

Econometrics III

Assignment Part 3, 4, 5

Tinbergen Insitute

Stanislav Avdeev
590050sa
stnavdeev@gmail.com

Bas Machielsens
590049bm
590049bm@eur.nl

March 29, 2021

Question 1

Part 1:

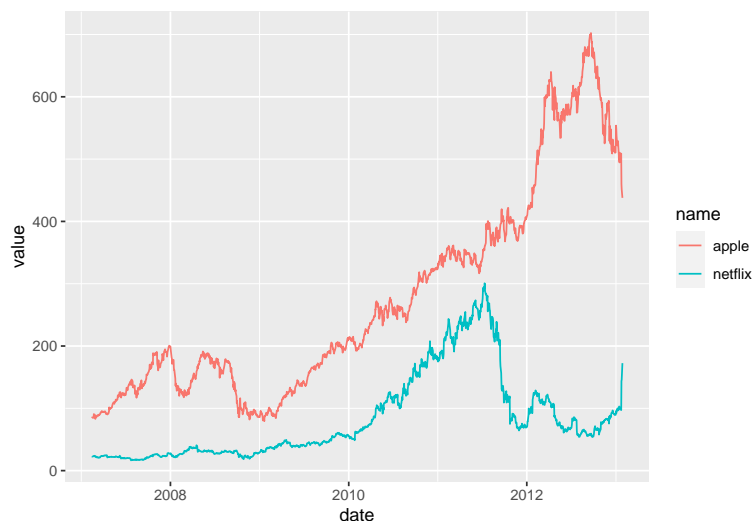
First, we plot the two time series:

```
df3 <- readr::read_csv("./data/data_assign_p3.csv")

twostocks <- c("apple", "netflix")

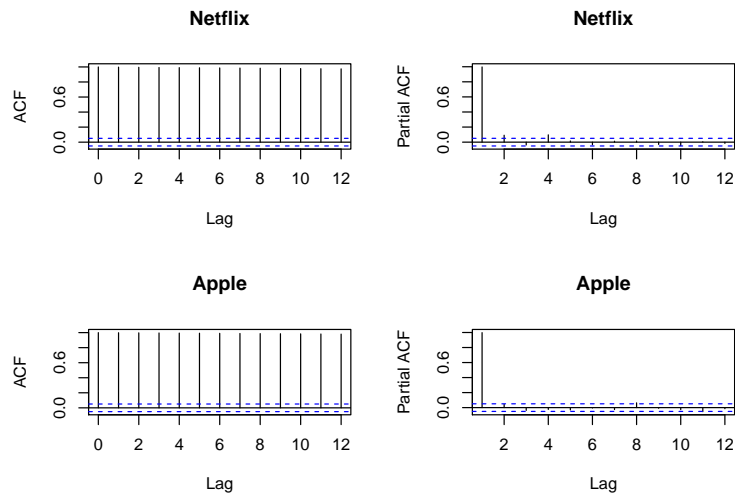
df3_twostocks <- df3 %>%
  janitor::clean_names() %>%
  pivot_longer(-date) %>%
  filter(is.element(name, twostocks)) %>%
  mutate(date = lubridate::dmy(date))

df3_twostocks %>%
  ggplot(aes(x = date, y = value,
             group = name, color = name)) + geom_line()
```



Then, we show the acf and pacf-functions:

```
n1 <- df3$NETFLIX %>%  
  acf(lag.max = 12, plot = F)  
  
n2 <- df3$NETFLIX %>%  
  pacf(lag.max = 12, plot = F)  
  
a1 <- df3$APPLE %>%  
  acf(lag.max = 12, plot = F)  
  
a2 <- df3$APPLE %>%  
  pacf(lag.max = 12, plot = F)  
  
par(mfrow=c(2,2))  
  
plot(n1, main = "Netflix");plot(n2, main = "Netflix")  
plot(a1, main = "Apple"); plot(a2, main = "Apple")
```



The ACF's tell us that the stock price is highly dependent on the past stock price, and this dependence decays only very slowly: even the 50 or 100-period lag still shows significant autocorrelation.

