivity.test(modell1, data=premium)
#
modell2 <- lm(y ~ reg + bl, data=premium)
tukey.additivity.test(modell2, data=premium)
#
modell3 <- aov(y ~ bl + reg, data=premium)
tukey.additivity.test(modell3, data=premium)
#
modell4 <- lm(y ~ bl + reg, data=premium)
tukey.additivity.test(modell4, data=premium)

```

white.noise.test *Nonparametric tests for a white noise*

Description

For a vector of residuals e_1, \dots, e_n , it tests the null hypothesis of a white noise using three non-parametric tests.

Usage

```

white.noise.test(residuals)

## S3 method for class 'white.noise.test':
print(x, ...)

```

Arguments

residuals	a vector of residuals for which we want to test the hypothesis of a white noise.
x	an object with the class <code>white.noise.test</code> .
...	further arguments to be passed to or from methods.

Details

For a vector of residuals e_1, \dots, e_n , it tests the null hypothesis of a white noise using three non-parametric tests:

Turning point test

The standardized test statistic has the form

$$T_P = \frac{P - \frac{2(n-2)}{3}}{\sqrt{\frac{16n-29}{90}}},$$

where P is the number of turning points. The statistic T_P follows asymptotically under the null hypothesis standard normal distribution.

Difference-sign test

The standardized test statistic has the form

$$T_S = \frac{S^+ - \frac{n-1}{2}}{\sqrt{\frac{n+1}{12}}}$$

where S^+ is the number of positive differences. The statistic T_S follows asymptotically under the null hypothesis standard normal distribution.

Rank test

The standardized test statistic has the form

$$T_V = \frac{V - \frac{n(n-1)}{4}}{\sqrt{\frac{n(n-1)(2n+5)}{8}}},$$

where V is the number of couples (e_s, e_t) , where $e_s < e_t, s < t$. The statistic T_V follows asymptotically under the null hypothesis standard normal distribution.

See Komárek a Komárková (2007, Sec. 13.1) for details.

Value

A list with the class `white.noise.test` having the following components:

<code>result</code>	A data frame having the following columns: Statistic values of (unstandardized) test statistics. Mu.Stat means of the test statistics under the null hypothesis. Var.Stat variances of the test statistics under the null hypothesis. Std.Stat values of standardized test statistics. p.value P-values for the tests.
<code>names</code>	a vector of character strings identifying the tests performed.

Author(s)

Arnošt Komárek <komarek@karlin.mff.cuni.cz>

References

Komárek, A. a Komárková, L. (2007).
Statistická analýza závislostí s příklady v R.
Jindřichův Hradec: Fakulta managementu VŠE.

Examples

```
### Example 1a
data(tsCPI)
fit <- ts.trend(tsCPI, trend="modified.exponential")
white.noise.test(residuals(fit))

### Example 1b
fit7 <- ts.masmooth(tsCPI, length=7)
white.noise.test(residuals(fit7))

### Example 1c
fit7c <- ts.masmooth(tsCPI, length=6)
white.noise.test(residuals(fit7c))

### Example 2
data(tsAccident)
fitAc3 <- ts.simpleseason.trend(tsAccident,
  decomposition="additive", trend="quadratic")
white.noise.test(residuals(fitAc3))

