MOOC Econometrics

Test Exercise 6

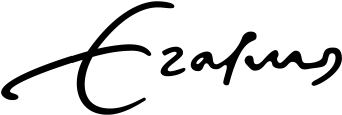
**Notes:**

* See website for how to submit your answers and how feedback is organized.
* This exercise uses the datafile TestExer6 and requires a computer.
* The dataset TestExer6 is available on the website.

**Goals and skills being used:**

* Experience the process of practical application of time series analysis.
* Get hands-on experience with the analysis of time series.
* Give correct interpretation of outcomes of the analysis.

**Questions**

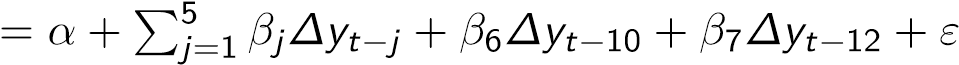
This test exercise uses data that are available in the data file T estExer6. T he q uestion o f i nterest i s t o model monthly production of Toyota passenger cars and to investigate whether the monthly production of all other brands of Japanese passenger cars has predictive power for the production of Toyota. Monthly production data are available from January 1980 until December 2000. The data for January 1980 until December 1999 are used for specification and estimation of models, and the data for 2000 are left out for forecast evaluation purposes.

In answering the questions below, you should use the seasonally adjusted production data denoted by ‘toyota-sa’ and ‘other-sa’. We will denote these variables by *y* = toyota-sa and *x* = other-sa.

1. Make time series plots of the variables *yt* and *xt*, and also of the share of Toyota in all produced passenger cars, that is *yt/*(*yt* + *xt*). What conclusions do you draw from these plots?
2. (i) Perform the Augmented Dickey-Fuller (ADF) test for *yt*. In the ADF test equation, include a constant (*α*) and three lags of *∆yt*, as well as the variable of interest, *yt*−1. Report the coefficient of *yt*−1 and its standard error and *t*-value, and draw your conclusion.

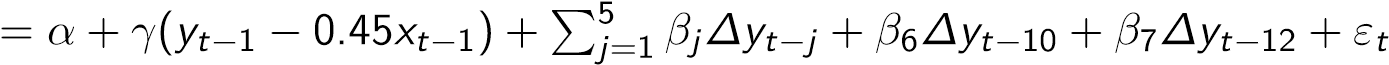
(ii) Perform a similar ADF test for *xt*.

1. Perform the two-step Engle-Granger test for cointegration of the time series *yt* and *xt*. In step 1, regress *yt* on a constant and *xt*. In step 2, perform a regression of the residuals *et* in the model *∆et* = *α* + *ρet*−1 + *β*1*∆et*−1 + *β*2*∆et*−2 + *β*3*∆et*−3 + *ωt*. What is your conclusion?
2. Construct the first twelve sample autocorrelations and sample partial autocorrelations of *∆yt* and use the outcomes to motivate an AR(12) model for *∆yt*. Check that only the lagged terms at lags 1 to 5, 10, and

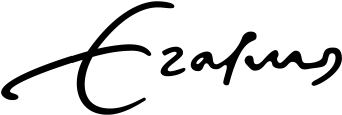
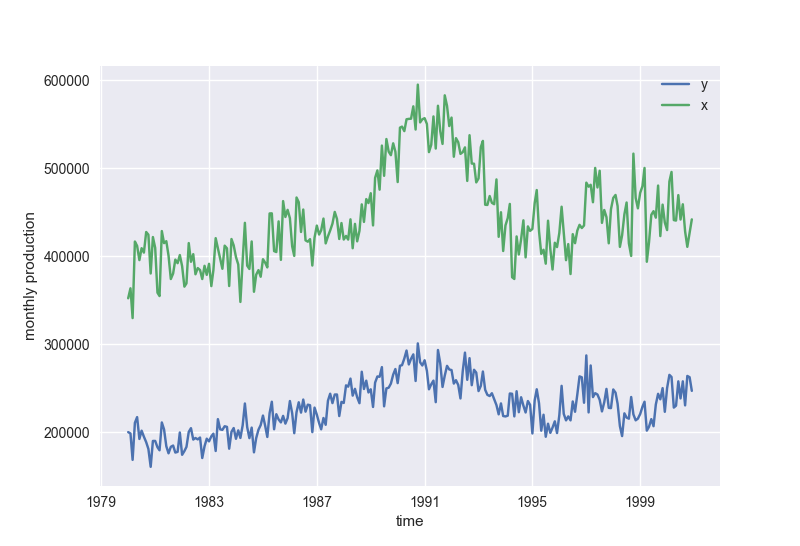
12 are significant, and estimate the following model: *∆yt* *t*