Part 2

We now implement a general to specific unit root test function:

```
unit_root_test <- function(column, order, critical_value){

  # Make the dataset
  series <- ts(column)
  first_differences <- diff(series, differences = 1)
  laggedvar <- stats::lag(series, -1)

  #other lagged first differences, delta x_{t-1}, ..., delta x_{t-p+1}
  lagged_fds <- list()

  for(i in 1:(order-1)){

    lagged_fds[[i]] <- stats::lag(first_differences, k = -i)

  }

  df <- cbind(first_differences, laggedvar, purrr::reduce(lagged_fds, cbind)) %>%
    as_tibble()

  colnames(df) <- c("dxt", "xtm1", paste("dxtm", 1:(order-1), sep = ""))

  df <- df %>%
    na.omit()

  # run the stepwise regression - find the best model
  null = lm(data = df, formula = "dxt ~ xtm1")
  full = lm(data = df, formula = paste("dxt ~ xtm1 +",
                                       paste(paste("dxtm", 1:(order-1), sep = ""),
                                             collapse = ' + '),
                                       collapse = " "))

  )

  bestmodel <- step(full,
    scope = list(lower = null, upper = full),
    direction = "backward",
    criterion = "BIC",
    k = log(nrow(df)))

  # perform the unit root test (MacKinnon, 2010)
  b_critical <- critical_value

  t_value <- bestmodel %>%
    summary() %>%
    .$coefficients %>%
    .[,3] %>%
    .["xtm1"]

  significant = abs(t_value) > abs(b_critical)

  data.frame(stock = deparse(substitute(column)),
    best_model = as.character(bestmodel$call[2]),
    t_value = t_value,
    sig = significant)

}
```

```
summary_tests <- list()

for(i in 1:length(colnames(df3[, -1]))){
  df <- unit_root_test(df3[, i+1], 12, -1.6156)
  summary_tests[[i]] <- df %>%
    mutate(name = colnames(df3[, -1][i]))
}

summary_tests <- summary_tests %>%
  purrr::reduce(rbind) %>%
  select(-stock)

rownames(summary_tests) <- NULL

knitr::kable(summary_tests)
```

best_model	t_value	sig	name
dxt ~ xtm1 + dxtm1 + dxtm3 + dxtm7	-0.6984454	FALSE	APPLE
dxt ~ xtm1 + dxtm1 + dxtm2	-2.0475011	TRUE	EXXON_MOBIL
dxt ~ xtm1	-1.1619768	FALSE	FORD
dxt ~ xtm1 + dxtm1 + dxtm3	-1.5455395	FALSE	GEN_ELECTRIC
dxt ~ xtm1	-2.2598273	TRUE	INTEL
dxt ~ xtm1	-2.3999971	TRUE	MICROSOFT
dxt ~ xtm1 + dxtm2	-1.0642387	FALSE	NETFLIX
dxt ~ xtm1 + dxtm2 + dxtm3	-0.6373731	FALSE	NOKIA
dxt ~ xtm1 + dxtm1	-1.3045653	FALSE	SP500
dxt ~ xtm1 + dxtm1 + dxtm6 + dxtm11	-2.6283893	TRUE	YAHOO

The test seems to be significant for a number of stocks, indicating that for these stocks, the null hypothesis of a unit root is rejected in favor of stationarity. With a 10% α -level, we expect to see a type I-error (rejecting the null while it is true) about one tenth of the time for every test. This means that the probability of having at least 1 type-I error is very large: $(1 - 0.9^{10}) = 0.6513216$. It would be better to use some kind of Bonferroni correction to correct for these compounding type I-errors, but alternatively, we could also lower the α -level.

