Package ‘SpeTestNP’

January 20, 2025

Type Package

Title Non-Parametric Tests of Parametric Specifications

Version 1.1.0

Date 2022-10-24

Depends stats, foreach, parallel, doParallel

Suggests knitr, rmarkdown, AER

Maintainer Hippolyte Boucher <Hippolyte.Boucher@outlook.com>

Author Hippolyte Boucher [aut, cre],
Pascal Lavergne [aut]

Description Performs non-parametric tests of parametric specifications.
Five tests are available.
Specific bandwidth and kernel methods can be chosen along with many other options.
Allows parallel computing to quickly compute p-values based on the bootstrap.
Methods implemented in the package are H.J. Bierens (1982) <doi:10.1016/0304-4076(82)90105-1>,
J.C. Escanciano (2006) <doi:10.1017/S0266466606060506>,
P.L. Gozalo (1997) <doi:10.1016/S0304-4076(97)86571-2>,
P. Lavergne and V. Patilea (2008) <doi:10.1016/j.jeconom.2007.08.014>,
P. Lavergne and V. Patilea (2012) <doi:10.1198/jbes.2011.07152>,
J.H. Stock and M.W. Watson (2006) <doi:10.1111/j.1538-4616.2007.00014.x>,
C.F.J. Wu (1986) <doi:10.1214/aos/1176350142>,
J. Yin, Z. Geng, R. Li, H. Wang (2010) <https://www.jstor.org/stable/24309002>
and J.X. Zheng (1996) <doi:10.1016/0304-4076(95)01760-7>.

Language en-US

VignetteBuilder knitr, rmarkdown

NeedsCompilation yes

Encoding UTF-8

URL https://github.com/HippolyteBoucher/SpeTestNP

Repository CRAN

Date/Publication 2022-10-28 17:10:15 UTC

Contents

`print.STNP` *Print a specification test* `STNP` *object*

Description

Prints the test statistic and p-value of a specification test object of class `STNP`

Usage

```
## S3 method for class 'STNP'
print(x, ...)
```

Arguments

`x`	An object of class `STNP` resulting from function `SpeTest`
`...`	Additional print arguments

Value

No return value, prints the test statistic and p-value

Author(s)

Hippolyte Boucher <Hippolyte.Boucher@outlook.com>

Pascal Lavergne <lavergnetse@gmail.com>

See Also

`SpeTest` is the function which performs a specification test and records it along with all its options in an object of class `STNP`

`summary.STNP` prints a summary of the specification test with all the options used

Examples

```
n <- 100
k <- 2
x <- matrix(rnorm(n*k),ncol=k)
y<-1+x%*%(1:k)+rnorm(n)

eq<-lm(y~x+0)


print(SpeTest(eq=eq,type="icm",nboot=50))
```

`SpeTest` *Nonparametric specification test*

Description

`SpeTest` tests a parametric specification. It returns the test statistic and its p-value for five different heteroskedasticity-robust nonparametric specification tests

Usage

```
SpeTest(eq, type="icm", rejection="bootstrap", norma="no", boot="wild",
nboot=50, para=FALSE, ker="normal",knorm="sd", cch="default", hv="default",
nbeta="default", direct="default", alphan="default")
```

Arguments

`eq` A fitted model of class `lm` or `nls`

`type` Test type

If `type` = `"icm"` the test of Bierens (1982) is performed (default)

If `type` = `"zheng"` the test of Zheng (1996) is performed

If `type` = `"esca"` the test of Escanciano (2006) is performed, significantly increases computing time

If `type` = `"pala"` the test of Lavergne and Patilea (2008) is performed

If `type` = `"sicm"` the test of Lavergne and Patilea (2012) is performed

`rejection` Rejection rule

If `rejection` = `"bootstrap"` the p-value of the test is based on the bootstrap (default)