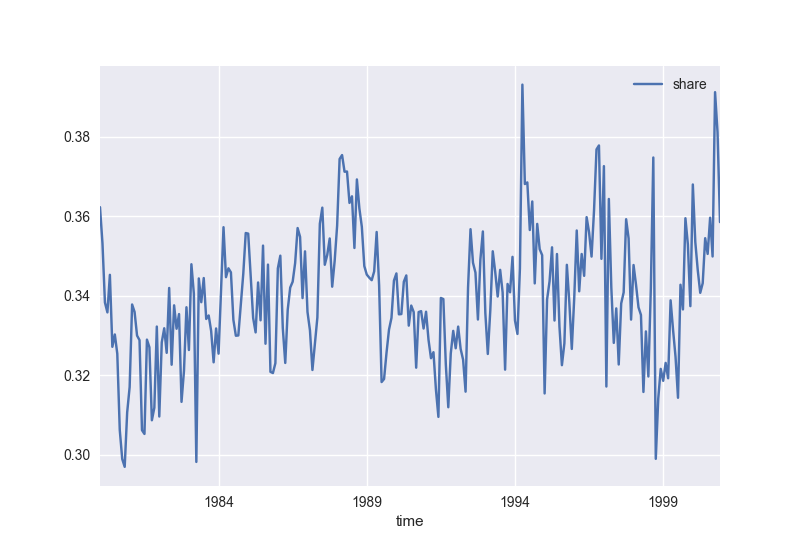
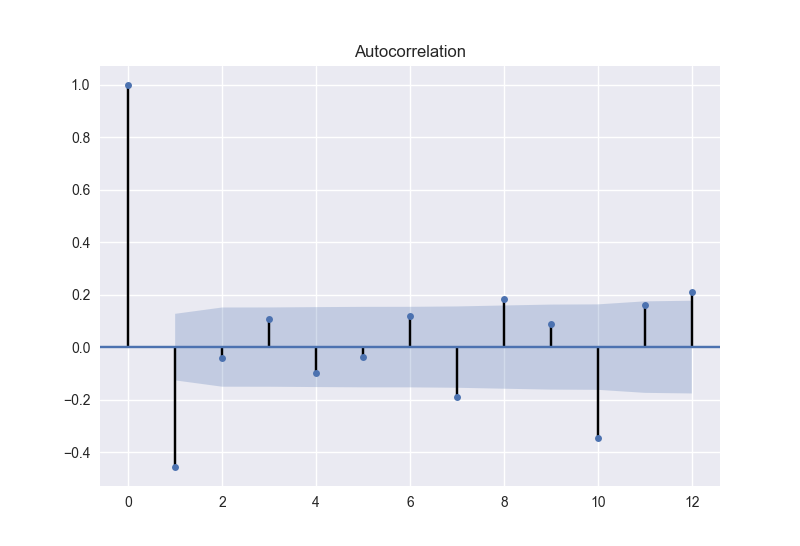
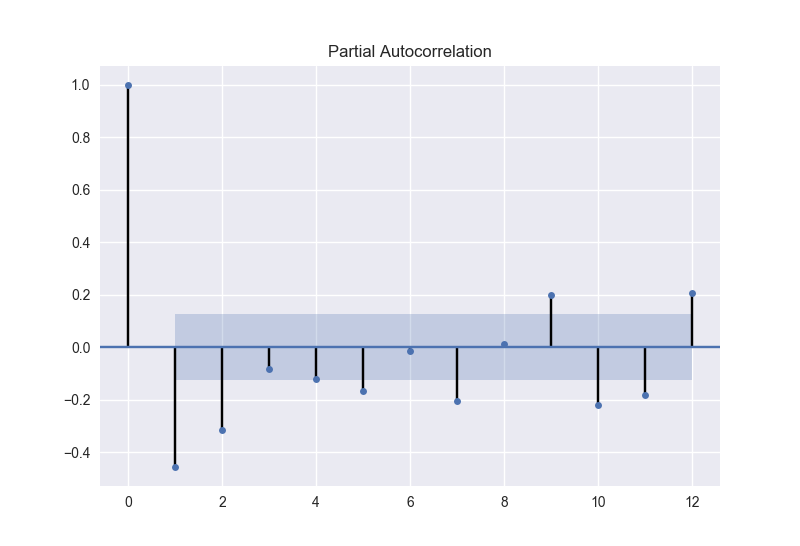