Part 3

The forecast $\mathbb{E}[P_{t+1}|D_t] = \mathbb{E}[P_{t+1}|P_t] = \mathbb{E}[P_t + \epsilon_t] = P_t$. Similarly, the forecast $\mathbb{E}[P_{t+2}] = \mathbb{E}[P_{t+1} + \epsilon_{t+1}] = \mathbb{E}[P_{t+1}] = P_t$. Generalizing this pattern, the forecast for $P_{t+h} = P_t$. The variance of the forecast is derived using the distribution:

$$\begin{aligned}
 P_{t+1} &= P_t + \epsilon_t \Rightarrow P_{t+1}|P_t \sim N(P_t, \sigma^2) \\
 P_{t+2} &= P_{t+1} + \epsilon_{t+1} = \\
 &= P_t + \epsilon_t + \epsilon_{t+1} \Rightarrow P_{t+2}|P_t \sim N(P_t, 2\sigma^2)
 \end{aligned}$$

Generalizing this pattern, we can see that $\text{Var}(P_{t+h}) = h \cdot \sigma^2$. Hence, we can implement our forecasts in the following way:

```
#forecast code
p_tplush <- df3 %>%
  slice_tail(n=1) %>%
```

```

select(c("APPLE", "MICROSOFT"))

var_apple <- lm(data = df3 %>%
  select("APPLE"),
  formula = APPLE ~ lag(APPLE, 1)) %>%
  .$residuals %>%
  var()

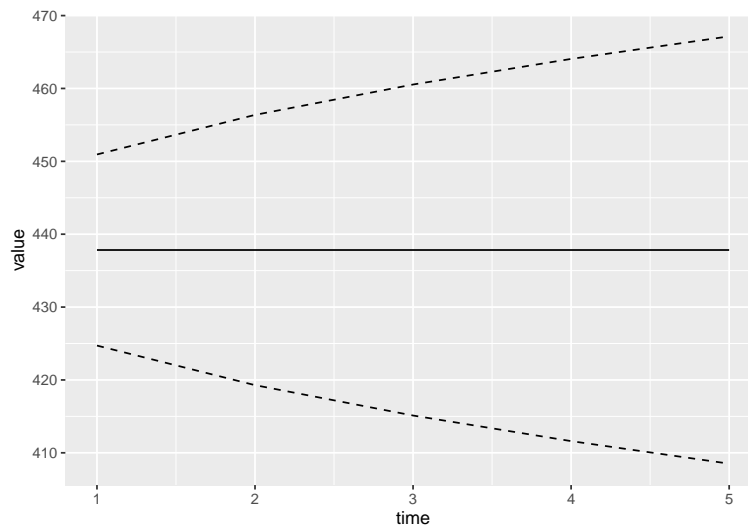
var_microsoft <- lm(data = df3 %>%
  select("MICROSOFT"),
  formula = MICROSOFT ~ lag(MICROSOFT, 1)) %>%
  .$residuals %>%
  var()

forecasts_apple <- data.frame(time = 1:5,
  value = rep(p_tplush %>%
    pull(1), 5),
  var = var_apple * 1:5)

forecasts_microsoft <- data.frame(time = 1:5,
  value = rep(p_tplush %>%
    pull(2), 5),
  var = var_microsoft * 1:5)

forecasts_apple %>%
  ggplot(aes(x = time)) +
  geom_line(aes(y = value)) +
  geom_line(aes(y = value + 1.96*sqrt(var)), lty = "dashed") +
  geom_line(aes(y = value - 1.96*sqrt(var)), lty = "dashed")

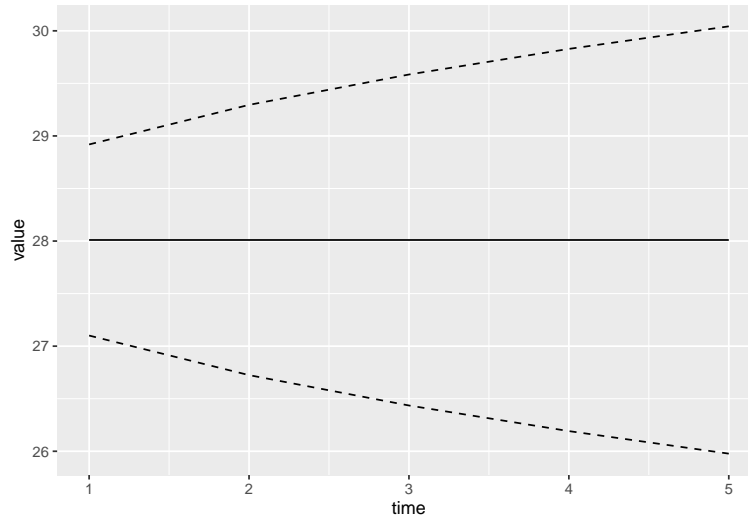
```



```

forecasts_microsoft %>%
  ggplot(aes(x = time)) +
  geom_line(aes(y = value)) +
  geom_line(aes(y = value + 1.96*sqrt(var)), lty = "dashed") +
  geom_line(aes(y = value - 1.96*sqrt(var)), lty = "dashed")

