ECON90033 – QUANTITATIVE ANALYSIS OF FINANCE I

TUTORIAL 12

Download the t12e1 Excel data file from the subject website and save it to your computer or USB flash drive. Read this handout and complete the tutorial exercises before your tutorial class so that you can ask for help during the tutorial if necessary.

This is the last tutorial of the semester, and it has a single exercise on *VAR*, cointegration and *VECM*.

Exercise 1

The *t12e1.xlsx* file contains monthly data from December 1949 to September 2023on two US time series

TB3: 3-Month Treasury Bill secondary market rate, percent, not seasonally adjusted,
AAA: Moody's seasoned Aaa corporate bond yield, percent, not seasonally adjusted.

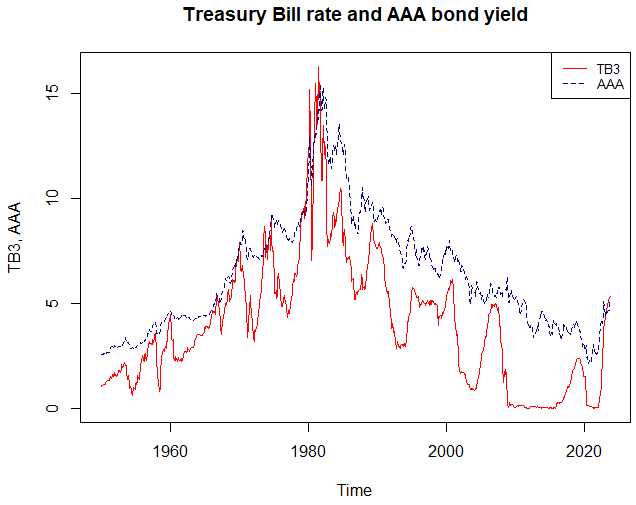
TB3 represents the short-term interest rate, and the *AAA* bond rate is an index of the performance of all bonds given an Aaa rating by Moody's Investors Service. *TB3* and *AAA* can be seen as alternative ways of investment in low-risk securities, and they are expected to have a positive relationship.

a) Launch *RStudio*, create a new project and script, and name both *t12e1*. Import the data set from the *t12e1.xlsx* file to *RStudio*, save it as *t12e1.RData*, and attach it to your *R* project.

Plot *TB3* and *AAA* by executing the following commands:

```
TB3 = ts(TB3, start = c(1949, 12), end = c (2023, 9), frequency = 12)
AAA = ts(AAA, start = c(1949, 12), end = c (2023, 9), frequency = 12)
plot.ts(TB3, ylab = "TB3, AAA", main = "Treasury Bill rate and AAA bond yield",
        col = "red")
lines(AAA, col = "darkblue", lty = 2)
legend("topright", legend = c("TB3", "AAA"),
        col = c("red", "darkblue"), lty = 1:2, cex = 0.8)
```

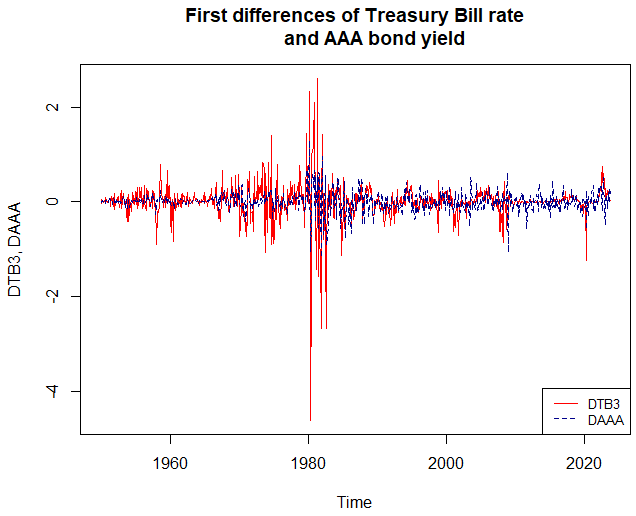
They produce the figure shown on the next page. It is clear even visually that both variables are non-stationary as they have prolonged upward and downward movements. However, the two series tend to follow similar time paths, so they might share a common stochastic trend.