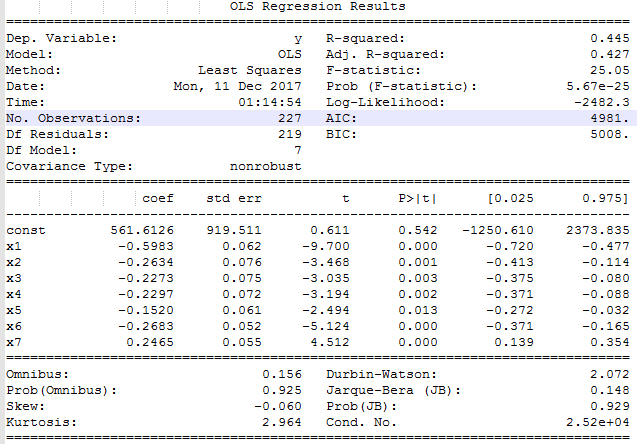
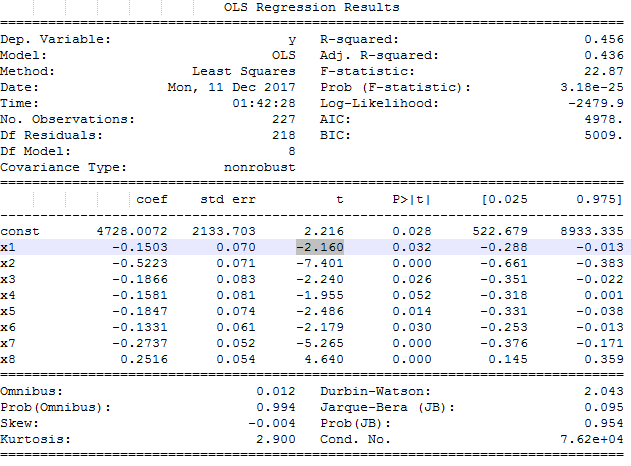
(recall that the estimation sample is Jan 1980 - Dec 1999).

