University of Duisburg-Essen

Faculty of Business Administration and Economics

Chair of Econometrics

Combining Non-Cointegration Tests in R

Advanced R for Econometricians

Projects

Submitted to the Faculty of

Business Administration and Economics

at the

University of Duisburg-Essen

from:

Jens Klenke, Janine Langerbein, Niklas Mika

Matriculation Number:	3071594 | 3061371 | 3071485
Study Path:	VWL M.Sc.
Reviewer:	Prof. Dr. Christoph Hanck
Secondary Reviewer:	Martin Arnold
Semester:	5$^{\text{th}}$ Semester
Graduation (est.):	Winter Term 2019
Deadline:	April 7th 2020

Contents

List of Tables

List of Figures

1   Introduction

An issue of substantial interest in time series analysis is, whether there exists any meaningful equilibrium relationship between two or more time series variables. Various hypothesis tests have been suggested for testing this so-called cointegration relationship, with the null hypothesis of no cointegration. Their local power, however, relies mostly on a specific nuisance parameter, namely the squared long-run correlations of error terms driving the variables. This may lead to inconclusive results, as one test may reject the null hypothesis, while others accept. The detection of cointegration relationships among time series variables is therefore complicated. Furthermore, the decision for an applicable test poses a challenge for the practitioner.

An approach for resolving this issue might be a combination of the different tests. Bayer and Hanck suggest a method for providing meta tests, with high power for all forms of the nuisance parameter (Bayer & Hanck, 2009). Their approach is based on Fisher's Chi-squared test (Fisher, 1925). It can be shown, that this provides an unambiguous test decision.

So far, there exists a Stata module for computing the above-mentioned non-cointegration test. However, there is no implementation in R yet. Therefore, the objective of this work was the development of the R package **bayerhanck**, to implement the eponymous test. In section 2, the theoretical background of the combined non-cointegration test will be explained further. Next, the structure of the associated R package and its functions will be explained in Section 3. In Section 4, the functions will be illustrated with a real data set. Finally, everything will be recapped in a conclusion.

2   Theory of Non-Cointegration Tests

As mentioned above, Bayer and Hanck (2009) point out, that there is no uniformly most powerful test for cointegration. This is due to the fact, that all common tests are affected by the nuisance parameter R^2, where R^2 is defined as the squared correlation of $\boldsymbol{\nu}_{1t}$ with ν_{2t}.[1] Furthermore, Pesavento (2004) has shown that for varying R^2 different tests are more powerful. Hence it is not clear which test should be applied to the time series data.

To resolve these issues, Bayer and Hanck propose a combination of various cointegration tests. To derive the properties of the underlying tests and

---

[1] $R^2 := \omega_{12}^{'} \boldsymbol{\Omega}_{11}^{-1} \omega_{12} / \omega_{22}$

their combination, Bayer and Hanck use the following model by Pesavento (2004):

$$\triangle \boldsymbol{x}_t = \boldsymbol{\tau}_1 + \boldsymbol{\nu}_{1t} \tag{1}$$
$$y_t = (\mu_2 - \boldsymbol{\theta}' \boldsymbol{\mu}_1) + (\tau_2 - \boldsymbol{\theta}' \boldsymbol{\tau}_1) \, t + \boldsymbol{\theta}' x_t + u_t$$

where $u_t = \rho u_{t-1} + v_{2t}$.

Equation (1) describes the regressors, whereas equation (2) describes the possible cointgeration vector. The observed sample can be denoted as $\boldsymbol{z}_0, \dots, \boldsymbol{z}_T$, where $\boldsymbol{z}_t = (\boldsymbol{x}_t', y_t)$.

Under the following two assumptions and $\rho = 1$ the $\mathcal{H}_0$ hypothesis holds for the vector $\boldsymbol{z}_t$:

Assumption 1:

$\{\boldsymbol{\nu}_t\}$ satisfied a Functional CLT, i.e. $T^{-1/2} \sum_{t=1}^{[\cdot T]} \boldsymbol{\nu}_t \Rightarrow \boldsymbol{\Omega}^{1/2} \boldsymbol{W}(\cdot)$, i.e $\boldsymbol{z}_t$ is $I(1)$

Assumption 2:

There are no cointegrated relationships among the variables in $\boldsymbol{x}_t$

Two essential properties of this model are: firstly, the local power of the underlying tests only depends on the parameters $c := T(\rho - 1)$ and R^2. Secondly, even asymptotically these statistics are only weakly correlated under $\mathcal{H}_0$. (Gregory et al., 2004)

With these central findings it may be possible to achieve a more robust test by combining individual underlying tests. Bayer and Hanck utilise the cointegration tests of Engel and Granger (Engle & Granger, 1987), Johansen (Johansen, 1988), Boswijk (Boswijk, 1994) and Banerjee (Banerjee et al., 1998) as their underlying tests.

