Package: RationalExp (via r-universe)

November 3, 2026

Title Rationalizing Rational Expectations. Tests and Deviations

Version 0.2.2

Description We implement a test of the rational expectations hypothesis based on the marginal distributions of realizations and subjective beliefs from D'Haultfoeuille, Gaillac, and Maurel (2018) <doi:10.3386/w25274>. This test can be used in cases where realizations and subjective beliefs are observed in two different datasets that cannot be matched, or when they are observed in the same dataset. The package also computes the estimator of the minimal deviations from rational expectations than can be rationalized by the data.

Depends R (>= 3.0.0)

License GPL-3

Encoding UTF-8

LazyData true

Suggests knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.1.0

Imports snowfall, stats

NeedsCompilation no

Author Xavier D'Haultfoeuille [aut], Christophe Gaillac [aut, cre], Arnaud Maurel [aut]

Maintainer Christophe Gaillac <christophe.gaillac@ensae.fr>

Repository https://cranhaven.r-universe.dev

Date/Publication 2026-11-03 06:02:00 UTC

RemoteUrl https://github.com/cranhaven/cranhaven.r-universe.dev

RemoteRef package/RationalExp

RemoteSha f69c3182b4a4ff6e3f105b02f401efeb92a4be96

RemoteSubdir RationalExp

Contents

boot_stat 2
c_cube 3
c_fun 4
estimDev 4
inverse 5
S1 6
T_stat 6
test 7
test_base 9
which.min2 10

Index **11**

`boot_stat` *Compute the bootstrap test statistic for parallel implementation*

Description

This is an internal function to separately compute the bootsrap test statsitic.

Usage

```
boot_stat(u, Y_tilde, X, D, epsilon, N3, p, prec, N, sample_mat,
  generalized, weights, y_grid, phi_n, M_bar, DX)
```

Arguments

`u`	bootstrap index;
`Y_tilde`	the vector stacking the realisations y then the anticipated values psi of respective sizes n_y and n_p.
`X`	the matrix of covariates. Set to a vector of 1 by default (in which case the test without covariates is performed).
`D`	the vector stacking the dummies for the dataset of realisation : n_y ones then n_p zeros
`epsilon`	the parameter epsilonon in Section 3 of DGM. Default value is 0.05.
`N3`	equals to N if covariates, to 1 other wise.
`p`	the parameter p in Section 3 of DGM. Default is 0.05.
`prec`	the number of points to be tested. Default is 30.
`N`	the total numeber of obs
`sample_mat`	matrix of bootrap indexes
`generalized`	"Add" if additive shocks for the generalized test
`weights`	survey weights

y_grid	the grid points. Default is quantile(Y_tilde,seq(0,1,length.out=30)).
phi_n	the GMS function in DGM
M_bar	the quantilty bar m in section 2 of DGM
DX	the total number of covariates

Details

By default, the test is implemented without covariates. To perform the test with covariates, one has to indicate in X a non-constant vector or matrix. Also, one can perform the « generalized » tests allowing for aggregate shocks by using the dummy variable generalized. Survey weights can be added. The user can modify the number of cores used by R to reduce the computational time. Tuning parameters used in the test can also be modified.

---

c_cube *Instrumental functions computations*

Description

This function defines, for each specified value of r_n the set of indicator funtions h(X_i) which are the key elements for the RE test with co covariates

Usage

```
c_cube(X_adj, N, DX, r_n)
```

Arguments

X_adj	the standardised version of the covariates X
N	the size of X
DX	the number of covariates
r_n	the parameter indexing the number of instrumental function, which is chosen according the the rule used in AS y default.

Value

a list containing, in order:

- X_adj he standardised version of the covariates X

- r_n the parameter indexing the number of instrumental function, which is chosen according the the rule used in AS y default.

- g_col a vector containing part of the weights

- Q_AR a matrix with the weights that enter the statistic T

- G_X a binary matrix indexing the observations X that fall into the hypercubes indexed by h.

c_fun — *Compute the difference between mean of subvectors of two vectors*

Description

Compute the difference between mean of subvectors of two vectors

Usage

```
c_fun(i, i_t, y, z)
```

Arguments

i	starting index
i_t	final index
y	first vector of elements
z	second vector of elements

Value

a real, the difference between means of subvectors of two vectors

estimDev — *Estimation of the minimal deviations from rational expectations with unconstrained information set g**

Description

This function estimates of the minimal deviations from rational expectations with unconstrained information set. Both vectors should have the same length. If not, one can randomly select a subset of the longer vector with length equal to that of the shorter one. The function returns a function via the approxfun of the package stats. This function can then be evaluated directly on a desired grid.

