Econometrics III
Assignment Part 1 & 2
Tinbergen Insitute

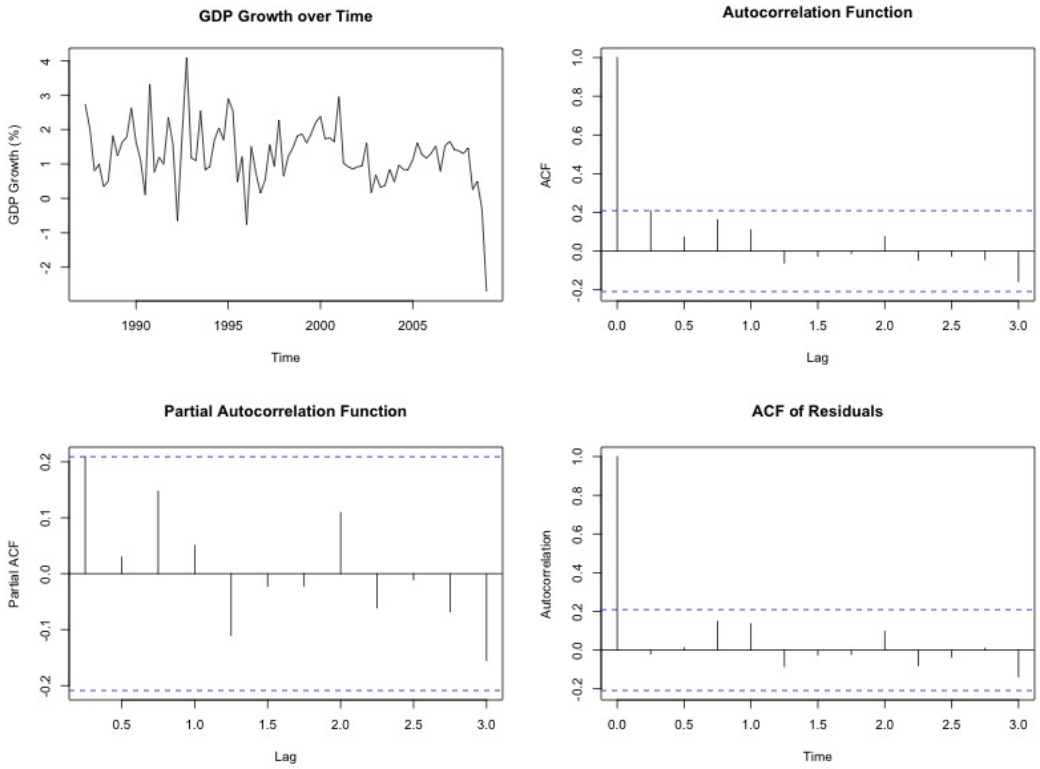
Stanislav Avdeev
590050sa
stnavdeev@gmail.com

Bas Machielsen
590049bm
590049bm@eur.nl

March 12, 2021

Question 1

Part (a)




```
# Part 1: Plot the Dutch GDP, ACF, and PACF
df <- readr::read_csv("./data/data_assign_p1.csv")

df <- ts(df$GDP_QGR,
```

```
         frequency = 4,
         start = c(1987, 2))
```

In the above figure, upper left, we observe the growth rates of Dutch GDP from 1987 to 2009.

```
autoc <- acf(df,
    lag.max = 12, plot = F)

Box.test(df,
         lag = 12,
         type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  df
## X-squared = 12.106, df = 12, p-value = 0.4372
```

```
pautoc <- pacf(df,
     lag.max = 12, plot = F)
```

In the above figure, we also plot the ACF and PACF, which is the ACF controlled for the other lagged correlations. The ACF tells us that the correlation of GDP growth with its lags is very low - hinting at very little time-dependence in this time-series. More precisely, the estimated correlation coefficients are not higher than 0.2 (for the lag of 1 period). If the GDP is indeed generated by an $AR(p)$ process, the estimates show that the process has low ϕ's (in absolute value), indicating a low time dependence.

Part (b)

```
ar4 <- dynlm(df ~ L(df, 1) + L(df, 2) + L(df, 3) + L(df, 4))
ar3 <- dynlm(df ~ L(df, 1) + L(df, 3) + L(df, 4))
ar2 <- dynlm(df ~ L(df, 1) + L(df, 3))
ar1 <- dynlm(df ~ L(df, 1))

ar1_m <- forecast::Arima(df, c(1,0,0), method = "CSS")

stargazer(ar4, ar3, ar2, ar1,
          type = "latex", header = FALSE,
          font.size = "small", column.sep.width = "0pt",
          omit.stat = c("adj.rsq", "ser", "f"))
```

The table below shows us the coefficients of the $AR(p)$ models with 1 to 4 lags included. Using a general-to-specific approach, we eliminate insignificant lags ($\alpha = 0.05$ level) at each step. Column (1) shows that only the first lag is significant at the 0.1 level. The second lag has the lowest ψ coefficient and the highest s.e., thus, we eliminate it first. Doing this iteratively, we are left with the first lag in the model which is significant at the 0.05 level. Investigating the coefficient, we observe there is a 1-period persistence of the time series on its lagged value. The *degree* of persistence, however, is not large, as the magnitude of the coefficient is fairly small (0.26).