Each of these tests has a slightly different approach to testing for a possible cointegration relationship. The Engle-Granger Test tests the null of non cointegration against the alternative hypothesis of at least one cointegration relationship. For this, a simple two-step procedure is used. Firstly, the residuals $\hat{u}_t$ of a regression of the dependent variable on the independent variable is computed. Then, the first differences of these residuals are

regressed on the lagged residuals. The test statistic for the Engle-Granger Test is then the t-statistic t_γ^{ADF} on γ in the aforesaid second regression $\triangle \hat{u}_t = \gamma \hat{u}_{t-1} + \sum_{p=1}^{P-1} \nu_p \triangle \hat{u}_{t-p} + \epsilon_t$. (Engle & Granger, 1987)

As opposed to the Engle-Granger Test, Johansen (1988)'s cointegration test is able to test for h multiple cointegration relationships. The null hypothesis is $h = 0$. To implement this test, a vector error correction model (VECM)

$$\triangle \mathbf{z}_t = \mathbf{\Pi} \mathbf{z}_{t-1} + \sum_{p=1}^{P-1} \mathbf{\Gamma}_p \triangle \mathbf{z}_{t-p} + \mathbf{d}_t + \boldsymbol{\epsilon}_t \tag{2}$$

needs to be estimated first. Bayer and Hanck employ the test statistic $\lambda_{\max}(h) = -T \, ln(1 - \hat{\pi}_1)$, where $\hat{\pi}_1$ denotes the largest solution of $|\pi \boldsymbol{S}_{11} - \boldsymbol{S}_{10} \boldsymbol{S}_{00}^{-1} \boldsymbol{S}_{01}| = 0$.[2]

The cointegration tests of Boswijk (1994) and Banerjee et al. (1998) are based on error correction model tests. Boswijk's test statistic $\hat{F}$ is the Wald statistic for $\mathcal{H}_0 : (\varphi_0, \boldsymbol{\phi}_1') = 0$, whereas Banerjee *et al.*'s test statistic is the t_γ^{ECR} ratio for $\mathcal{H}_0 : \varphi_0 = 0$ of the error correction model. To derive the error correction model, the usual least squares estimate is used: $\triangle y_t = d_t + \pi_{0x}' \triangle \boldsymbol{x}_t + \varphi_0 y_{t-1} + \boldsymbol{\varphi}_1' \boldsymbol{x}_{t-1} + \sum_{P=1}^{P} \left(\boldsymbol{\pi}_{px}' \triangle \boldsymbol{x}_{t-p} + \pi_{py} \triangle y_{t-p} \right) + \epsilon_t$.

From now on, t_i denotes the test statistic of the underlying test i. If the this test rejects for large (small) values, take $\xi_i := t_i(-\xi_i = t_i)$. To combine these tests into a more powerful joint non cointegration test, Bayer and Hanck use an aggregator employed by Fisher (1925)'s Chi-squared test. The test statistic $\tilde{\chi}_{\mathcal{I}}^2$ is than defined as:

$$\tilde{\chi}_{\mathcal{I}}^2 := -2 \sum_{i \in \mathcal{I}} \ln(p_i), \tag{3}$$

where $\mathcal{I}$ is the index set of the ξ_i.

The test statistic has the property, that the distribution under the $\mathcal{H}_0$ hypothesis converges to a random variable $\mathcal{F}_{\mathcal{I}}$ ($\tilde{\chi}_{\mathcal{I}}^2 \to_d \mathcal{F}_{\mathcal{I}}$). This guarantees, that the statistic has a well-defined asymptotic null distribution. According to Bayer and Hanck (2009), this null distribution is nuisance parameter free and only depends on the amount of tests combined. Consequently, the joint $\mathcal{H}_\prime$ hypotheses can be simulated.

---

[2] $\boldsymbol{S}_{ij}$ denotes the moment matrices of reduced rank regression residuals.