1. Extend the model of part (d) by adding the Error Correcion (EC) term (*yt* −0.45*xt*), that is, estimate the ECM

*∆yt*  (estimation sample is Jan 1980

- Dec 1999). Check that the EC term is significant at the 5% level, but not at the 1% level.

1. Use the models of parts (d) and (e) to make two series of 12 forecasts of monthly changes in production of Toyota passenger cars in 2000. At each month, you should use the data that are then available, for example, to forecast production for September 2000 you can use the data up to and including August 2000. However, do not re-estimate the model and use the coefficients as obtained in parts (d) and (e). For each of the two forecast series, compute the values of the root mean squared error (RMSE) and of the mean absolute error (MAE). Finally, give your interpretation of the outcomes.  
     
   ========================================  
   Answers:  
     
   (a) This is the time series plots of yt (TOYOTA\_SA) and xt (OTHER\_SA).  
     
     
     
     
   As shown in the chart, the y and x have no obvious trend, but they may be co-integrated.

The following the plot of share of Toyota. It looks that the share% changed over time.  
  
  
(b)   
 (1) ADF test fot y(t),   
 dy = b0 + b1\*y1 + b2\*dy1 + b3\*dy2 + b4\*dy3 + eps  
 coeff of y(t-1): -0.0832, se = 0.037, t-value = -2.262  
 ADF critical value = -2.9  
 -2.262 > -2.9, so y(t) is non-staionary.  
  
 (2) ADF test fot x(t),   
 dx = b0 + b1\*x1 + b2\*dx1 + b3\*dx2 + b4\*dx3 + eps  
 coeff of x(t-1): -0.0696, se = 0.033, t-value = -2.106  
 ADF critical value = -2.9  
 -2.106 > -2.9, so x(t) is non-staionary.  
  
(c) EG step 1:  
 y(t) = 26790 + 0.4516 x(t) + e(t)  
   
 EG step 2:  
 de(t) = 24.9917 – 0.293 e(t-1) - 0.2858 de(t-1) - 0.1416 de(t-2) - 0.0960 de(t-3)  
 t-value of e(t-1): -4.306  
 Critical value = -3.4  
 as -4.306 < -3.4 🡺 e(t) is stationary, so y(t) and x(t) is co-integrated.  
  
(d)  
 ACF for first 12 sample autocorrelation:  
   
  
 PACF for first 12 sample autocorrelation:  
   
  
 The above two charts shows that the lagged terms at lags 1 to 5, 10, 12 are significant.   
 The threshold is 2/math.sqrt(240-12) = 0.13.  
  
 The regression: dy(t) = 561.6126 - 0.5983 dy(t-1) - 0.2634 dy(t-2) - 0.2273 dy(t-3) - 0.2297 dy(t-4)  
 -0.1520 dy(t-5) - 0.2683 dy(t-10) + 0.2465 dy(t-12)  
  
  
  
(e)   
The regression result shows that ECM term’s t-value = -2.16.  
the critical value is 1.96 for 5% level and 2.58 for 1% level.  
-2.16 < -1.96, ECM term is significant for 5% level.  
-2.16 > -2.58, ECM term is not significant for 1% level.  
  


(f) The forecasting result:  
  
from (d):  
  
 RMSE1= 16991.8042351 MAE1= 14703.2259138  
  
from (e):  
  
  
  
 RMSE2= 18204.7593804 MAE2= 15555.668404  
  
The model in (d) has lower model error, since its RMSE and MAE are lower.  
Adding ECM term in the model does not help, so xt (other brands) does not have prective power.  
  
===============================================  
Python code:

**import** datetime  
**import** pandas **as** pd  
**import** numpy **as** np  
**import** statsmodels.api **as** sm  
**import** seaborn **as** sns  
**import** matplotlib.pyplot **as** plt  
**import** scipy.stats **as** ss  
**from** statsmodels.graphics.tsaplots **import** plot\_acf  
**from** statsmodels.graphics.tsaplots **import** plot\_pacf  
  
df = pd.read\_excel(**'TestExer6-CARS-round2.xlsx'**)  
  
df[**'time'**] = df[**'YYYY-MM'**].apply(datetime.datetime.strptime, args=(**'%YM%m'**,))  
df[**'y'**] = df[**'TOYOTA\_SA'**]  
df[**'x'**] = df[**'OTHER\_SA'**]  
df[**'share'**] = df[**'y'**]/(df[**'y'**]+df[**'x'**])  
  