To get further insights in the nature of these variables, calculate and plot their first differences by executing

```
DTB3 = diff(TB3)
DAAA = diff(AAA)
plot.ts(DTB3, ylab = "DTB3, DAAA", main = "First differences of Treasury Bill rate
        and AAA bond yield", col = "red")
lines(DAAA, col = "darkblue")
legend("bottomright", legend = c("DTB3", "DAAA"),
         col = c("red", "darkblue"), lty = 1:2, cex = 0.8)
```

The new figure,



shows that the differenced series do not contain trends, so at least as their means are concerned, they might be stationary.

b) To confirm these observations, perform the *ADF* and *KPSS* tests on the levels and first differences of *TB3* and *AAA*.

Based on the time series plots, the data generating processes certainly do not have a single linear trend component, but they might have a broken linear trend component. For this reason, an intercept must be included in the *ADF*, *ERS* and *KPSS* test regressions (Model 2) and both an intercept and a trend in the *ZA* test regression (Model 3) for the level series. As for the first differences, although differencing wipes out the intercept from Model 2 (and the linear trend from Model 3), the *ERS* and *KPSS* test regressions must have at least an intercept, so in these tests we keep using Model 2.

Perform the following commands:

```
# DF test
library(urca)
adf.TB3 = ur.df(TB3, type = "drift", selectlags = "BIC")
summary(adf.TB3)
adf.DTB3 = ur.df(diff(TB3), type = "none", selectlags = "BIC")
summary(adf.DTB3)
adf.AAA = ur.df(AAA, type = "drift", selectlags = "BIC")
summary(adf.AAA)
adf.DAAA = ur.df(diff(AAA), type = "none", selectlags = "BIC")
summary(adf.DAAA)

# KPSS test
kpss.TB3 = ur.kpss(TB3, type = "mu")
summary(kpss.TB3)
kpss.DTB3 = ur.kpss(diff(TB3), type = "mu")
summary(kpss.DTB3)
kpss.AAA = ur.kpss(AAA, type = "mu")
summary(kpss.AAA)
kpss.DAAA = ur.kpss(diff(AAA), type = "mu")
summary(kpss.DAAA)
```

The results are not shown here, but if you run these commands you can see that the consensus is that both variables are integrated of order one, i.e., I(1), no matter whether the significance level is 10% or 5%.

c) Since *TB3* and *AAA* are both I(1), they might be cointegrated, i.e., CI(1,1). Check this possibility by performing cointegration tests.

You learnt about three tests for cointegration, the Engle-Granger (*EG*) test, and the trace and maximum eigenvalue tests of Johansen (*J*).

The *EG* test is basically an *ADF*-type test on the residuals from a simple linear regression of *TB3* on *AAA*, or *AAA* on *TB3*, but the critical values are different. Using *R*, *EG* tests for the null hypothesis that two or more time series, each of which is I(1), are not cointegrated, can be performed with the following function of the *aTSA* library:

coint.test(y, X, d = , nlag =)

where y is one of the variables, X is the other variable (or matrix of the other variables), d is the order of differencing for all variables (the default is 0), and nlag is one less than the lag length in the test regression (the default value of nlag is floor(4(length(y)/100)^(2/9))).*

Since the results can be sensitive to normalization, i.e., to the choice of variable *y*, the best is to run the test twice by executing the following commands:

```
library(aTSA)
coint.test(TB3, AAA)
coint.test(AAA, TB3)
```

They return the following printouts:

```
Response: TB3
Input: AAA
Number of inputs: 1
Model: y ~ X + 1
------------------------------
Engle-Granger Cointegration Test
alternative: cointegrated

Type 1: no trend
    lag      EG p.value
   6.00   -3.88    0.01
-----
 Type 2: linear trend
    lag      EG p.value
  6.000  -0.633   0.100
-----
 Type 3: quadratic trend
    lag      EG p.value
 6.0000  0.0773  0.1000
-----------
Note: p.value = 0.01 means p.value <= 0.01
    : p.value = 0.10 means p.value >= 0.10
```

```
Response: AAA
Input: TB3
Number of inputs: 1
Model: y ~ X + 1
------------------------------
Engle-Granger Cointegration Test
alternative: cointegrated

Type 1: no trend
    lag      EG p.value
  6.000  -3.337   0.017
-----
 Type 2: linear trend
    lag      EG p.value
    6.0     0.3     0.1
-----
 Type 3: quadratic trend
    lag      EG p.value
   6.00    1.14    0.10
-----------
Note: p.value = 0.01 means p.value <= 0.01
    : p.value = 0.10 means p.value >= 0.10
```