Under the alternative hypotheses $\mathcal{H}_1$ the test statistic probability converges to $\tilde{\chi}^2_{\mathcal{I}} \to_P \infty$, which means that the test statistic is consistent if at least one of the underlying tests is consistent.

3 Implementation of the Package bayerhanck in R

The package consists of four functions for the underlying tests, as well as the function for the combined test. Furthermore, the cumulative distribution function of the null hypothesis can be plotted.

3.1 Creating the package

The package **bayerhanck** was created using `create_package()` from **devtools** (Hester, 2020). This generates an R project, files for the package metadata, such as `DESCRIPTION`, `NAMESPACE` and `LICENSE`, and a sub directory for the R code of subsequent functions. The individual functions were documented with the help of **roxygen2** (Wickham, 2020b). The roxygen comments in the source file were then converted to `.Rd` files in the sub directory `man/`.

3.2 Implementation of the underlying Tests

The underlying tests can be carried out by their eponymous commands, namely `englegranger()`, `johansen()`, `banerjee()` and `boswijk()`. The former two partly rely on already implemented functions from the packages **urca** (Pfaff, 2020) and **tsDyn** (Stigler, 2020). Due to the absence of associated functions for the latter two, those had to be programmed manually. All functions take several arguments, from which only `formula` and `data` need to be filled out by the practitioner. Further arguments orientate themselves on the default values defined in the Stata implementation of the Bayer-Hanck Test. For the argument `lags`, which determines the number of lags to be included in the model, the default value is therefore set to 1. The argument `trend` describes the deterministic components of the model. The practitioner may choose from `none`, for no deterministics, `const`, for a (unrestricted) constant and `trend`, for a (unrestricted) constant plus (unrestricted) trend. The default value is set to `const`. Aside from the

Johansen Test, all functions where programmed in such a way, that `trend` = `"none"` simply removes the intercept from all performed linear regressions in the functions. For `trend` = `"trend"` the intercept was maintained and a trend component was added with `seq_along()`. This generates a sequence, which usually takes the length of the dependent variable y. For `trend` = `"const"` the intercept was maintained and thus no changes were made to the linear regression formula. The trend selection for the Johansen Test will be explained later on.

The functions of the underlying tests all return an object of classes `co.test` and `list`. The console also returns the value of the test statistic, as well as the name of the executed test.

In accordance with the previously explained structure, the function `englegranger()` takes the form:

```
englegranger(formula, data, lags = 1, trend = "const")
```

The structure of this function is orientated towards the implementation of the Engle-Granger test in the aforesaid Stata module. Therefore, none of the various existing functions in R is used. Firstly, a linear regression is performed, according to the formula entered in `englegranger()`. Next, an augmented Dickey-Fuller test is applied to the residuals from this regression. For this, the function `ur.df()` from the **urca** package is used. The output value of the test statistic is then used as the test statistic of the Engle-Granger test.

The function for the Johansen test contains the additional argument `type`, which specifies if an maximum eigenvalue or a trace test should be conducted. Therefore, the options are either `trace` or `eigen`, with `eigen` being the default choice. The structure of the function takes the form:

```
johansen(formula, data, type = "eigen", lags = 1, trend = "const")
```

First of all, if the practitioner chooses `trend` = `"trend"`, it has to be renamed to `"both"`. This describes the required name for the following functions to correctly specify the requested deterministic component. Then, the function estimates a VECM by Johansen (MLE) method with `VECM` from the package **tsDyn**[3]. A test of the cointegrating rank is conducted.

_____

[3]Compared to `ca.jo` from the **urca** package, the functions from **tsDyn** allow for the specification of an unrestricted constant and trend, which where used in the Stata Package for the Bayer-Hanck Test.

For this, the function `rank.test()` from the same package is used. Here, the maximum eigenvalue for $h = 0$ is used as the output test statistic of `johansen()`.

The construction of the functions for the Banerjee and Boswijk test was almost identical. Therefore, both functions will be illustrated in a single step. The arguments for both functions are identical to those from `englegranger()`:

```
banerjee(formula, data, lags = 1, trend = "const")
boswijk(formula, data, lags = 1, trend = "const")
```

For the construction of these functions, first differences had to be taken of the dependent, as well as the independent variables. Furthermore, a matrix of the lagged values of all variables in first differences was constructed. Then, the procedure for the Banerjee and Boswijk test were implemented as outlined above. For calculating the test statistics, the coefficients, as well as the covariance-matrix were extracted from the last linear regression model. For the Banerjee test, the coefficient of the first regressor has to be divided by the associated variance from the covariance-matrix. Here, one has to consider, that the position of said values changes, when `trend = "none"`, as the intercept is removed[4].