```



Hence, there is no investment advice that we can give: the value is predicted to remain constant, and for all predictions, the probability of an increase in stock price equals the probability of a decrease in stock price. The expected value of any investment strategy is 0, and the value is neither expected to increase, nor to decrease.

Part 4

Do you find a statistically significant contemporaneous relation between Microsoft and Exxon Mobile stock prices?

```
lm(df3, formula = MICROSOFT ~ EXXON_MOBIL) %>%
  stargazer(header = F, omit.stat = c("adj.rsq", "ser", "f"))
```

Table 2:

<i>Dependent variable:</i>	
MICROSOFT	
EXXON_MOBIL	0.203*** (0.009)
Constant	11.245*** (0.713)
Observations	1,499
R ²	0.251
<i>Note:</i> *p<0.1; **p<0.05; ***p<0.01	

Do you agree that changes in Microsoft stock prices are largely explained by fluctuations in the stock price of Exxon Mobile?

We do not agree that changes in Microsoft stock prices are largely explained by fluctuations in the stock price of Exxon Mobile. It is likely that these results are driven by a shared stochastic trend in both of variables, in other words, the variables might be cointegrated. It can be shown that if two variables share a stochastic trend, the t-value of the estimated coefficient tends to infinity, and the probability of obtaining statistical significance to 1, even under the assumption of completely unrelated trends.

Question 4

Part 1

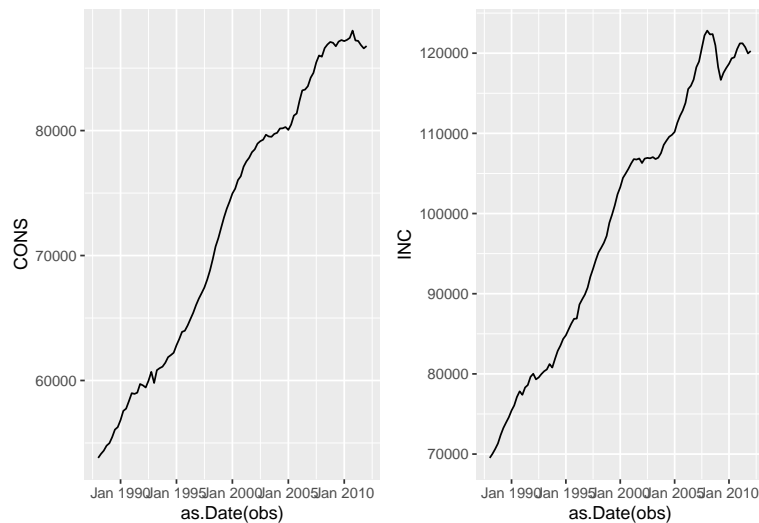
```
df4 <- read_csv("data/data_assign_p4.csv")

df4$obs <- ts(df4$obs,
              frequency = 4,
              start = c(1988, 1))

p_cons <- ggplot(data = df4, aes(x = as.Date(obs), y = CONS)) +
  geom_line() +
  scale_x_date(date_labels = "%b %Y")

p_inc <- ggplot(data = df4, aes(x = as.Date(obs), y = INC)) +
  geom_line() +
  scale_x_date(date_labels = "%b %Y")

cowplot::plot_grid(p_cons, p_inc)
```