Each printout has three panels, corresponding to the three possible specifications of the *ADF* test regression, i.e., Model 1, Model 2, and Model 3.[1] Only the first panels are relevant this time.

Comparing the first panels to each other, you can see that normalization matters since the test statistic is -3.88 on the first printout and -3.337 on the second printout. Accordingly, the reported *p*-values are also different, ≤ 0.01 on the first printout and 0.017 on the second. Nevertheless, in both cases the null hypothesis of no cointegration can be rejected at the 2% significance level, implying that *TB3* and *AAA* are cointegrated of orders 1 and 1, i.e., CI(1,1).

Using *R*, the *J* tests can be performed with the following function of the *urca* package:

ca.jo(x, type = , ecdet = , K = , spec =)

where x is a data matrix to be tested for cointegration, type is either "trace" or "eigen" for the J trace test and maximum eigenvalue test, K is the lag length in the corresponding VAR specification (the lag length in VECM is K – 1), and ecdet refers to the deterministic terms in the EC term and it can be "none" (no deterministic term in EC but a constant outside EC), "const" (a constant in EC, but no deterministic term outside EC), or "trend" (a trend variable but no constant in EC, and a constant outside EC).

[1] LK: The '*no trend*', '*linear trend*', '*quadratic trend*' labels on the printouts refer to the fact that under the null hypothesis of the *ADF* test the intercept generates a linear trend and the trend variable generates a quadratic trend.

In general, ecdet = "none" is appropriate for (linearly) trending series, granted that all trends are stochastic, ecdet = "const" is proper only if none of the variables appears to have a sustained tendency to increase or decrease, and "trend" is reasonable when there is some long-run linear growth that the cointegrating relation does not account for.

Since neither *TB3* nor *AAA* are trending linearly but appear to follow some broken trend, in this example *ecdet = "const"* is a reasonable option.

As regards the lag length, we can rely on the *VARselect()* function, like in Exercise 1 of Tutorial 10.

```
data = cbind(AAA, TB3)
library(vars)
VARselect(data, type = "const")
```

return the following printout:

```
$selection
AIC(n)  HQ(n)  SC(n) FPE(n)
    10     10      3     10

$criteria
                1            2            3            4            5
AIC(n) -5.33838552 -5.532230868 -5.610549434 -5.610157777 -5.606011398
HQ(n)  -5.32587465 -5.511379421 -5.581357408 -5.572625171 -5.560138214
SC(n)  -5.30567753 -5.477717557 -5.534230798 -5.512033816 -5.486082113
FPE(n)  0.00480362  0.003957152  0.003659061  0.003660497  0.003675711
                6            7            8            9           10
AIC(n) -5.601631527 -5.658245229 -5.652268434 -5.670675971 -5.68893683
HQ(n)  -5.547417763 -5.595690886 -5.581373512 -5.591440471 -5.60136075
SC(n)  -5.459896917 -5.494705294 -5.466923174 -5.463525388 -5.45998092
FPE(n)  0.003691852  0.003488657  0.003509581  0.003445582  0.00338325
```

SC selects *VAR*(3), while the other three criteria favour *VAR*(10).

Estimate *VAR*(3) and perform the *BG* test for autocorrelation of orders 1 – 3,

```
var3 = VAR(data, p = 3, type = "const")
serial.test(var3, lags.bg = 3, type = "BG")
```

to get this printout:

```
        Breusch-Godfrey LM test

data:  Residuals of VAR object var3
Chi-squared = 16.225, df = 12, p-value = 0.1811
```

The *p*-value is 0.1811 > 0.10, so the null hypothesis of no autocorrelation of orders 1 – 3 is maintained even at the 10% significance level.[2]

For this reason we use $K = 3$.