### Example 3
data(tsAirpass)
fitAirp3 <- ts.simpleseason.trend(tsAirpass,
  decomposition="multiplicative", trend="gompertz")
white.noise.test(residuals(fitAirp3))
```

Index

*Topic **classes**

num2string, 19

*Topic **datasets**

cars04, 5

tsAccident, 59

tsAirpass, 60

tsCPI, 61

tsGNPdef, 62

*Topic **design**

tukey.additivity.test, 62

*Topic **distribution**

rdiscrete, 35

*Topic **dplot**

ts.explore, 46

*Topic **hplot**

cube, 6

plot.tsfit, 28

resplot, 39

ts.explore, 46

*Topic **htest**

asympt.mean.test, 1

bhapkar.test, 3

estim.mean, 7

estim.var, 9

levene.var.test, 11

lmbeta.test, 12

lr.gof.test, 14

lr.indep.test, 16

odds.ratio.test, 19

onesample.var.test, 22

pearson.indep.test, 24

pearson.test, 26

prop.diff.test, 30

prop.Z.test, 33

rel.risk.test, 36

stuart.test, 43

symmetry.test, 44

tukey.additivity.test, 62

white.noise.test, 64

*Topic **models**

lmbeta.test, 12

masmooth.coef, 17

plot.tsfit, 28

resplot, 39

rsemistudent, 41

ts.fit.trends, 47

ts.masmooth, 50

ts.simpleseason.trend, 52

ts.trend, 57

*Topic **regression**

lmbeta.test, 12

resplot, 39

rsemistudent, 41

*Topic **ts**

masmooth.coef, 17

plot.tsfit, 28

resplot, 39

ts.fit.trends, 47

ts.masmooth, 50

ts.simpleseason.trend, 52

ts.trend, 57

accdeaths, 60

aov, 63

arrows, 6

asympt.mean.test, 1, 9

bartlett.test, 12

bhapkar.test, 3, 44

cars04, 5

chisq.test, 17, 25

cpi, 61

cube, 6

cubeA (cube), 6

cubeB (cube), 6

cubeC (cube), 6

cubeY (cube), 6

estim.mean, 3, 7

- estim.var, 9
- fitted.ts.masmooth (*ts.masmooth*), 50
- fitted.ts.simpleseason.trend (*ts.simpleseason.trend*), 52
- fitted.ts.trend (*ts.trend*), 57
- gnp.def, 62
- levene.var.test, 11
- lm, 13, 39, 40, 42, 63
- lmbeta.test, 12
- lowess, 40, 46
- lr.gof.test, 14, 27
- lr.indep.test, 16, 25
- masmooth.coef, 17, 51
- mcnemar.test, 4, 45
- model.frame, 11
- num2string, 19
- odds.ratio.test, 19, 32, 38
- onesample.var.test, 10, 22, 24
- pearson.indep.test, 17, 24
- pearson.test, 15, 26
- plot, 29, 40, 46, 48
- plot.ts.fit.trends (*ts.fit.trends*), 47
- plot.tsfit, 28, 51, 55, 59
- predict.ts.masmooth (*ts.masmooth*), 50
- predict.ts.simpleseason.trend (*ts.simpleseason.trend*), 52
- predict.ts.trend (*ts.trend*), 57
- print.asymp.mean.test (*asymp.mean.test*), 1
- print.bhpkar.test (*bhpkar.test*), 3
- print.estim.mean (*estim.mean*), 7
- print.estim.var (*estim.var*), 9
- print.lr.gof.test (*lr.gof.test*), 14
- print.lr.indep.test (*lr.indep.test*), 16
- print.odds.ratio.test (*odds.ratio.test*), 19
- print.onesample.var.test (*onesample.var.test*), 22
- print.pearson.indep.test (*pearson.indep.test*), 24
- print.pearson.test (*pearson.test*), 26
- print.prop.diff.test (*prop.diff.test*), 30
- print.prop.Z.test (*prop.Z.test*), 33
- print.rel.risk.test (*rel.risk.test*), 36
- print.stuart.test (*stuart.test*), 43
- print.symmetry.test (*symmetry.test*), 44
- print.ts.fit.trends (*ts.fit.trends*), 47
- print.ts.masmooth (*ts.masmooth*), 50
- print.ts.simpleseason.trend (*ts.simpleseason.trend*), 52
- print.ts.trend (*ts.trend*), 57
- print.tukey.additivity.test (*tukey.additivity.test*), 62
- print.white.noise.test (*white.noise.test*), 64
- prop.diff.test, 21, 30, 38
- prop.test, 21, 32, 35, 38
- prop.Z.test, 33
- rdiscrete, 35
- rel.risk.test, 21, 32, 36
- residuals.lm, 39, 40
- residuals.ts.masmooth, 40
- residuals.ts.masmooth (*ts.masmooth*), 50
- residuals.ts.simpleseason.trend, 40
- residuals.ts.simpleseason.trend (*ts.simpleseason.trend*), 52
- residuals.ts.trend, 40
- residuals.ts.trend (*ts.trend*), 57
- resplot, 39
- resplot.tsfit, 51, 55, 59
- rsemistudent, 41
- rstandard, 42
- rstandard.lm, 39, 40
- rstudent, 42

sample, 36
stuart.test, 4, 43
summary.lm, 14
symmetry.test, 44

t.test, 3, 9
ts, 29, 46–51, 53, 55, 57, 59–62
ts.explore, 46
ts.fit.trends, 47, 47, 55, 59
ts.masmooth, 18, 28, 29, 40, 50, 55
ts.simpleseason.trend, 28, 29, 40, 52
ts.trend, 28, 29, 40, 47, 49, 55, 57
tsAccident, 59
tsAirpass, 60
tsCPI, 61
tsGNPdef, 62
tukey.additivity.test, 62

white.noise.test, 51, 55, 59, 64