If `rejection` = `"asymptotics"` and `type` = `"zheng"` or `type` = `"esca"` or `type` = `"sicm"` the p-value of the test is based on asymptotic normality of the normalized test statistic under the null hypothesis

If `type` = `"icm"` or `type` = `"esca"` the argument `rejection` is ignored and the p-value is based on the bootstrap

norma Normalization of the test statistic

If `norma = "no"` the test statistic is not normalized (default)

If `norma = "naive"` the test statistic is normalized with a naive estimator of the variance of its components

If `norma = "np"` the test statistic is normalized with a nonparametric estimator of the variance of its components

boot Bootstrap method to compute the test p-value

If `boot = "wild"` the wild bootstrap of Wu (1986) is used (default)

If `boot = "smooth"` the smooth conditional moments bootstrap of Gozalo (1997) is used

nboot Number of bootstraps used to compute the test p-value, by default `nboot = 50`

para Parallel computing

If `para = FALSE` parallel computing is not used to generate the bootstrap samples to compute the test p-value (default)

If `para = TRUE` parallel computing is used to generate the bootstrap samples to compute the test p-value, significantly decreases computing time, makes use of all CPU cores except one

ker Kernel function used in the central matrix and for the nonparametric covariance estimator

If `ker = "normal"` the central matrix kernel function is the normal p.d.f (default)

If `ker = "triangle"` the central matrix kernel function is the triangular p.d.f

If `ker = "logistic"` the central matrix kernel function is the logistic p.d.f

If `ker = "sinc"` the central matrix kernel function is the sine cardinal function

knorm Normalization of the kernel function

If `knorm = "sd"` then the standard deviation using the kernel function equals 1 (default)

If `knorm ="sq"` then the integral of the squared kernel function equals 1

cch Central matrix kernel bandwidth

If `type = "icm"` or `type = "esca"` then `cch` always equals `1`

If `type = "zheng"` the `"default"` bandwidth is the scaled rule of thumb: `cch = 1.06*n^(-1/(4+k))` where `k` is the number of regressors

If `type = "sicm"` or `type = "pala"` the `"default"` bandwidth is the scaled rule of thumb: `cch = 1.06*n^(-1/5)`

The user may change the bandwidth when `type = "zheng"`, `type = "sicm"` or `type = "pala"`.

hv If `norma = "np"` or `rejection = "bootstrap"` and `boot = "smooth"`, `hv` is the bandwidth of the nonparametric errors covariance estimator, by `"default"` the bandwidth is the scaled rule of thumb `hv = 1.06*n^(-1/(4+k))`

nbeta If `type = "pala"` or `type = "sicm"`, `nbeta` is the number of "betas" in the unit hypersphere used to compute the statistic, computing time increases as `nbeta` gets larger

By `"default"` it is equal to 20 times the square root of the number of exogenous control variables

direct	If type = "pala", direct is the favored direction for beta, by "default" it is the OLS estimator if class(eq) = "lm" If type = "sicm", direct is the initial direction for beta. This direction should be a vector of 0 (for no direction), 1 (for positive direction) and -1 (for negative direction) For ex, c(1,-1,0) indicates that the user thinks that the first regressor has a positive effect on the dependent variable, that the second regressor has a negative effect on the dependent variable, and that he has no idea about the effect of the third regressor By "default" no direction is given to the hypersphere
alphan	If type = "pala", alphan is the weight given to the favored direction for beta, by "default" it is equal to log(n)*n^(-3/2)

Details

To perform a nonparametric specification test the only argument needed is a model eq of class lm or of class nls. But other options can and should be specified: the test type type, the rejection rule rejection, the normalization of the test statistic norm, the bootstrap type boot and the size of the vector being generated which is equal to the number of bootstrap samples nboot, whether the vector is generated using parallel computing para, the central matrix kernel function ker and its standardization ker, the bandwidths cch and hv. If the user has knowledge of the tests coined by Lavergne and Patilea he may choose a higher number of betas for the hypersphere (which may significantly increase computational time) and an initial "direction" to the hypersphere for the SICM test (none is given by "default") or a starting beta for the PALA test (which is the OLS estimator by "default" if class(eq) = "nls").