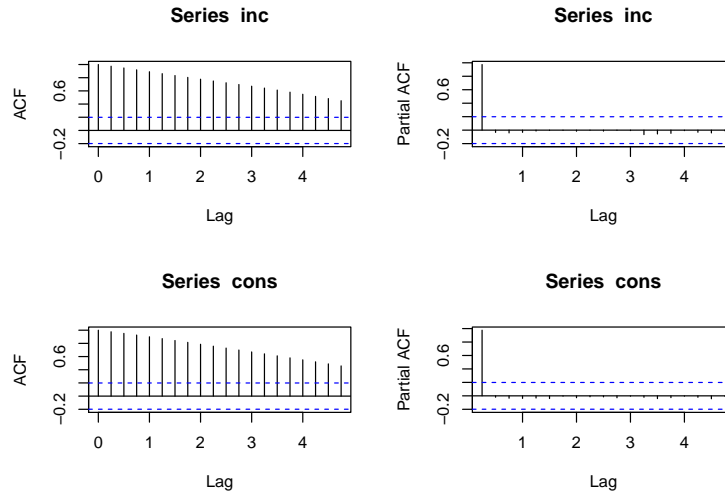


```
inc <- ts(df4$INC,
          frequency = 4,
          start = c(1988, 1))

cons <- ts(df4$CONS,
           frequency = 4,
           start = c(1988, 1))

inc_acf <- acf(inc, plot = F); inc_pcf <- pacf(inc, plot = F)
cons_acf <- acf(cons, plot = F); cons_pcf <- pacf(cons, plot = F)

par(mfrow=c(2,2))
plot(inc_acf); plot(inc_pcf); plot(cons_acf); plot(cons_pcf)
```



We observe that in both cases, the autocorrelation function shows extremely high estimates for each subsequent lag: even for a large number of periods, there is still a large degree of autocorrelation in the data. The partial autocorrelations, however, show a completely different view: there is no partial autocorrelation when controlled for other influences. This means that there is a lot of dependence in the series unconditionally, but conditionally on previous values, there seems to be no correlation. This behavior seems to be consistent with random walk behavior.

Part 2

We use the same function as in the previous question to find the best model (according to BIC), and compute the t-value and conduct a Dickey-Fuller test:

```
urcons <- unit_root_test(df4$CONS, order = 12, -1.9393)
urinc <- unit_root_test(df4$INC, order = 12, -1.9393)

unitroots <- rbind(urcons, urinc) %>%
  mutate(stock = str_replace(stock, "df4\\$", ""))

rownames(unitroots) <- NULL

knitr::kable(unitroots)
```

stock	best_model	t_value	sig
CONS	dxt ~ xtm1 + dxtm3	-1.619306	FALSE
INC	dxt ~ xtm1 + dxtm1	-1.169998	FALSE

As we can see in the table, the unit root hypothesis is not rejected at a 5% level in both cases.

Part 3

Now, we perform a unit root test on the first differences of both series:

```
urcons2 <- unit_root_test(diff(df4$CONS), order = 12, -1.9393)
urinc2 <- unit_root_test(diff(df4$INC), order = 12, -1.9393)

unitroots2 <- rbind(urcons2, urinc2) %>%
  mutate(stock = str_replace(stock, "df4\\$", ""))
```



```
rownames(unitroots2) <- NULL
```

```
knitr::kable(unitroots2)
```

stock	best_model	t_value	sig
diff(CONS)	dxt ~ xtm1 + dxtm1 + dxtm2	-2.382383	TRUE
diff(INC)	dxt ~ xtm1 + dxtm8	-5.231291	TRUE

Now, we see that both hypotheses are rejected! We conclude that both of these series are integrated at order $I(1)$, because the hypothesis of a unit root in the first differences is rejected.

Part 4

Assuming both series are $I(1)$, test for cointegration between consumption and income by regressing consumption on income and performing a unit-root test on the residuals. Report the estimated regression coefficients.

Plot the regression residuals. Use the Schwartz Information Criterion (SIC) to determine the number of ADF lags in your unit-root residual test. Report the cointegration test statistic. Do you reject cointegration?

```
consinreg <- lm(data = df4,
  formula = CONS ~ INC)

consinreg %>%
  stargazer(header = F,
    omit.stat = c("adj.rsq", "ser", "f"))
```

Table 5:

	<i>Dependent variable:</i>
	CONS
INC	0.665*** (0.006)
Constant	6,783.373*** (554.991)
Observations	97
R ²	0.993
Note:	*p<0.1; **p<0.05; ***p<0.01

```
urt2 <- unit_root_test(consinreg$residuals,
  order = 12,
  -1.6156)
```

```
rownames(urt2) <- NULL
```

```
knitr::kable(urt2)
```

stock	best_model	t_value	sig
consinreg\$residuals	dxt ~ xtm1 + dxtm2 + dxtm8	-2.127911	TRUE

We observe that the hypothesis of a unit root in the residuals is rejected, indicating that the two series are cointegrated: $Cons, Inc \sim CI(1, 1)$.