df[**'y-1'**] = df[**'y'**].shift(1)  
df[**'dy'**] = df[**'y'**].diff()  
df[**'dy-1'**] = df[**'dy'**].shift(1)  
df[**'dy-2'**] = df[**'dy'**].shift(2)  
df[**'dy-3'**] = df[**'dy'**].shift(3)  
df[**'dy-4'**] = df[**'dy'**].shift(4)  
df[**'dy-5'**] = df[**'dy'**].shift(5)  
df[**'dy-10'**] = df[**'dy'**].shift(10)  
df[**'dy-12'**] = df[**'dy'**].shift(12)  
  
df[**'x-1'**] = df[**'x'**].shift(1)  
df[**'dx'**] = df[**'x'**].diff()  
df[**'dx-1'**] = df[**'dx'**].shift(1)  
df[**'dx-2'**] = df[**'dx'**].shift(2)  
df[**'dx-3'**] = df[**'dx'**].shift(3)  
  
df[**'ecm1'**] = df[**'y-1'**] - 0.45\*df[**'x-1'**]  
  
n = len(df)  
n2 = 12  
df1 = df.head(n-n2)  
df2 = df.tail(n2) *# df.iloc[-12:]*print(**"\n(a):"**)  
print(df.head())  
  
fig,ax = plt.subplots()  
  
**for** name **in** [**'y'**,**'x'**]:  
 ax.plot(df[**'time'**], df[name], label=name)  
  
ax.set\_xlabel(**"time"**)  
ax.set\_ylabel(**"monthly production"**)  
ax.legend(loc=**'best'**)  
*# plt.show()*df.plot(x=**'time'**, y=**'share'**)  
*# plt.show()  
  
# sns.set(color\_codes=True)  
# ax = sns.regplot(data=df, x='SATV', y='FGPA', marker='+')  
# plt.show()*print(**"\n(b):"**)  
print(**'(1)'**)  
  
df\_b = df1.iloc[4:]  
y = np.array(df\_b[**'dy'**])  
x = np.array(df\_b[[**'y-1'**, **'dy-1'**, **'dy-2'**, **'dy-3'**]])  
  
X = sm.add\_constant(x)  
model = sm.OLS(y, X)  
r = model.fit()  
print(**'Linear Regression: dy = b0 + b1\*y1 + b2\*dy1 + b3\*dy2 + b4\*dy3 + eps'**)  
print(r.summary())  
  
print(**'(2)'**)  
  
y = np.array(df\_b[**'dx'**])  
x = np.array(df\_b[[**'x-1'**, **'dx-1'**, **'dx-2'**, **'dx-3'**]])  
  
X = sm.add\_constant(x)  
model = sm.OLS(y, X)  
r = model.fit()  
print(**'Linear Regression: dx = b0 + b1\*x1 + b2\*dx1 + b3\*dx2 + b4\*dx3 + eps'**)  
print(r.summary())  
  
print(**'\n(c):'**)  
  
y = np.array(df1[**'y'**])  
x = np.array(df1[**'x'**])  
  
X = sm.add\_constant(x)  
model = sm.OLS(y, X)  
r = model.fit()  
print(**'Linear Regression: y = b0 + b1\*x + eps'**)  
print(r.summary())  
  
df1[**'yhat\_c'**] = list(r.fittedvalues)  
df1[**'e\_c'**] = list(r.resid)  
df1[**'e1\_c'**] = df1[**'e\_c'**].shift(1)  
df1[**'de\_c'**] = df1[**'e\_c'**].diff()  
df1[**'de1\_c'**] = df1[**'de\_c'**].shift(1)  
df1[**'de2\_c'**] = df1[**'de\_c'**].shift(2)  
df1[**'de3\_c'**] = df1[**'de\_c'**].shift(3)  
  
*# print(df1.head())*df\_c = df1.iloc[4:]  
  
y = np.array(df\_c[**'de\_c'**])  
x = np.array(df\_c[[**'e1\_c'**, **'de1\_c'**, **'de2\_c'**, **'de3\_c'**]])  
  