The statistic can be normalized with a naive estimator of the conditional covariance of its elements as in Zheng (1996), or with a nonparametric estimator of the conditional covariance of its elements as in in Yin, Geng, Li, Wang (2010). The p-value is based either on the wild bootstrap of Wu (1986) or on the smooth conditional moments bootstrap of Gozalo (1997).

Value

SpeTest returns an object of class STNP.

summary and print can be used on objects of this class.

An object of class STNP is a list which contains the following elements:

stat	The value of the test statistic used in the test
pval	The test p-value
type	The type of test which was used
boot	The type of bootstrap which was used to compute the p-value
nboot	The number of bootstrap samples used to compute the p-value
ker	The central matrix kernel function which was used
knorm	The kernel matrix standardization: "sq" if the second moment equals 1 or "sd" if the standard deviation equals 1
cch	The central matrix kernel function bandwidth

hv	The nonparametric covariance estimator bandwidth
nbeta	The number of directions in the unit hypersphere used to compute the test statistic if type = "pala" or type = "sicm"
direct	The preferred / initial direction in the unit hypersphere if type = "pala" or type = "sicm"
alphan	The weight given to the preferred direction if type = "pala"

Note

The data used to obtain the fitted model eq should not contain factors, factor variables should be transformed into dummy variables a priori

Requires the packages stats (already installed and loaded by default in Rstudio), foreach, parallel and doParallel (if parallel computing is used to generate the test p-value) to be installed

For more information and to be able to use the package to its full potential see the references

Author(s)

Hippolyte Boucher <Hippolyte.Boucher@outlook.com>

Pascal Lavergne <lavergnetse@gmail.com>

References

H.J. Bierens (1982), "Consistent Model Specification Test", *Journal of Econometrics*, 20 (1), 105-134

J.C. Escanciano (2006), "A Consistent Diagnostic Test for Regression Models using Projections", *Economic Theory*, 22 (6), 1030-1051

P.L. Gozalo (1997), "Nonparametric Bootstrap Analysis with Applications to Demographic Effects in Demand Functions", *Journal of Econometrics*, 81 (2), 357-393

P. Lavergne and V. Patilea (2008), "Breaking the Curse of Dimensionality in Nonparametric Testing", *Journal of Econometrics*, 143 (1), 103-122

P. Lavergne and V. Patilea (2012), "One for All and All for One: Regression Checks with Many Regressors", *Journal of Business and Economic Statistics*, 30 (1), 41-52

C.F.J. Wu (1986), "Jackknife, bootstrap and other resampling methods in regression analysis (with discussion)", *Annals of Statistics*, 14 (4), 1261-1350

J. Yin, Z. Geng, R. Li, H. Wang (2010), "Nonparametric covariance model", *Statistica Sinica*, 20 (1), 469-479

J.X. Zheng (1996), "A Consistent Test of Functional Form via Nonparametric Estimation Techniques", *Journal of Econometrics*, 75 (2), 263-289

See Also

print and print.STNP applied to an object of class STNP print the specification test statistic and its p-value

summary and summary.STNP applied to an object of class STNP print a summary of the specification test with all the options used

`SpeTest_Stat` is the function which only returns the specification test statistic

`SpeTest_Dist` generates a vector drawn from the distribution of the test statistic under the null hypothesis using the bootstrap

Examples

```
n <- 100
k <- 2
x <- matrix(rnorm(n*k),ncol=k)
y<-1+x%*%(1:k)+rnorm(n)

eq<-lm(y~x+0)

summary(SpeTest(eq=eq,type="icm",norma="naive",boot="smooth"))

eq<-nls(out~expla1*a+b*expla2+c,start=list(a=0,b=4,c=2),
data=data.frame(out=y,expla1=x[,1],expla2=x[,2]))

print(SpeTest(eq=eq,type="icm",norma="naive",boot="smooth"))
```