Part c

In the fourth plot above, we show the autocorrelation of the residuals. We observe that none of the coefficients attain significance. Hence, we think that the model is well-specified. Formally, we compute the Durbin-Watson test statistic, of which the output is shown below. The null hypothesis of no autocorrelation is not

Table 1:

	Dependent variable:			
	df			
	(1)	(2)	(3)	(4)
L(df, 1)	0.232*	0.240*	0.257**	0.267**
	(0.124)	(0.122)	(0.119)	(0.117)
L(df, 2)	0.055			
	(0.126)			
L(df, 3)	0.203	0.210*	0.210*	
	(0.126)	(0.124)	(0.120)	
L(df, 4)	0.094	0.092		
	(0.125)	(0.124)		
Constant	0.479	0.533*	0.631***	0.896***
	(0.299)	(0.271)	(0.238)	(0.181)
Observations	84	84	85	87
R^2	0.099	0.097	0.089	0.058
Note:			*p<0.1; **p<0.05; ***p<0.01	

rejected. In an unreported plot, we also investigate the *partial* autocorrelation of residuals, in which there is also no sign of autocorrelation.

```
durbinWatsonTest(ar1)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1     -0.02081405       1.85759   0.488
##  Alternative hypothesis: rho != 0
```

```
autocred <- acf(ar1_m$resid, 12,
                plot = FALSE)
```

Part d

```
# Part 4: Forecast AR model for 2 years
df_pred <- predict(ar1_m, n.ahead = 8)$pred
```

We derived the forecasts by making use of the conditional expectation as the forecast minimizing the forecast error under quadratic loss. The conditional expectation of the forecast for $T+1$ is: $\mathbb{E}[X_{T+1}|X_1, \dots, X_T] = \mathbb{E}[\phi X_T + \alpha + \epsilon_{T+1}|X_1, \dots, X_T] = \phi X_T + \alpha$. Similarly, the forecast for $T+2$ equals $\phi\mathbb{E}[X_{T+1} + \alpha] = \phi(\alpha + \phi X_T) + \alpha$. Generalizing this pattern, we then end up with the following recursive forecasts (for $j > 1$):

$$\mathbb{E}[X_{T+j}] = \alpha + \alpha \cdot \left(\sum_{i=1}^{j-1} \phi^i \right) + \phi^j X_T$$

Part e

We construct the forecasts using a 95% symmetric confidence interval around the forecast.

```r
# Part 5: Produce CI
phi <- ar1_m$coef[1]
sigma <- sqrt(ar1_m$sigma2)

df_ciu <- predict(ar1_m, n.ahead = 8)$pred + predict(ar1_m, n.ahead = 8)$se*1.96
df_cil <- predict(ar1_m, n.ahead = 8)$pred - predict(ar1_m, n.ahead = 8)$se*1.96
```

We can reproduce the forecasts manually by using a standard confidence interval around the aforementioned predicted value, $CI(\mathbb{E}[X_{T+j}]) = \alpha + \alpha \cdot \left(\sum_{i=1}^{j-1} \phi^i\right) + \phi^j X_T \pm 1.96 \cdot \sigma$, where $\sigma_j^2 = \sigma_\epsilon^2 \cdot (1 + \phi^2 + \cdots + \phi^{2(j-1)})$.

Part f

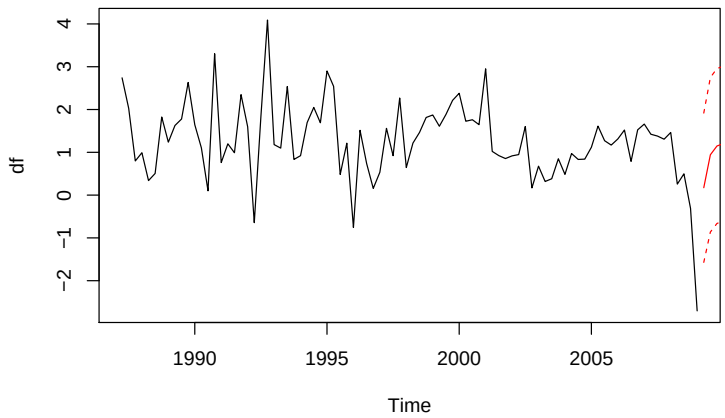
We check for normality of the residuals using the Jarque-Bera test:

```r
jb.norm.test(ar1_m$resid)
```

```
##
##  Jarque-Bera test for normality
##
## data:  ar1_m$resid
## JB = 31.722, p-value = 0.001
```

