Danmarks Tekniske Universitet DTU

Assignment 2: ARMA Processes and Seasonal Processes

02417 - Time Series Analysis

GROUP

Pavlou, Ioannis - s212858
Blachet, Apolline - s222903
Kapakoglou, Georgios - s223001
Kozaris, Charalampos - s230224

January 20, 2024

Contents

List of Figures **i**

List of Tables **i**

Question 2.1: Stability **1**

Question 2.2: Predicting the number of sales of apartments **5**

Question 2.3: ARMA Process **8**

Code **11**

List of Figures

1 Stochastic process illustration . . . 2
2 Autocorrelation function (ACF) for each realization . . . 3
3 Partial autocorrelation function (PACF) for each realization . . . 3
4 Sales of apartments over time in the capital region of Denmark . . . 7
5 Histogram plot of the estimates of parameter ϕ_2 for each process . . . 8
6 Estimated pairs of parameters (ϕ_1, ϕ_2) . . . 10
7 Autocorrelation function (ACF) for each realization (separate) . . . 18
8 Partial autocorrelation function (PACF) for each realization (separate) . . . 18

List of Tables

1 Variance of each of the realizations . . . 4
2 Predicted values for the first two quarters of 2019 . . . 6

Question 2.1: Stability

In the first section of the assignment we consider a X_t stochastic process defined by the following equation:

$$X_t - 0.8X_{t-1} = \epsilon_t + 0.8\epsilon_{t-1} - 0.5\epsilon_{t-2} \tag{1}$$

, where ϵ_t is a white noise process with $\sigma = 0.4$.

1) Firstly, it has to be determined whether the process is stationary and invertible. In order to do so, we examine if the roots of $\phi(z^{-1})$ and the roots of $\theta(z^{-1})$ with respect to z all lie within the unit circle, respectively.

The following code is utilized in "RStudio":

Listing 1: Question 2.1.1

```
root.phi = polyroot(rev(c(1, -0.8)))
root.theta = polyroot(rev(c(1, 0.8, -0.5)))
```

, and the result is that the process is stationary, but not invertible as the latter roots do not all lie within the unit circle:

$$roots(\phi(z^{-1})) = \begin{bmatrix}0.8\end{bmatrix} \quad \text{and} \quad roots(\theta(z^{-1})) = \begin{bmatrix}0.4124038 & -1.2124038\end{bmatrix}. \tag{2}$$

Thus, the distribution of the stochastic process does not change over time, but on the other hand a finite amount of observations cannot determine its state.

2) The analytical solution:

$$Y_{\epsilon_Y}(k) + \Phi_1 Y_{\epsilon_Y}(k-1) + \cdots + \Phi_p Y_{\epsilon_Y}(k-p) = \theta_k \sigma_\epsilon^2, k = 0, 1, \ldots \tag{eq:5.97}$$

$$\begin{aligned} k = 0: \quad & Y_{\epsilon_Y}(0) = \theta_0 \sigma_\epsilon^2 \\ k = 1: \quad & Y_{\epsilon_Y}(1) = \theta_1 \sigma_\epsilon^2 \\ k = 2: \quad & Y_{\epsilon_Y}(2) = \theta_2 \sigma_\epsilon^2 \end{aligned} \tag{3}$$

$$\begin{aligned} & Y_{\epsilon_Y}(0) = \sigma_\epsilon^2 = 0.16 \\ & Y_{\epsilon_Y}(1) = (\theta_1 - \Phi_1)\sigma_\epsilon^2 = (0.8 - (-0.8))\sigma_\epsilon^2 = 1.6 \cdot 0.16 = 0.256 \\ & Y_{\epsilon_Y}(2) = -\Phi_1 Y_{\epsilon_Y}(k-1) = 0.8 \cdot Y_{\epsilon_Y}(1) = 0.8 \cdot 1.6 \cdot 0.16 = 0.2048 \end{aligned} \tag{4}$$