`SpeTest_Dist` *Nonparametric specification test statistic distribution*

Description

`SpeTest_Dist` generates a vector from the nonparametric specification test statistic distribution under the null hypothesis for one of five different tests using the bootstrap

Usage

```
SpeTest_Dist(eq, type="icm", norma="no", boot="wild", nboot=50, para=FALSE, ker="normal",
knorm="sd", cch="default", hv="default", nbeta="default", direct="default",
alphan="default")
```

Arguments

`eq` A fitted model of class `lm` or `nls`

`type` Test type

If `type = "icm"` the vector is generated from the distribution of the test of Bierens (1982) under the null hypothesis (default)

If `type = "zheng"` the vector is generated from the distribution of the test of Zheng (1996) under the null hypothesis

If `type = "esca"` the vector is generated from the distribution of the test of Escanciano (2006) under the null hypothesis, significantly increases computing time

If `type = "pala"` the vector is generated from the distribution of the test of Lavergne and Patilea (2008) under the null hypothesis

If type = "sicm" the vector is generated from the distribution of the test of Lavergne and Patilea (2012) under the null hypothesis

norma
: Normalization of the test statistic

  If norma = "no" the test statistic is not normalized (default)

  If norma = "naive" the test statistic is normalized with a naive estimator of the variance of its components

  If norma = "np" the test statistic is normalized with a nonparametric estimator of the variance of its components

boot
: Bootstrap type to generate the vector drawn from the distribution under the null hypothesis of the test statistic

  If boot = "wild" the wild bootstrap of Wu (1986) is used

  If boot = "smooth" the smooth conditional moments bootstrap of Gozalo (1997) is used

nboot
: Size of the vector drawn from the test statistic distribution under the null using the bootstrap, by default nboot = 50

para
: Parallel computing

  If para = FALSE parallel computing is not used to generate the vector from the test statistic distribution under the null (default)

  If para = TRUE parallel computing is used to generate the vector from the test statistic distribution under the null, significantly decreases computing time, makes use of all CPU cores except one

ker
: Kernel function used in the central matrix and for the nonparametric covariance estimator

  If ker = "normal" the central matrix kernel function is the normal p.d.f (default)

  If ker = "triangle" the central matrix kernel function is the triangular p.d.f

  If ker = "logistic" the central matrix kernel function is the logistic p.d.f

  If ker = "sinc" the central matrix kernel function is the sine cardinal function

knorm
: Normalization of the kernel function

  If knorm = "sd" then the standard deviation using the kernel function equals 1 (default)

  If knorm ="sq" then the integral of the squared kernel function equals 1

cch
: Central matrix kernel bandwidth

  If type = "icm" or type = "esca" then cch always equals 1

  If type = "zheng" the "default" bandwidth is the scaled rule of thumb: cch = 1.06*n^(-1/(4+k)) where k is the number of regressors

  If type = "sicm" or type = "pala" the "default" bandwidth is the scaled rule of thumb: cch = 1.06*n^(-1/5)

  The user may change the bandwidth when type = "zheng", type = "sicm" or type = "pala"

hv
: If norma = "np" or rejection = "bootstrap" and boot = "smooth", hv is the bandwidth of the nonparametric errors covariance estimator, by "default" the bandwidth is the scaled rule of thumb hv = 1.06*n^(-1/(4+k))

nbeta	If type = "pala" or type = "sicm", nbeta is the number of "betas" in the unit hypersphere used to compute the statistic, computing time increases as nbeta gets larger By "default" it is equal to 20 times the square root of the number of exogenous control variables
direct	If type = "pala", direct is the favored direction for beta, by "default" it is the OLS estimator if class(eq) = "lm" If type = "sicm", direct is the initial direction for beta. This direction should be a vector of 0 (for no direction), 1 (for positive direction) and -1 (for negative direction) For ex, c(1,-1,0) indicates that the user thinks that the first regressor has a positive effect on the dependent variable, that the second regressor has a negative effect on the dependent variable, and that he has no idea about the effect of the third regressor By "default" no direction is given to the hypersphere
alphan	If type = "pala", alphan is the weight given to the favored direction for beta, by "default" it is equal to log(n)*n^(-3/2)