Execute

```
data = cbind(AAA, TB3)
j.trace = ca.jo(data, type = "trace", K = 3, ecdet = "const")
summary(j.trace)
```

to get the following printout:

```
######################
# Johansen-Procedure #
######################

Test type: trace statistic , without linear trend and constant in cointegration

Eigenvalues (lambda):
[1]  3.781984e-02  4.780644e-03 -2.930396e-19

Values of teststatistic and critical values of test:

          test 10pct  5pct  1pct
r <= 1 |  4.23  7.52  9.24 12.97
r = 0  | 38.27 17.85 19.96 24.60

Eigenvectors, normalised to first column:
(These are the cointegration relations)

            AAA.l3     TB3.l3   constant
AAA.l3    1.000000  1.0000000  1.0000000
TB3.l3   -1.026532  0.0557922 -0.2276885
constant -2.258470 -7.0850332  7.3511208

Weights W:
(This is the loading matrix)

           AAA.l3       TB3.l3      constant
AAA.d -0.01913395 -0.002857251  7.630392e-19
TB3.d  0.01069988 -0.008111077 -7.637752e-19
```

[2] LK: If you repeat the *BG* test with higher *lags.bg*, you can see that there is higher order autocorrelation. Higher order models, like *VAR*(10), have the same problem, so there is no benefit in using more lags. For this reason, we stick to *VAR*(3) acknowledging that it is not a 'perfect' model.

This printout has four pats. The first shows three estimated eigenvalues in decreasing order.

The second part shows the trace test statistics for the null hypotheses of $r_0 \leq 1$ and $r_0 = 0$, and the 1%, 5%, 10% critical values. They have to be evaluated in reverse order, i.e., starting with $r_0 = 0$, until the null hypothesis is first maintained. As you can see, the first test statistic, 38.27, is larger than the corresponding 1% critical value (24.60), but the second test statistic (4.23) is smaller than the corresponding 10% critical value (7.52). Hence, at the 1%, 5% and 10% critical values alike, $r_0 = 0$ is rejected but $r_0 \leq 1$ is maintained, implying that $r = 1$.

The third part shows three estimated cointegration relations that correspond to the three eigenvalues in the first table. Since we concluded $r = 1$, we consider only the first, eigenvalue and cointegrating relation. The estimated *EC* term, normalized with respect to the first endogenous variable, is

$$\widehat{AAA}_t - 2.2585 - 1.0265TB3_t = \varepsilon_t$$

Finally, the fourth part of the printout shows the estimated speed of adjustment coefficients. Again, only the first column is relevant, i.e., α' = [-0.0191, 0.0107].

The *J* maximum eigenvalue test can be performed similarly. Execute,

```
j.eigen = ca.jo(data, type = "eigen", K = 3, ecdet = "const")
summary(j.eigen)
```

Only the second part of the new printout

```
######################
# Johansen-Procedure #
######################

Test type: maximal eigenvalue statistic (lambda max) ,
without linear trend and constant in cointegration


Values of teststatistic and critical values of test:

          test 10pct  5pct  1pct
r <= 1 |  4.23  7.52  9.24 12.97
r = 0  | 34.04 13.75 15.67 20.20
```

differs from the previous printout, the rest are the same. The statistical decisions, however, do not change, we reject the first null hypothesis but maintain the second, implying $r = 1$.

d) Estimate a *VECM*.

Like testing for cointegration, *VECM* can be estimated based on the Engle-Granger or on the Johansen approach.

In the former case, we have to estimate the long-run equilibrium relationship between the two variables, and then estimate the *VECM* equations one-by-one with the lagged residuals from the equilibrium regression used as the *EC* term.