$$\gamma(0) + \Phi_1 \gamma(1) = Y_{\epsilon_Y}(0) + \theta_1 Y_{\epsilon_Y}(1) + \theta_2 Y_{\epsilon_Y}(2) \tag{i}$$

$$\gamma(1) + \Phi_1\gamma(0) = \theta_1 Y_{\epsilon_Y}(0) + \theta_2 Y_{\epsilon_Y}(1) \quad \text{(ii)}$$

$$\gamma(2) = 0.8\gamma(1) \quad \text{(iii)}$$

$$\gamma(k) \quad = -\Phi_1\gamma(k-1) \quad (5)$$

By plugging in the values,

$(i) <=> \gamma(0) - 0.8\gamma(1) = 0.16 + 0.8 \cdot 1.6 \cdot 0.16 - 0.5 \cdot 0.8 \cdot 1.6 \cdot 0.16 = 0.2968$
$(i) <=> \gamma(0) - 0.8\gamma(1) = 0.2968$
$(ii) <=> \gamma(1) - 0.8\gamma(0) = 0.8 \cdot 0.16 - 0.5 \cdot 1.6 \cdot 0.16 = 0$
$(ii) <=> \gamma(1) = 0.8\gamma(0)$
$(ii) + (i) <=> \gamma(0) - 0.8 \cdot 0.8\gamma(0) = 0.2968$
$<=> 0.36 \cdot \gamma(0) = 0.2968$

So,
$\gamma(0) = 0.840$
$\gamma(1) = 0.672$
$\gamma(2) = 0.8 \cdot 0.672 = 0.4576$
$\gamma(k) = 0.8\gamma(k-1),$ for $k > 2$

Thus, the second order moment representation of the process, which is the variance of the process, $V[X_t]$, equals to $\gamma(0) = 0.84$.

3) In Figure 1 are illustrated 10 realizations (different color) of the stochastic process, with 200 observations each. As observed, the first and second order moment representation of the distribution do not depend on time, which again indicates the stationarity of the stochastic process.

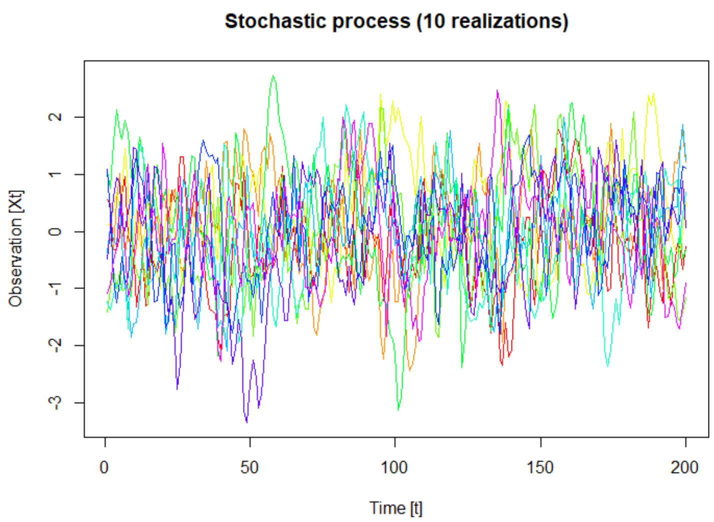

Figure 1: Stochastic process illustration

4) Estimations of the Auto-Correlation Functions of each realization were made and are illustrated in Figure 2. As expected from an ARMA (1, 2) process, the ACF plots show damped exponential and/or sine functions from lag $q + 1 - p = 2 + 1 - 1 = 2$. It can be observed that for some realizations the effect of the sine function is not visible in the ACF.