Details

To generate a vector from the specification test statistic distribution under the null using the bootstrap the only argument needed is a model eq of class lm or of class nls. But other options can and should be specified: the test statistic type type, the normalization of the test statistic norma, the bootstrap type boot and the size of the vector being generated which is equal to the number of bootstrap samples nboot, whether the vector is generated using parallel computing para, the central matrix kernel function ker and its standardization knorm, the bandwidths cch and hv. If the user has knowledge of the tests coined by Lavergne and Patilea he may choose a higher number of betas for the hypersphere (which may significantly increase computational time) and an initial "direction" to the hypersphere for the SICM test (none is given by "default") or a starting beta for the PALA test (which is the OLS estimator by "default" if class(eq) = "nls").

The statistic can be normalized with a naive estimator of the conditional covariance of its elements as in Zheng (1996), or with a nonparametric estimator of the conditional covariance of its elements as in in Yin, Geng, Li, Wang (2010). The vector is generated either from the wild bootstrap of Wu (1986) or from the smooth conditional moments bootstrap of Gozalo (1997).

Value

SpeTest_Dist returns a vector of size nboot which is drawn from the distribution of the test statistic under the null hypothesis using the bootstrap

Note

The data used to obtain the fitted model eq should not contain factors, factor variables should be transformed into dummy variables a priori

Requires the packages stats (already installed and loaded by default in Rstudio), foreach, parallel and doParallel (if parallel computing is used to generate the vector) to be installed

For more information and to be able to use the package to its full potential see the references

Author(s)

Hippolyte Boucher <Hippolyte.Boucher@outlook.com>

Pascal Lavergne <lavergnetse@gmail.com>

References

H.J. Bierens (1982), "Consistent Model Specification Test", *Journal of Econometrics*, 20 (1), 105-134

J.C. Escanciano (2006), "A Consistent Diagnostic Test for Regression Models using Projections", *Economic Theory*, 22 (6), 1030-1051

P.L. Gozalo (1997), "Nonparametric Bootstrap Analysis with Applications to Demographic Effects in Demand Functions", *Journal of Econometrics*, 81 (2), 357-393

P. Lavergne and V. Patilea (2008), "Breaking the Curse of Dimensionality in Nonparametric Testing", *Journal of Econometrics*, 143 (1), 103-122

P. Lavergne and V. Patilea (2012), "One for All and All for One: Regression Checks with Many Regressors", *Journal of Business and Economic Statistics*, 30 (1), 41-52

C.F.J. Wu (1986), "Jackknife, bootstrap and other resampling methods in regression analysis (with discussion)", *Annals of Statistics*, 14 (4), 1261-1350

J. Yin, Z. Geng, R. Li, H. Wang (2010), "Nonparametric covariance model", *Statistica Sinica*, 20 (1), 469-479

J.X. Zheng (1996), "A Consistent Test of Functional Form via Nonparametric Estimation Techniques", *Journal of Econometrics*, 75 (2), 263-289

See Also

`SpeTest` is the function which performs a specification test and records it along with all its options in an object of class `STNP`

`SpeTest_Stat` is the function which only returns the specification test statistic

Examples

```
n <- 100
k <- 2
x <- matrix(rnorm(n*k),ncol=k)
y<-1+x%*%(1:k)+rnorm(n)

eq<-lm(y~x+0)

SpeTest_Dist(eq=eq,type="zheng",boot="wild",nboot=10)

eq<-nls(out~expla1*a+b*expla2+c,start=list(a=0,b=4,c=2),
data=data.frame(out=y,expla1=x[,1],expla2=x[,2]))

SpeTest_Dist(eq=eq,type="zheng",boot="wild",nboot=20)
```