Usage

```
estimDev(psi, y)
```

Arguments

psi	vector of subjective expectations
y	vector of realisations of an individual outcome.

Examples

```
n_p=200
n_y=200
sig=0.1
u=1
b=0.10
a=2
rho= 0.4
psi <- rnorm(n_p,0,u)
pp_y <- runif(n_y,0,1)
zeta <- rnorm(n_y,a,sig)
zeta1 <- rnorm(n_y,-a,sig)
pp1_y <- 1*(pp_y <b)
pp2_y <- 1*(pp_y >1-b)
pp3_y <- 1*(pp_y <=(1-b) & pp_y >=b)
psi_y <-rnorm(n_p,0,u)
y = rho*psi_y+ pp1_y*zeta + pp2_y*zeta1

g_star <- estimDev(psi,y)
```

---

inverse *Inverse the function f*

Description

This function implements the numerical inverse of the function f.

Usage

```
inverse(f, lower = -3, upper = 3)
```

Arguments

f	the function to be inverted
lower	a lower bound for the inverse
upper	an lower bound for the inverse

S1 *Core part of the Statistic T*

Description

This function implements the core part of the Cramer-von-Mises test statistic T, denoted by S in AS.

Usage

```
S1(m_bar, sigma_bar, M1, N_k, p)
```

Arguments

m_bar	the sample vector of moments for a specified vector (h_a,r,y)
sigma_bar	the sample covariance matrix of m_bar
M1	number of inequality moments
N_k	index of the $ h_a,r$ function considered
p	parameter p in the statistic

Value

a real number with the statistic evaluated

T_stat *Computation of the test statistic*

Description

This function implements the Computation of the test statistic T given in section 3. "Statistical tests" of "Rationalizing Rational Expectations? Tests and Deviations".

Usage

```
T_stat(m_bar, Sigma_bar, prob_weight, N_g, N_k, p)
```

Arguments

m_bar	the moments m_bar for the different instrumental functions h considered
Sigma_bar	the matrix of all the variances of the moments m_bar for the different instrumental functions h considered
prob_weight	vector of weigths for the test statistic
N_g	number of instrumental functions h considered
N_k	number of moments
p	the parameter p in the Statistic.

Value

a real T which is the test statistic

`test` *Implementation of the RE test with possible survey weights (direct and with parallel computing)*

Description

This function performs the test of rational expectations described in Section 3 of D'Haultfoeuille et al. (2018). By default, the test is implemented without covariates. To perform the test with covariates, one has to indicate in X a non-constant vector or matrix. Also, one can perform the « generalized » tests allowing for aggregate shocks by using the dummy variable generalized. Survey weights can be added. The user can modify the number of cores used by R to reduce the computational time. Tuning parameters used in the test can also be modified.

Usage

```
test(Y_tilde, D, X = matrix(1, length(Y_tilde), 1),
  weights = rep(1/length(Y_tilde), length(Y_tilde)),
  generalized = "No", nbCores = 1, tuningParam = NULL)
```

Arguments

`Y_tilde`	the vector stacking the realisations y then the anticipated values psi of respective sizes n_y and n_p.
`D`	the vector stacking the dummies for the dataset of realisation : n_y ones then n_p zeros
`X`	the matrix of covariates. Set to a vector of 1 by default (in which case the test without covariates is performed).
`weights`	the vector of survey weights. Uniform by default.
`generalized`	whether a generalized test should be performed or not: "Add" for additive shocks (default), "Mult" for multiplicative shocks. Set by default to "No" (no generalized test).
`nbCores`	the number of cores used by the program. To reduce the computational time, this function can use several cores, in which case the library snowfall should be loaded first. By default nbCores is set to 1.
`tuningParam`	a dictionnary (see the example below for modification of the default parameters) containing: - the parameter p in Section 3 of DGM. Default is0.05. - epsilon the parameter epsilonon in Section 3 of DGM. Default value is 0.05 and p is set to 0 if a generalized test is performed. - B the number of bootstrap samples. Default value is 500. - grid_y: the number of points to be tested.

