MULTHYPTEST PACKAGE

1. Description

This package was designed to implement hypothesis and power tests relating to constrained regression problems as outlined in Frank Wolak's paper, *Testing Inequality Constraints in Linear Economic Models* (1989). However, that is not to say that its use should be limited to solely constrained regression; the package does generalize to any hypothesis/power question as long as certain conditions (like normality) are met. There are two main hypotheses that this package can implement using the hyptest and complex_hyptest functions. First is equality vs inequality (EI). Given a test variable, x $\in \mathbb{R}^n$, a constraint matrix G, and constraint vector h, the null hypothesis in the EI test asserts that Gx=h, while the alternative is that Gx$\geq$h. The second hypothesis this package inspects is inequality vs unrestricted (IU). In this test, the null asserts that Gx$\geq$h and the alternative is that x is unconstrained or x $\in \mathbb{R}^n$. Both the EI and IU hypothesis tests provide critical insight to determine whether it is reasonable to place different constraints on a test variable given its mean and covariance structure. In the context of regression, these tests help determine whether hypothesized constraints placed on Beta should be considered or not. Like in any hypothesis test, the null should be rejected if the p-value generated by the hyptest or complex_hyptest function is less than the desired signifigance level of the test (tests are all one-sided in implementation so don't manipulate the outputted p-value). The last component of this package is the mult_power function which calculates the power associated with the IU and EI test. Viewing this in the context of regression, the power is other metric of how reasonable it is to impose the hypothesized constraints on Beta. In simple terms, power helps describe how much an unconstrained test term would differ from its constrained counterpart; high power indicates a large difference, while a small power indicates a small difference.

2. Functions

2.1. hyptest.

2.1.1. *Usage.* The hyptest function can perform the IU and EI hypothesis tests exactly as outlined in the description. Should your analysis requires using a mixture of inequality and equality constraints in your null hypothesis at the same time, use complex_hyptest instead.

2.1.2. *Inputs.* hyptest requires 5 inputs
cov: The estimated covariance matrix of your data
mean_hat: The estimated mean (estimated Beta in regression context)
G: The hypothesized quadratic constraint matrix so that Gx$\geq$h or Gx=h (type dependent)
h: The hypothesized quadratic constraint vector such that Gx$\geq$h or Gx=h (type dependent)
type: To run the IU test, input "IU". To run the EI test, input "EI"

2.1.3. *Return.* Graphs the distribution of the test statistic and outputs a parametrically and non-parametrically calculated p-value associated with the Beta_hat given the covariance structure and null constraints.

2.1.4. *Examples.* Below is an example where the term is $\hat{\beta} \in \mathbb{R}^3$ with covariance $\hat{\Sigma} \in \mathbb{R}^{3\times 3}$
EI test: $H_0 : \beta_1 = 0, \beta_2 = 1$ with alternative $H_1 : \beta_1 \geq 0, \beta_2 \geq 1$

$$G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad h = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

EI input code: hyptest($\hat{\Sigma}, \hat{\beta}, G, h,$"EI")
IU test: $H_0 : \beta_1 \geq 0, \beta_2 \geq 1$ with alternative $H_1 : \beta_1 \in \mathbb{R}, \beta_2 \in \mathbb{R}$

$$G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad h = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

IU input code: hyptest($\hat{\Sigma}, \hat{\beta}, G, h,$"IU")

2.1.5. *Code.* The hyptest function is displayed below:

2.2. **complex_hyptest.**

2.2.1. *Usage.* The complex_hyptest function can perform both EI and IU tests like hyptest, but adds a level of complexity to the problem. Unlike hyptest, complex_hyptest tacks on another set of equality restrictions (Ax=b) into the null hypothesis. In both the EI and IU cases, the alternative associated with these new constraints is unrestricted. It is easiest to see exactly what the null and the alternative being implemented in complex_hyptest by looking at the example section.

2.2.2. *Inputs.* complex_hyptest requires 7 inputs
cov: The estimated covariance matrix of your data
mean_hat: The estimated mean
G: The hypothesized quadratic constraint matrix so that Gx≥h or Gx=h (type dependent)
h: The hypothesized quadratic constraint vector such that Gx≥h or Gx=h (type dependent)
A: The hypothesized quadratic equality constraint matrix such that Ax=b
b: The hypothesized quadratic equality constraint vector such that Ax=b
type: To run the IU test, input "IU". To run the EI test, input "EI"

2.2.3. *Return.* Graphs the distribution of the test statistic and outputs a parametrically and non-parametrically calculated p-value associated with the Beta_hat given the covariance structure and null constraints.