According to the test, we have to reject H0, indicating that normality of the innovations is violated. In the figure below, we plot the forecasts (red) next to the actual data (blue).

```r
true <- ts(c(0.49, -0.31, -2.7, -1.63, 0.28, 0.33, 0.66, 1.59, 0.51, 0.71, 0.81),
         frequency = 4,
         start = c(2009, 2))
ts.plot(df)
#points(true, type = "l", col = 'blue', lty = 1)
points(df_pred, type = "l", col = 'red', lty = 1)
points(df_ciu, type = "l", col = 'red', lty = 2)
points(df_cil, type = "l", col = 'red', lty = 2)
```



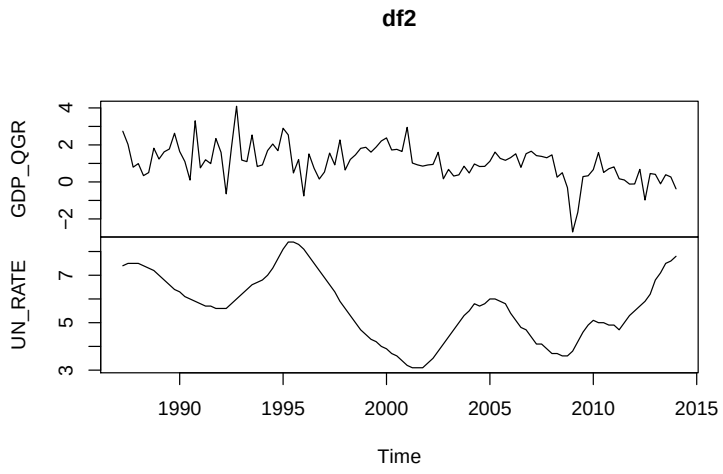
Question 2

```
# Get the data
df2 <- readr::read_csv("./data/data_assign_p2.csv")

df2 <- ts(df2[,2:3],
          frequency = 4,
          start = c(1987, 2))
```

Part (a)

```
# Part a: Plot the Dutch GDP/unemployment, AR and ADL
plot(df2)
```




```
ar4 <- dynlm(df2[,1] ~ L(df2[,1], 1) + L(df2[,1], 2) + L(df2[,1], 3) + L(df2[,1], 4))
ar3 <- dynlm(df2[,1] ~ L(df2[,1], 1) + L(df2[,1], 3) + L(df2[,1], 4))
ar2 <- dynlm(df2[,1] ~ L(df2[,1], 1) + L(df2[,1], 3))

# Create lags
library(data.table)
df_lags <- setDT(as.data.frame(df2[,1]))[, paste("x", 1:4) := shift(x, 1:4)][]
df_lags2 <-  setDT(as.data.frame(df2[,2]))[, paste("x", 1:4) := shift(x, 1:4)][]
df_lags <- cbind(df_lags, df_lags2)
df_lags <- df_lags[,c(1:5, 7:10)]
colnames(df_lags) <- as.character(c(0:8))

# To check whether we can use lm and not dynlm with lags
lm_1 <- lm(df_lags$`0` ~ df_lags$`1` + df_lags$`2` + df_lags$`3` + df_lags$`4` + df_lags$`5` + df_lags$

adl8 <- dynlm(df2[,1] ~ L(df2[,1], 1) + L(df2[,1], 2) + L(df2[,1], 3) + L(df2[,1], 4) + L(df2[,2], 1) +

stargazer(adl8, lm_1, type = "text", header = FALSE)
```

```
##
```

```
## ============================================================
##                                    Dependent variable:
##                                ---------------------------
##                                   df2[, 1]         `0`
##                                   dynamic          OLS
##                                    linear
##                                     (1)             (2)
## ------------------------------------------------------------
## L(df2[, 1], 1)                    0.305***
##                                   (0.101)
##
## L(df2[, 1], 2)                     0.078
##                                   (0.105)
##
## L(df2[, 1], 3)                    0.218**
##                                   (0.104)
##
## L(df2[, 1], 4)                     0.010
##                                   (0.102)
##
## L(df2[, 2], 1)                     0.620
##                                   (0.713)
##
## L(df2[, 2], 2)                     -0.088
##                                   (1.326)
##
## L(df2[, 2], 3)                     -1.931
##                                   (1.335)
##
## L(df2[, 2], 4)                    1.502**
##                                   (0.717)
##
## `1`                                              0.305***
##                                                   (0.101)
##
## `2`                                               0.078
##                                                   (0.105)
##
## `3`                                              0.218**
##                                                   (0.104)
##
## `4`                                               0.010
##                                                   (0.102)
##
## `5`                                               0.620
##                                                   (0.713)
##
## `6`                                              -0.088
##                                                   (1.326)
##
## `7`                                              -1.931
##                                                   (1.335)
##
## `8`                                              1.502**
```

```
##                                              (0.717)
##
## Constant                                    -0.184        -0.184
##                                            (0.398)        (0.398)
##
## ------------------------------------------------------------------
## Observations                                 104            104
## R2                                          0.282          0.282
## Adjusted R2                                 0.222          0.222
## Residual Std. Error (df = 95)               0.859          0.859
## F Statistic (df = 8; 95)                  4.674***       4.674***
## ==================================================================
## Note:                                 *p<0.1; **p<0.05; ***p<0.01
```

```
# yes, we can
```