Part 5

As the series are cointegrated, we can estimate an error correction model, incorporate long-run equilibrium information as well as short-run dynamics. The error-correction model is of the following general form:

$$\Delta Y_t = \alpha - \gamma \bar{Z}_{t-1} + \phi_1 \Delta Y_{t-1} + \dots + \phi_p \Delta Y_{t-p} + \beta_0 \Delta X_t + \dots + \beta_q \Delta X_{t-q} + u_t$$

We first construct the necessary data matrix in Python:

```
import pandas as pd

# p for dep var, q for indep var
def generate_datamatrix(p, q, y, x):
    # dependent variable
    fdy = r.df4[str(y)] - r.df4[str(y)].shift(1)
    # ytm1
    ytm1 = r.df4[str(y)].shift(1)
    # xtm1
    xtm1 = r.df4[str(x)].shift(1)

    d = {}

    d['xt'] = r.df4[str(x)]
    d['xtm1'] = xtm1
    # x first differences
    d['dxtm0'] = r.df4[str(x)] - r.df4[str(x)].shift(1)
    # lagged first differences
    for i in range(1, q+1):
        d['dxtm{0}'.format(i)] = d['dxtm0'].shift(i)

    # y first differences
    e = {}

    e['yt'] = r.df4[str(y)]
    e['fdy'] = fdy
    e['ytm1'] = ytm1

    for j in range(1, p+1):
        e['dytm{0}'.format(j)] = e['fdy'].shift(j)

    # put everything in a dataframe
    dfx = pd.DataFrame.from_dict(d)
    dfy = pd.DataFrame.from_dict(e)

    df_out = pd.concat([dfx.reset_index(drop=True), dfy], axis=1)

    return df_out

data_ecm = generate_datamatrix(4, 4, 'CONS', 'INC')
```

Now, we implement the general-to-specific procedure in R. First, we estimate the residuals \bar{Z}_t , and compute \bar{Z}_{t-1} which we then lag to add as a predictor in the general-to-specific procedure.

```
zt <- lm(data = py$data_ecm,
         formula = yt ~ xt)$residuals

ztm1 <- lag(residuals, 1)
```

Now, we add \bar{Z}_{t-1} to the dataset, and formulate the g2s procedure (again according to the BIC):

```
data_ecm <- py$data_ecm %>%
  mutate(ztm1 = ztm1) %>%
  na.omit()

indep_vars <- py$data_ecm %>%
  colnames()

indep_vars <- indep_vars[grepl("dxt|dyt", indep_vars)]

longformula <- paste("fdy ~ ztm1 +", paste(indep_vars, collapse = " + "))
# run the stepwise regression - find the best model
null = lm(data = data_ecm, formula = "fdy ~ ztm1")
full = lm(data = data_ecm, formula = longformula)

bestmodel <- step(full,
  scope = list(lower = null, upper = full),
  direction = "backward",
  criterion = "BIC",
  k = log(nrow(data_ecm)))
```

Starting with $p = q = 4$, we get the following model:

```
stargazer(bestmodel, header = F,
  omit.stat = c("adj.rsq", "ser", "f"))
```

Report and interpret the short-run and long-run multipliers. Report and interpret the error correction coefficient.

6. How strong is the correction to equilibrium? Is there over-shooting? Do you find evidence of Granger causality? Justify your answer.

Table 7:

<i>Dependent variable:</i>	
	fdy
ztm1	−0.120*** (0.034)
dxtm0	0.179*** (0.044)
dytm3	0.282*** (0.093)
Constant	147.246*** (46.319)
Observations	92
R ²	0.408
<i>Note:</i> *p<0.1; **p<0.05; ***p<0.01	