For the test statistic of the Boswijk test, the vector of coefficients will firstly be matrix multiplied with the inverse covariance-matrix. Next, this product will then again be matrix multiplied with the vector of coefficients.

3.3   Implementation of the function `bayerhanck()`

Bayer and Hanck (2009)'s combined non-cointegration test can be carried out by the command `bayerhanck()`. It accepts all of the arguments of the previously explained underlying tests. Furthermore, it also contains the additional arguments `test`and `crit`, which can be passed to the function call. The function thus takes the form:

```
bayerhanck(formula, data, lags = 1, trend = "const",
           test = "all", crit = 0.05)
```

---

[4]Due to the different order in the Stata output, this could have been ignored there, but poses a threat in R.

The argument `test` determines which combination of the underlying tests should be executed. Here, the practitioner can choose between the default `all` (performs all of the aforementioned tests) or `eg-j`, which only carries out the Engle-Granger and Johansen test.

The argument `crit` sets the significance level of the test. The practitioner may choose between 0.01, 0.05 and 0.1, with the default 0.05.

After inserting the formula and the data, the selected underlying tests are called. The test statistics of each selected test are obtained and stored in a vector. Then, the p-values are calculated by using the corresponding empirical null distribution. This distribution depends on the number of variables, the trend specification and the combination of the underlying tests. Next, the p-values are logarithmised and summed. In accordance with equation (3), they are multiplied with -2 to obtain the test statistic. In the last step, the critical value is selected from the empirical null distribution, subject to the specified significance level.

3.4   Implementation of the function `bayerhanck_1()`

The obtained values from the previously presented function `bayerhanck()` use the critical values provided by the technical appendix from Bayer and Hanck (2009). As an extension to this function, the function `bayerhanck_1()` has been developed. It contains additional null distributions, calculated with a Monte Carlo simulation.

The simulation considers different combinations of the number of variables $K \in \{2, ..., 12\}$, nuisance parameter $R^2 \in \{0, 0.05, ..., 0.95\}$, and cases of determinstic trends in the variables, with (i) = no deterministics, (ii) = constant, and (iii) = constant and trend means. Since only the null distribution is of any interest, the parameter c will be hold constant to zero. The simulation is run with 25,000 repetitions with a vector length of $T = 1000$.

The first step in the loop, to generate the data basis for the simulation, is to draw random shocks u from a normal distribution. Next, the Wiener process is simulated, which is calculated as a cumulative sum by the root of the length of the vector(T). Furthermore, the Ohrnstein Uhlenbeck Process has to be modelled.

Afterwards, the Vienna and Ohrstein Uhlenbeck Process is corrected according to the type of trend. If a constant trend is present, the mean value is subtracted and if both a constant and a trend are present, both are

subtracted. Finally, the common components are calculated.

Then the test statistics of each test are calculated and stored in a vector, which is then sorted. Because we have simulated under the assumptions of $\mathcal{H}_0$ we have now simulated the asymptotic zero distribution.[5]

Table 1 shows the 5% critical values of the simulation in the first block. The values of Bayer and Hanck are in the second, while their difference is shown in the third block. Overall, the simulated values do not deviate too far from those by Bayer and Hanck. It can therefore be assumed, that the simulation delivers reliable results.

Table 1: Comparison of the simulated values with the critical values obtained by Bayer and Hanck

K-1	Simulation			Bayer & Hanck			Difference		
	(i)	(ii)	(iii)	(i)	(ii)	(iii)	(i)	(ii)	(iii)
1	21.4612	21.9236	22.2115	21.352	21.931	22.215	0.1092	-0.0074	-0.0035
2	20.7519	21.034	21.4147	20.776	21.106	21.342	-0.0241	-0.072	0.0727
3	20.4037	20.5608	20.7352	20.237	20.486	20.788	0.1667	0.0748	-0.0528
4	19.8926	20.3695	20.4537	19.951	20.143	20.44	-0.0584	0.2265	0.0137
5	19.7254	20.0637	20.2773	19.747	19.888	20.17	-0.0216	0.1757	0.1073
6	19.5992	19.6468	19.9456	19.564	19.761	19.934	0.0352	-0.1142	0.0116
7	19.4889	19.5973	19.7426	19.471	19.688	19.722	0.0179	-0.0907	0.0206
8	19.5606	19.4804	19.4853	19.471	19.447	19.678	0.0896	0.0334	-0.1927
9	19.3962	19.5299	19.5159	19.365	19.492	19.582	0.0312	0.0379	-0.0661
10	19.1658	19.3574	19.4524	19.268	19.365	19.398	-0.1022	-0.0076	0.0544
11	19.2348	19.2533	19.3622	19.151	19.345	19.404	0.0838	-0.0917	-0.0418