SpeTest_Stat *Nonparametric specification test statistic*

Description

SpeTest computes the nonparametric specification test statistic for one of five different tests

Usage

```
SpeTest_Stat(eq, type="icm", norma="no", ker="normal", knorm="sd",
cch="default", hv="default", nbeta="default", direct="default",
alphan="default")
```

Arguments

- `eq` — A fitted model of class `lm` or `nls`
- `type` — Test statistic type

  If `type = "icm"` the test statistic of Bierens (1982) is returned (default)

  If `type = "zheng"` the test statistic of Zheng (1996) is returned

  If `type = "esca"` the test statistic of Escanciano (2006) is returned, significantly increases computing time

  If `type = "pala"` the test statistic of Lavergne and Patilea (2008) is returned

  If `type = "sicm"` the test statistic of Lavergne and Patilea (2012) is returned
- `norma` — Normalization of the test statistic

  If `norma = "no"` the test statistic is not normalized (default)

  If `norma = "naive"` the test statistic is normalized with a naive estimator of the variance of its components

  If `norma = "np"` the test statistic is normalized with a nonparametric estimator of the variance of its components
- `ker` — Kernel function used in the central matrix and for the nonparametric covariance estimator

  If `ker = "normal"` the central matrix kernel function is the normal p.d.f (default)

  If `ker = "triangle"` the central matrix kernel function is the triangular p.d.f

  If `ker = "logistic"` the central matrix kernel function is the logistic p.d.f

  If `ker = "sinc"` the central matrix kernel function is the sine cardinal function
- `knorm` — Normalization of the kernel function

  If `knorm = "sd"` then the standard deviation using the kernel function equals 1 (default)

  If `knorm ="sq"` then the integral of the squared kernel function equals 1
- `cch` — Central matrix kernel bandwidth

  If `type = "icm"` or `type = "esca"` then `cch` always equals `1`

  If `type = "zheng"` the `"default"` bandwidth is the scaled rule of thumb: `cch = 1.06*n^(-1/(4+k))` where `k` is the number of regressors

If type = "sicm" or type = "pala" the "default" bandwidth is the scaled rule of thumb: cch = 1.06*n^(-1/5)

The user may change the bandwidth when type = "zheng", type = "sicm" or type = "pala"

hv — If norma = "np" or rejection = "bootstrap" and boot = "smooth", hv is the bandwidth of the nonparametric errors covariance estimator, by "default" the bandwidth is the scaled rule of thumb hv = 1.06*n^(-1/(4+k))

nbeta — If type = "pala" or type = "sicm", nbeta is the number of "betas" in the unit hypersphere used to compute the statistic, computing time increases as nbeta gets larger

By "default" it is equal to 20 times the square root of the number of exogenous control variables

direct — If type = "pala", direct is the favored direction for beta, by "default" it is the OLS estimator if class(eq) = "lm"

If type = "sicm", direct is the initial direction for beta. This direction should be a vector of 0 (for no direction), 1 (for positive direction) and -1 (for negative direction)

For ex, c(1,-1,0) indicates that the user thinks that the first regressor has a positive effect on the dependent variable, that the second regressor has a negative effect on the dependent variable, and that he has no idea about the effect of the third regressor

By "default" no direction is given to the hypersphere

alphan — If type = "pala", alphan is the weight given to the favored direction for beta, by "default" it is equal to log(n)*n^(-3/2)

Details

To compute the specification test statistic the only argument needed is a model eq of class lm or of class nls. But other options can and should be specified: the test statistic type type, the normalization of the test statistic norma, the central matrix kernel function ker and its standardization ker, the bandwidths cch and hv. If the user has knowledge of the tests coined by Lavergne and Patilea he may choose a higher number of betas for the hypersphere (which may significantly increase computational time) and an initial "direction" to the hypersphere for the SICM test (none is given by "default") or a starting beta for the PALA test (which is the OLS estimator by "default" if class(eq) = "nls").