To perform the first step, execute

```
EC.1 = lm(AAA ~ TB3)
summary(EC.1)
```

As you can see on the printout shown on the next page, the estimated long-run equilibrium relationship is

$$\widehat{AAA}_t = 3.1347 + 0.7962TB3_t$$

and there is a significantly positive relationship between *AAA* and *TB3*.

```
Call:
lm(formula = AAA ~ TB3)

Residuals:
    Min      1Q  Median      3Q     Max
-2.6549 -1.2472 -0.0918  1.2109  3.6640

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.13472    0.07539   41.58   <2e-16 ***
TB3          0.79623    0.01485   53.63   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.359 on 884 degrees of freedom
Multiple R-squared:  0.7649,    Adjusted R-squared:  0.7646
F-statistic:  2876 on 1 and 884 DF,  p-value: < 2.2e-16
```

Save the residuals from this regression,

```
e.1 = ts(EC.1$residuals, start = c(1950,1), end = c (2023,9), frequency = 12)
```

take the first difference of *AAA*,

```
DAAA = window(diff(AAA), start = c(1950,2), end = c (2023,9),
                    frequency = 12)
```

the lag of *e.1*,

```
le.1 = window(lag(e.1, -1), start = c(1950,2), end = c (2023,9), frequency = 12)
```

and regress the former on the latter,

```
ec.11 = lm(DAAA ~ le.1)
summary(ec.11)
```

to get the following printout:

```
Call:
lm(formula = DAAA ~ le.1)

Residuals:
     Min       1Q   Median       3Q      Max
-1.16592 -0.08071 -0.01575  0.07993  1.23484

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.003034   0.006834   0.444    0.657
le.1        -0.026308   0.005047  -5.213 2.32e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2032 on 882 degrees of freedom
Multiple R-squared:  0.02989,   Adjusted R-squared:  0.02879
F-statistic: 27.17 on 1 and 882 DF,  p-value: 2.319e-07
```

From this printout the first equation of *VECM* is

$$\widehat{DAAA}_t = 0.0030 - 0.0263 ec_{t-1}$$

To get the second equation of *VECM*, execute

```
DTB3 = window(diff(TB3), start = c(1950,2), end = c (2023,9),
              frequency = 12)
ec.12 = lm(DTB3 ~ le.1)
summary(ec.12)
```

The new printout is on the next page. From this printout the second equation of *VECM* is

$$\widehat{DTB3}_t = 0.0048 + 0.0042 ec_{t-1}$$

Considering the details, the first regression is strongly significant, though R^2 is very low. The estimate of the speed of adjustment coefficient (the coefficient of *le.1*) is significantly negative but significantly larger than -2, as required by stability.[3]

[3] LK: These conclusions can be drawn from one-tail *t*-tests. The first is a left-tail test with the null hypothesis that the slope parameter is 0, and since the point estimate is negative and the reported two-tail p-value, *Pr(>|t|)*, is practically zero, the null hypothesis can be rejected. The second test is a right-tail test with the null hypothesis that the slope parameter is -2. The corresponding test statistic is (-0.0263 – (-2)) / 0.0050 = 394.74, much larger than any right-tail *t* critical value with *df* = 882, so again the null hypothesis can be rejected.

```
Call:
lm(formula = DTB3 ~ le.1)

Residuals:
    Min      1Q  Median      3Q     Max
-4.6153 -0.0866  0.0029  0.1135  2.6071

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.004786   0.013276   0.360    0.719
le.1        0.004173   0.009805   0.426    0.671

Residual standard error: 0.3947 on 882 degrees of freedom
Multiple R-squared:  0.0002053, Adjusted R-squared:  -0.0009282
F-statistic: 0.1811 on 1 and 882 DF,  p-value: 0.6705
```

The second regression is insignificant and the estimate of the speed of adjustment coefficient is only insignificantly positive. This means that the system of *AAA* and *TB3* adjusts to deviations from the long-run equilibrium through changes in *AAA* and the development of *TB3* is not directly linked to the equilibrium error.

Before moving on to the Johansen approach, two comments are warranted. First, to save time, we did not check whether the estimated *VECM* equations are free of autocorrelation. In fact, as you can easily verify, they are not, but by augmenting these equations with the first lags of the first differences of *AAA* and *TB3*, it is possible to eliminate autocorrelation.