X = sm.add\_constant(x)  
model = sm.OLS(y, X)  
r = model.fit()  
print(**'Linear Regression: de = b0 + b1\*e1 + b2\*de1 + b3\*de2 + b4\*de3 + eps'**)  
print(r.summary())  
  
print(**'\n(d):'**)  
*# print(df1.head())*df\_d = df1.iloc[1:]  
plot\_acf(df\_d[**'dy'**], lags=12)  
  
plot\_pacf(df\_d[**'dy'**], lags=12)  
*# plt.show()  
  
# print(df1.head(15))*df\_d = df1.iloc[13:]  
  
y = np.array(df\_d[**'dy'**])  
x = np.array(df\_d[[**'dy-1'**, **'dy-2'**, **'dy-3'**, **'dy-4'**, **'dy-5'**, **'dy-10'**, **'dy-12'**]])  
  
X = sm.add\_constant(x)  
model = sm.OLS(y, X)  
r = model.fit()  
print(r.summary())  
  
*#########################*print(**'\n(f1):'**)  
  
df\_f = df.tail(12)  
*# print(df\_f)*y1 = np.array(df\_f[**'dy'**])  
x1 = np.array(df\_f[[**'dy-1'**, **'dy-2'**, **'dy-3'**, **'dy-4'**, **'dy-5'**, **'dy-10'**, **'dy-12'**]])  
  
X1 = sm.add\_constant(x1)  
y1\_hat = r.predict(X1)  
df\_f[**'dy1\_hat'**] = list(y1\_hat)  
RMSE1 = np.sqrt(np.dot(y1-y1\_hat, y1-y1\_hat)/12.0)  
MAE1 = sum(abs(y1-y1\_hat))/12.0  
print(**'RMSE1='**,RMSE1, **'MAE1='**, MAE1)  
  
fig,ax = plt.subplots()  
  
**for** name **in** [**'dy'**,**'dy1\_hat'**]:  
 ax.plot(df\_f[**'time'**], df\_f[name], label=name)  
  
ax.set\_xlabel(**"time"**)  
ax.set\_ylabel(**"dy"**)  
ax.legend(loc=**'best'**)  
  
print(**'\n(e):'**)  
  
*# print(df1.head(15))*df\_e = df1.iloc[13:]  
  
y = np.array(df\_e[**'dy'**])  
x = np.array(df\_e[[**'ecm1'**, **'dy-1'**, **'dy-2'**, **'dy-3'**, **'dy-4'**, **'dy-5'**, **'dy-10'**, **'dy-12'**]])  
  
X = sm.add\_constant(x)  
model = sm.OLS(y, X)  
r = model.fit()  
print(r.summary())  
  
*################################*print(**'\n(f2):'**)  
  
y2 = np.array(df\_f[**'dy'**])  
x2 = np.array(df\_f[[**'ecm1'**, **'dy-1'**, **'dy-2'**, **'dy-3'**, **'dy-4'**, **'dy-5'**, **'dy-10'**, **'dy-12'**]])  
  
X2 = sm.add\_constant(x2)  
y2\_hat = r.predict(X2)  
df\_f[**'dy2\_hat'**] = list(y2\_hat)  
*# plt.plot(y2, y2\_hat)  
# plt.show()*RMSE2 = np.sqrt(np.dot(y2-y2\_hat, y2-y2\_hat)/12.0)  
MAE2 = sum(abs(y2-y2\_hat))/12.0  
print(**'RMSE2='**,RMSE2, **'MAE2='**, MAE2)  
  
fig,ax = plt.subplots()  
  
**for** name **in** [**'dy'**,**'dy2\_hat'**]:  
 ax.plot(df\_f[**'time'**], df\_f[name], label=name)  
  
ax.set\_xlabel(**"time"**)  
ax.set\_ylabel(**"dy"**)  
ax.legend(loc=**'best'**)  
plt.show()