With this simulation, the function `bayerhanck_1` can be implemented. Analogous to the basic version, it takes the form:

```
bayerhanck_1(formula, data, lags = 1, trend = "const",
            test = "all", crit = 0.05)
```

Due to the availability of the simulated distributions under the $\mathcal{H}_0$, the function is now capable of calculating and printing p-values. In its current implementation, the function stayed with the initial options for the argument `test`, `eg-j` and `all`. At this point in time, the amount of data is already large and further increases with each additional test combination by about 7 Megabyte. This is due to the fact, that for each combination of regressors and trend types, a simulated $\mathcal{H}_0$ distribution with 25,000 values is needed. In the current state, the package with the function `bayerhanck_1` has a size of about 16 Megabyte with only two test options. However, in order to reduce the amount of data and allow for further test combinations, the

---

[5]For futher details see Pesavento (2004)

simulated distribution might be reduced to 10,000 values per combination of regressors and trend types, once the package is further developed.

3.5 Implementation of the methods `summary.co.test` and `summary.bh.test`

For a better representation of the calculated test statistics, this package defines methods for the generic function `summary`. For displaying this, one has to call:

```
summary(object, ...)
```

It requires an object of classes `co.test` or `bh.test`, generated by the aforesaid functions. `summary` then prints the formula, the number of lags and the trend specification, entered in the function calls. Furthermore, the matrix with the test statistics, as well as the calculated p-values for the underlying tests is displayed. In a final step, the Bayer-Hanck statistic and its corresponding critical value are shown.

3.6 Implementation of the method `plot.bh.test`

Furthermore, a method for the generic function `plot` has been written. It is available for objects of the class `bh.test` and plots the simulated cumulative distribution function under $\mathcal{H}_0$ with a `ggplot`.

```
plot(x, theme = "dark", ...)
```

The practitioner may use the argument `theme` to choose between `theme = "light"` for a light theme, and `theme = "dark"` for a dark theme. The latter was designed in such a way, that it matches the associated theme of the Shiny app. The dark theme is set to default. However, the practitioner is free to choose any other theme for `ggplot`, by simply adding it with the + operator.

Here, the execution of `plot` accesses the test statistic of the `bayerhanck` function, the number of regressors and the trend type. It can therefore deduce, which null distribution is to be plotted. This information is then passed to the function `ggplot` from the package **ggplot2** (Wickham, 2020a). `ggplot::stat_ecdf()` is then used to plot the null distribution. In addition, the value of the Bayer-Hanck statistic is marked with a dotted x-intercept line.

4   Illustration

The package shall be illustrated on the data set `ibmskpo` from the package **MTS** (Tsay, 2020). It contains monthly simple returns of the stocks of International Business Machines *ibm*, Coca Cola *ko*, and the S&P composite index *sp*. We will test for a cointegration relationship between the variables *sp*, *ko*, and *ibm*. The sample period is from January 1961 till Decembet 2011 and the data can be load into the global entironment with the command `data("mts-examples", package = "MTS")`.To keep a certain clarity we will show the test only in the variant `eg-j`. The procedure can be applied in the same with the option `all`, to include all four underlying tests.

The functions for the underlying tests will be demonstrated with a Engle-Granger test and the Johansen test:

```
eg_test <- englegranger(sp ~ ibm + ko, data = ibmspko)


jo_test <- johansen(sp ~ ibm + ko, data = ibmspko)
```

Next, we will have a look at the summary of the tests:

```
summary(eg_test)


---------------------------------------------------------
Engle-Granger Test
---------------------------------------------------------
Formula: sp ~ ibm + ko
Lags: 1
Trend: const


Value of test statistic: -17.6686
```

and

```
summary(jo_test)


---------------------------------------------------------
Johansen Test
---------------------------------------------------------
Formula: sp ~ ibm + ko
```

```
Lags: 1
Trend: const


Value of test statistic: 279.5175
```