The second comment is that just like the *EG* test, the results of the *EG* two-step procedure can be sensitive to normalization. For this reason, in practice it would be important to re-estimate the long-run equilibrium relationship between the two variables normalized to *TB3* this time, take the residuals from the new equilibrium regression, and estimate the two equations of *VECM*. To save time, we skip this step in the tutorial, but you can still do it on your own in preparation for the exam.

Let's turn our attention instead to the Johansen procedure.

After having performed the *J* test with the *ca.jo()* function, the following function of the *urca* package provides the corresponding OLS regression of *VECM*:

cajorls(z, r =)

where z is the previously created ca.jo object, either with type = "trace" or type = "eigen".

In our example, $r = 1$, so execute[4]

```
vecm.j = cajorls(j.trace, r = 1)
print(vecm.j)
```

to get the following printout:

```
Call:
lm(formula = substitute(form1), data = data.mat)

Coefficients:
          AAA.d     TB3.d
ect1     -0.01913   0.01070
AAA.dl1   0.34070   0.29626
TB3.dl1   0.04067   0.34051
AAA.dl2  -0.27653  -0.24907
TB3.dl2   0.02975  -0.13554


$beta
              ect1
AAA.l3    1.000000
TB3.l3   -1.026532
constant -2.258470
```

Comparing this printout to the *J trace* test printout on page 8, you can see that (i) the coefficients of the *EC* term (*ect1*) are the same estimated speed of adjustment coefficients term as in the fourth part of the *J trace* test printout, and (ii) the *$beta* vector is the estimated cointegration relation and it is the same as the first column vector in the third part of the *J trace* test printout.

From this printout the estimated *VECM*(2) is:

$$\widehat{DAAA_t} = -0.0191 ect_{t-1} + 0.3407 DAAA_{t-1} - 0.2765 DAAA_{t-2} + 0.0407 DTB3_{t-1} + 0.0298 DTB3_{t-2}$$

$$\widehat{DTB3_t} = 0.0107 ect_{t-1} + 0.2963 DAAA_{t-1} - 0.2491 DAAA_{t-2} + 0.3405 DTB3_{t-1} - 0.1355 DTB3_{t-2}$$

and the *EC* term is

$$ect_t = AAA_t - 2.2585 - 1.0265 TB3_t$$

[4] It does not make any difference whether *j.trace* or *j.eigen* is used as *z*.

e) Obtain the *VAR*(3) representation of the estimated *VECM*(2).

A *VECM* can be transformed to an equivalent *VAR* in levels with the following function:

vec2var(z, r =)

where z is an object of class 'ca.jo' generated by function ca.jo().

Execute

```
var.j = vec2var(j.trace, r = 1)
print(var.j)
```

to get the following printout:

```
Coefficient matrix of lagged endogenous variables:

A1:
       AAA.l1     TB3.l1
AAA 1.3406951 0.04066789
TB3 0.2962563 1.34051385


A2:
        AAA.l2      TB3.l2
AAA -0.6172266 -0.01092015
TB3 -0.5453254 -0.47605324


A3:
       AAA.l3      TB3.l3
AAA 0.2573975 -0.01010613
TB3 0.2597690  0.12455562


Coefficient matrix of deterministic regressor(s).

       constant
AAA  0.04321345
TB3 -0.02416536
```

From this, the *VAR*(3) model can be written out as

$$\widehat{AAA_t} = 0.0432 + 1.3407\,AAA_{t-1} - 0.6172\,AAA_{t-2} + 0.2574\,AAA_{t-3} + 0.0407TB3_{t-1} - 0.0109TB3_{t-2} - 0.0101TB3_{t-3}$$

$$\widehat{TB3_t} = -0.0242 + 0.2963\,AAA_{t-1} - 0.5453\,AAA_{t-2} + 0.2598\,AAA_{t-3} + 1.3405TB3_{t-1} - 0.4761TB3_{t-2} + 0.1246TB3_{t-3}$$

Given this *VAR* model, it is possible to obtain impulse response functions and perform forecast error variance decomposition just like in Exercise 1 of Tutorial 11.