The statistic can be normalized with a naive estimator of the conditional covariance of its elements as in Zheng (1996), or with a nonparametric estimator of the conditional covariance of its elements as in in Yin, Geng, Li, Wang (2010).

Value

SpeTest_Stat returns the nonparametric specification test statistic

Note

The data used to obtain the fitted model eq should not contain factors, factor variables should be transformed into dummy variables a priori

Requires the packages `stats` (already installed and loaded by default in Rstudio), `foreach`, `parallel` and `doParallel` (if parallel computing is used to generate the vector) to be installed

For more information and to be able to use the package to its full potential see the references

Author(s)

Hippolyte Boucher <Hippolyte.Boucher@outlook.com>

Pascal Lavergne <lavergnetse@gmail.com>

References

H.J. Bierens (1982), "Consistent Model Specification Test", *Journal of Econometrics*, 20 (1), 105-134

J.C. Escanciano (2006), "A Consistent Diagnostic Test for Regression Models using Projections", *Economic Theory*, 22 (6), 1030-1051

P.L. Gozalo (1997), "Nonparametric Bootstrap Analysis with Applications to Demographic Effects in Demand Functions", *Journal of Econometrics*, 81 (2), 357-393

P. Lavergne and V. Patilea (2008), "Breaking the Curse of Dimensionality in Nonparametric Testing", *Journal of Econometrics*, 143 (1), 103-122

P. Lavergne and V. Patilea (2012), "One for All and All for One: Regression Checks with Many Regressors", *Journal of Business and Economic Statistics*, 30 (1), 41-52

C.F.J. Wu (1986), "Jackknife, bootstrap and other resampling methods in regression analysis (with discussion)", *Annals of Statistics*, 14 (4), 1261-1350

J. Yin, Z. Geng, R. Li, H. Wang (2010), "Nonparametric covariance model", *Statistica Sinica*, 20 (1), 469-479

J.X. Zheng (1996), "A Consistent Test of Functional Form via Nonparametric Estimation Techniques", *Journal of Econometrics*, 75 (2), 263-289

See Also

`SpeTest` is the function which performs a specification test and records it along with all its options in an object of class `STNP`

`SpeTest_Dist` generates a vector drawn from the distribution of the test statistic under the null hypothesis using the bootstrap

Examples

```
n <- 100
k <- 2
x <- matrix(rnorm(n*k),ncol=k)
y<-1+x%*%(1:k)+rnorm(n)

eq<-lm(y~x+0)

SpeTest_Stat(eq=eq,type="icm")

eq<-nls(out~expla1*a+b*expla2+c,start=list(a=0,b=4,c=2),
```

```
data=data.frame(out=y,expla1=x[,1],expla2=x[,2]))

SpeTest_Stat(eq=eq,type="icm")
```

`summary.STNP` *Summarize a specification test* `STNP` *object*

Description

Prints a summary of a specification test object of class `STNP` with all the options used, including if options were `"default"`

Usage

```
## S3 method for class 'STNP'
summary(object, ...)
```

Arguments

`object`	An object of class `STNP` resulting from function `SpeTest`
`...`	Additional summary arguments

Value

No return value, prints a summary of the test

Author(s)

Hippolyte Boucher <Hippolyte.Boucher@outlook.com>

Pascal Lavergne <lavergnetse@gmail.com>

See Also

`SpeTest` is the function which performs a specification test and records it along with all its options in an object of class `STNP`

`print.STNP` prints the specification test statistic and p-value only

Examples

```
n <- 100
k <- 2
x <- matrix(rnorm(n*k),ncol=k)
y<-1+x%*%(1:k)+rnorm(n)

eq<-lm(y~x+0)


summary(SpeTest(eq=eq,type="icm",norma="np"))
```

Index