Next, we will calculate the Bayer-Hanck Test, both with the values from Bayer and Hanck (2009)'s technical appendix and the Simulation. First, the results for the basic version are shown:

```
# test = "eg-j"
bh_eg_j <- bayerhanck(sp ~ ibm  + ko, data = ibmspko,
                          test = "eg-j")
```

The summary for the basic version of `bayerhanck()` shows:

```
# test = "eg-j"
summary(bh_eg_j)


-----------------------------------------------------------
Bayerhanck Test for Non-Cointegration
-----------------------------------------------------------
Formula: sp ~ ibm + ko
Lags: 1
Trend: const


Underlying Tests:
               Engle-Granger Johansen
Test Statistics     -17.6686 279.5175
p-Values              0.0000   0.0000


Value of the Fisher Type Test statistic: 110.5241
5% Critical value for the Test statistic: 10.895
```

Next, we will calculate the Bayer-Hanck test with specifically simulated values:

```
# test = "eg-j"
bh_eg_j_1 <- bayerhanck_1(sp ~ ibm + ko, data = ibmspko,
                          test = "eg-j")
```

The summary then shows:

```
# test = "eg-j"
summary(bh_eg_j_1)


------------------------------------------------------------
Bayerhanck Test for Non-Cointegration
------------------------------------------------------------
Formula: sp ~ ibm + ko
Lags: 1
Trend: const

Underlying Tests:
                Engle-Granger Johansen
Test Statistics      -17.6686 279.5175
p-Values               0.0000   0.0000

Value of the Fisher Type Test statistic: 110.5241
5% Critical value for the Test statistic: 10.8454
P-value of the Fisher Type Test statistic: 0
```

The test statistics of the individual tests, but also of the combined tests, are relatively clear. Basically all tests reject the $\mathcal{H}_0$, that there is no cointegration. But as Bayer and Hanck (2009) have shown, there are several examples from the literature where the evidence is not so clear. A further interpretation of the results shall not be made here.

To demonstrate the implemented method for the generic function `plot()`, we will additionally plot the object from `bayerhanck_1` for `test = "all"`. For a harmonious appearance in this documentation, we will set `theme = "light"`.

```
plot(bh_all_1, theme = "light")
```
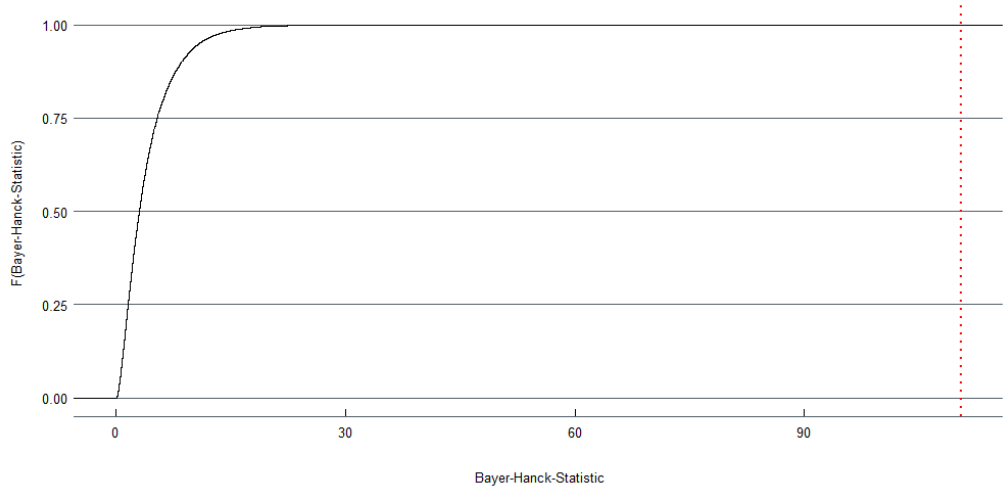
Figure 1: Cumulative Distribution Function under $\mathcal{H}_0$

5 Conclusion

The implementation of the R package **bayerhanck** has been documented. It contains functions for two different versions of the Bayer-Hanck Test for non cointegration, as well as functions for the underlying cointegration tests. The former has been implemented in a functional way, so that the functions `bayerhanck()` and `bayerhanck_1()` call the underlying test functions. Since the underlying tests can be executed autonomously, the scope of application is increased. The sub-functions certainly require more effort so the tests can be applied to more versatile problems. This applies above all to the tests of Banerjee and Boswijk, as we found no well working implementation in R.