Default is quantile(Y_tilde,seq(0,1,length.out=30)).
- c: the parameter c inSection 3 of DGM. Default is 0.3.
- kappa : the parameter kappapa in Section 3 of DGM. Default is 0.001.
Default values are associated with the test without covariates.

Value

a list containing, in order:

- N, the number of observations
- cv01, the 1% critical value
- cv05, the 5% critical value
- cv10, the 10% critical value
- T_n, the Test ststistic
- B, the number of bootstrap samples
- p_value, the p-value
- T_reps, the vector of bootstraped test statitics.

References

D'Haultfoeuille X, Gaillac C, Maurel A (2018). "Rationalizing Rational Expectations? Tests and Deviations." NBER Working paper <doi:10.3386/w25274>

Andrews D, Shi X (2017). "Inference Based on Many Conditional Moment Inequalities." Journal of Econometrics, 196(2), 275–287.

Andrews DW, Kim W, Shi X (2017). "Commands for testing conditional moment inequalities and equalities." The Stata journal, 17(1).

Examples

```
## The RE test without covariates
n_p=600
n_y=n_p
N <- n_y + n_p
rho <-0.29
sig=0.1
u=1
b=0.10
a=2

psi <-rnorm(n_p,0,u)
pp_y <- runif(n_y,0,1)
zeta <- rnorm(n_y,a,sig)
zeta1 <- rnorm(n_y,-a,sig)
pp1_y <- 1*(pp_y <b)
pp2_y <- 1*(pp_y >1-b)
pp3_y <- 1*(pp_y <=(1-b) & pp_y >=b)
psi_y <-rnorm(n_y,0,u)
y =  rho*psi_y+ pp1_y*zeta + pp2_y*zeta1
```

```
D <- rbind(matrix(1,n_y,1),matrix(0,n_p,1))
Y_tilde <- rbind(matrix(y,n_y,1),matrix(psi,n_p,1))

#res <- test(Y_tilde ,D)
```

`test_base` *The test statistic for the RE test with survey weights*

Description

This is an internal function used in the function test to compute the test statistic with survey weights.

Usage

```
test_base(Y_tilde, X, D, data_test, epsilon, B, N3, c, kappa, p, N,
  weights)
```

Arguments

`Y_tilde`	the vector stacking the realisations y then the anticipated values psi of respective sizes n_y and n_p.
`X`	the matrix of covariates. Set to a vector of 1 by default (in which case the test without covariates is performed).
`D`	the vector stacking the dummies for the dataset of realisation : n_y ones then n_p zeros
`data_test`	the matrix of sample moments
`epsilon`	the parameter epsilonon inSection 3
`B`	the number of bootstrap samples
`N3`	a parameter equal to 1 if no covariates, to N otherwise
`c`	the parameter c in Section 3
`kappa`	the parameter kappapa in Section 3
`p`	the parameter p in Section 3. Equals 0.0 if generalized RE test.
`N`	total number of observations
`weights`	the vector of survey weights. Uniform by default.

Details

By default, the test is implemented without covariates. To perform the test with covariates, one has to indicate in X a non-constant vector or matrix. Also, one can perform the « generalized » tests allowing for aggregate shocks by using the dummy variable generalized. Survey weights can be added. The user can modify the number of cores used by R to reduce the computational time. Tuning parameters used in the test can also be modified.

Value

a list containing, in order:

- T_n : the test statistic

- phi_n: the vector of coresponding GMS functions

- M_bar : the matrix of M_bar in Section 3

References

D'Haultfoeuille X, Gaillac C, Maurel A (2018). "Rationalizing Rational Expectations? Tests and Deviations." CREST Working paper

Andrews D, Shi X (2017). "Inference Based on Many Conditional Moment Inequalities." Journal of Econometrics, 196(2), 275–287.

Andrews DW, Kim W, Shi X (2017). "Commands for testing conditional moment inequalities and equalities." The Stata journal, 17(1).

`which.min2` *Find the min of a list starting from the end*

Description

Find the min of a list starting from the end

Usage

```
which.min2(x, last.index = FALSE, ...)
```

Arguments

`x`	list of elements
`last.index`	starting from the last index (=TRUE). Default is false
`...`	hypotetical additional elements

Index

boot_stat, 2

c_cube, 3
c_fun, 4

estimDev, 4

inverse, 5

S1, 6

T_stat, 6
test, 7
test_base, 9

which.min2, 10