2.2.4. *Example.* Below is an example where $\hat{\beta} \in \mathbb{R}^3$ with covariance $\hat{\Sigma} \in \mathbb{R}^{3\times 3}$
IU test: $H_0 : \beta_1 + 2\beta_2 \geq 0, 3\beta_2 + 4\beta_3 \geq 1, 5\beta_1 + 6\beta_3 = 1$ vs unconstrained $H_1 : \beta \in \mathbb{R}^3$.

$$G = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 3 & 4 \end{bmatrix} \qquad h = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad A = \begin{bmatrix} 5 & 0 & 6 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \end{bmatrix}$$

IU input code: complex_hyptest($\hat{\Sigma}, \hat{\beta}, G, h, A, b,$"IU")
EI test: $H_0 : \beta_1 + 2\beta_2 = 0, 3\beta_2 + 4\beta_3 = 1, 5\beta_1 + 6\beta_3 = 1$ vs $H_1 : \beta_1 + 2\beta_2 \geq 0, 3\beta_2 + 4\beta_3 \geq 1$.

$$G = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 3 & 4 \end{bmatrix} \qquad h = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad A = \begin{bmatrix} 5 & 0 & 6 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \end{bmatrix}$$

EI input code: complex_hyptest($\hat{\Sigma}, \hat{\beta}, G, h, A, b,$"EI")
Notice how in both cases, the new constraints defined by A and b bind with equality in the null hypothesis but do not make an appearace in the alternative, because they are unrestricted under the alternative distribution.

2.3. **mult_power.**

2.3.1. *Usage.* The mult_power function determines the power of a given alternative vs the least favorable null hypothesis at the given α level. It serves as a valuable supplement to the hyptest functions in this package, but can also be a strong stand-alone tool that can determine the the power in any problem where there is null hypothesis defined by equality or inequality constraints.

2.3.2. *Inputs.* mult_power requires 7 inputs and has 1 optional input
cov: The estimated covariance matrix of your data
true_mu: The mean you wish to test against the null constraints (Use estimated mean)
G: The hypothesized quadratic constraint matrix so that Gx≥h or Gx=h (type dependent)
h: The hypothesized quadratic constraint vector such that Gx≥h or Gx=h (type dependent)
A: The hypothesized quadratic equality constraint matrix such that Ax=b
b: The hypothesized quadratic equality constraint vector such that Ax=b
type: To calculate the power associated with the IU test, input "IU". To calculate the power associated with the EI test, input "EI"
(*optional input) alpha: Alpha is set to .05 by default, but should you wish to obtain the power at a different signifigance level, you may enter it in as an optional 8th input.

2.3.3. *Return.* Computes and returns the power of true_mu vs the least favorable mean given the constraints.

2.3.4. *Example.* Below is an example where the term is $\hat{\beta} \in \mathbb{R}^3$ with covariance $\hat{\Sigma} \in \mathbb{R}^{3\times 3}$
IU Power: $H_0 : \beta_1 + 2\beta_2 \geq 0, 3\beta_2 + 4\beta_3 \geq 1, 5\beta_1 + 6\beta_3 = 1$ vs unconstrained $H_1 : \beta \in \mathbb{R}^3$.

$$G = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 3 & 4 \end{bmatrix} \qquad h = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad A = \begin{bmatrix} 5 & 0 & 6 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \end{bmatrix}$$

IU Power code: mult_power($\hat{\Sigma}, \hat{\beta}, G, h, A, b,$"IU")
EI Power: $H_0 : \beta_1 + 2\beta_2 = 0, 3\beta_2 + 4\beta_3 = 1, 5\beta_1 + 6\beta_3 = 1$ vs $H_1 : \beta_1 + 2\beta_2 \geq 0, 3\beta_2 + 4\beta_3 \geq 1$.

$$G = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 3 & 4 \end{bmatrix} \qquad h = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad A = \begin{bmatrix} 5 & 0 & 6 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \end{bmatrix}$$

EI Power: mult_power($\hat{\Sigma}, \hat{\beta}, G, h, A, b,$"EI",.1)

3. Disclaimer

While this package is versatile, it does require certain conditions to be met. Firstly, the term being tested must be distributed normally. It is easy to tell if this condition is being violated, because the parametric and non-parametric p-values generated using hyptest or complex_hyptest will be drastically different. The second condition is that the covariance matrix must be positive semi-definite. Otherwise, the program will not be able to invert the matrix and the quadratic optimization program will fail. The last and most important condition is that there cannot be more total constraints than the tested term's dimension and these constraints must be linearly independent. In the complex_hyptest case, it is especially easy to violate this condition because there are two constraint matrices, G and A. Make sure that the the combined matrix, $\begin{bmatrix} G \\ A \end{bmatrix}$ is full row rank and has rows $\leq$ the dimension of the term being tested. Otherwise, the functions will either fail to run or will produce erroneous results.

4. Installation

Download this repository and extract it. From the directory where the Multhyptest file is located type devtools::install() into the R console. After the installation is complete, you can load the library by typing library(Multhyptest).

5. Navigation

All code is located within the R folder. Abbreviated instructions to use each function can be found in the man folder. The description file has an abbreviated description and also has my contact information for any feedback about the package.