In the current version we also developed a further implementation with the `bayerhanck_1()` function. This function additionally calculates the p-value of the test statistics based on the empirical distribution stored in the function. For the aforementioned data set with two regressors and approximately 26,000 observations, the function needs about one second computation time. Further speed up of the calculation might be desired, if one wants to work with bigger data. One consideration might be to write the code directly in `C`, instead of `R`, to save a step in the translation process.

Furthermore, two methods for the generic functions `summary()` and `plot()` were defined Another implementation, which was not covered in this documentation, is a user friendly application within a shiny app. If it is published on a server, potential users might use the functions of this package without using `R` themselves.

Of course, this package will be an object to further development. A plausible advancement could be to increase the number of possible tests and allow other combinations, than already implemented. For this, the simulated distributions of the null distribution would have to be stored. The power of each combination would have to be checked first. However, this might lead to the problem, that users may tend to try out different combinations, until they get their desired result.

References

Banerjee, **A.**, **Dolado**, **J. J.**, & **Mestre**, **R.** (1998). Error-correction mechanism tests for cointegration in a single-equation framework. *Journal of Times Series Analysis*, 267–283.

Bayer, **C.**, & **Hanck**, **C.** (2009). Combining Non-Cointegration Tests.

Boswijk, **H. P.** (1994). Testing for an unstable root in conditional and structural error correction models. *Journal of Econometrics*, *63*(1), 37–60. https://EconPapers.repec.org/RePEc:eee:econom:v:63:y:1994:i:1:p:37-60

Engle, **R.**, & **Granger**, **C.** (1987). Co-integration and error correction: Representation, estimation, and testing. *Econometrica*, *55*(2), 251–76. https://EconPapers.repec.org/RePEc:ecm:emetrp:v:55:y:1987:i:2:p:251-76

Fisher, **R.** (1925). *Statistical methods for research workers*. Edinburgh Oliver & Boyd.

Gregory, **A. W.**, **Haug**, **A. A.**, & **Lomuto**, **N.** (2004). Mixed signals among tests for cointegration [Publisher: Wiley]. *Journal of Applied Econometrics*, *19*(1), 89–98. Retrieved April 5, 2020, from https://www.jstor.org/stable/25146268

Hester, **J.** (2020, April 7). *Devtools package | r documentation*. Retrieved April 7, 2020, from https://www.rdocumentation.org/packages/devtools/versions/2.2.2

Johansen, **S.** (1988). Statistical analysis of cointegration vectors. *Journal of Economic Dynamics and Control*, *12*(2-3), 231–254. https://ideas.repec.org/a/eee/dyncon/v12y1988i2-3p231-254.html

Pesavento, **E.** (2004). Analytical evaluation of the power of tests for the absence of cointegration. *Journal of Econometrics*, *122*, 349–384.

Pfaff, **B.** (2020, April 7). *Urca package | r documentation*. Retrieved April 7, 2020, from https://www.rdocumentation.org/packages/urca/versions/1.3-0

Stigler, **M.** (2020, April 7). *tsDyn package | r documentation*. Retrieved April 7, 2020, from https://www.rdocumentation.org/packages/tsDyn/versions/0.9-44

Tsay, **R. S.** (2020, April 7). *MTS package | r documentation.* Retrieved April 7, 2020, from https://www.rdocumentation.org/packages/MTS/versions/1.0

Wickham, **H.** (2020a, April 7). *Ggplot2 package | r documentation.* Retrieved April 7, 2020, from https://www.rdocumentation.org/packages/ggplot2/versions/3.3.0

Wickham, **H.** (2020b, April 7). *Roxygen2 package | r documentation.* Retrieved April 7, 2020, from https://www.rdocumentation.org/packages/roxygen2/versions/7.1.0

Eidesstattliche Versicherung

Ich versichere an Eides statt durch meine Unterschrift, dass ich die vorstehende Arbeit selbständig und ohne fremde Hilfe angefertigt und alle Stellen, die ich wörtlich oder annähernd wörtlich aus Veröffentlichungen entnommen habe, als solche kenntlich gemacht habe, mich auch keiner anderen als der angegebenen Literatur oder sonstiger Hilfsmittel bedient habe. Die Arbeit hat in dieser oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen.

Essen, den _____   _____

Jens Klenke, Janine Langerbein, Niklas Mika