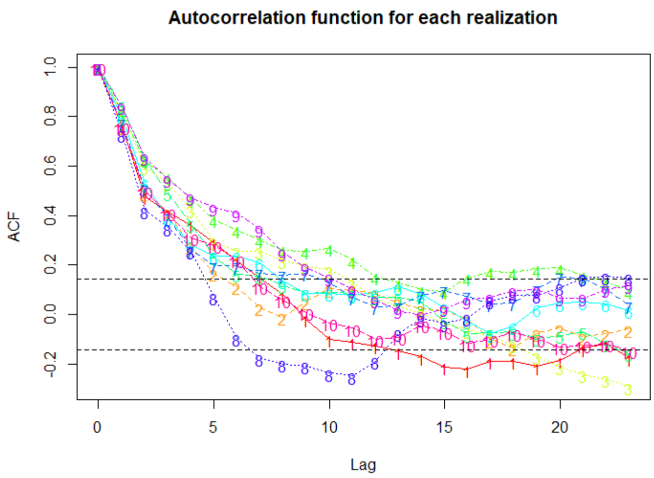

Figure 2: Autocorrelation function (ACF) for each realization

5) Estimations of the Partial Auto-Correlation Functions of each realization were made and are illustrated in Figure 3. The expected behaviour for an ARMA (1, 2) process is a plot dominated by damped exponential and/or sine functions from lag $max(0, p - q) = max(0, 1 - 2) = 0$. This behaviour can generally be observed in the PACF, with some estimations showing insignificant oscillation around zero for greater lags.

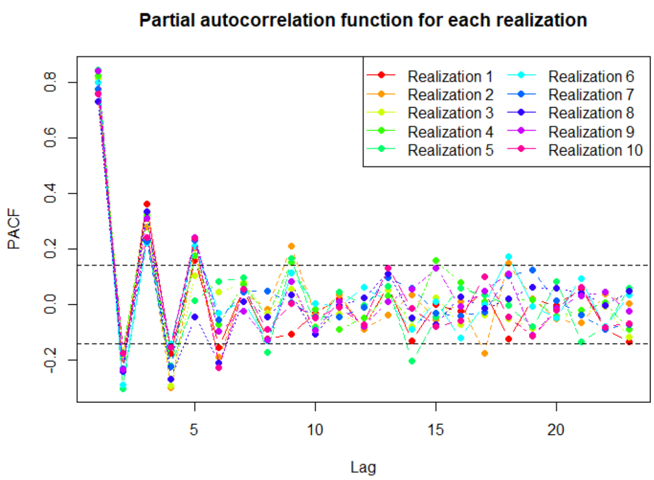

Figure 3: Partial autocorrelation function (PACF) for each realization

6) On Table 1 are included the calculated variances for each realization respectively (mean variance of all the realizations: 0.843).

Table 1: Variance of each of the realizations

Realization	1	2	3	4	5	6	7	8	9	10
Variance	0.699	0.767	0.747	0.925	1.197	0.930	0.638	0.665	1.052	0.813

7) When solving a problem analytically and numerically, there can be differences between the answers obtained from the two methods. These differences can occur when iterating through multiple realizations and then finding the mean of them, leading to slight variations in the final answer. More specifically, in our case, the analytical solution gave us a variance of 0.840 while the numerical solution gave a mean variance of 0.843 for 10 realizations of 200 observations, which indicates that the mean variance of a sufficiently large number of realizations will not be significantly different from the analytical solution. However, there are some individual variances that differ.

In general, the analytical solution of a problem involves mathematical equations that describe precisely the behavior of a process. In contrast, numerical solutions involve approximating the solution of the problem by performing calculations using numerical methods. Lastly, it is not uncommon to observe differences between the analytical and numerical solutions in iterative processes, especially when dealing with real-world data and not having enough observations.

Question 2.2: Predicting the number of sales of apartments

In this section of the assignment we aim to predict the number of sales of apartments in the capital region of Denmark. For this purpose, we utilize the already identified model of the quarterly number of sales, which is described by the following structure:

$$(1 - 1.04B + 0.2B^2)(1 - 0.86B^4)(Y_t - \mu) = (1 - 0.42B^4)\epsilon_t \tag{6}$$

, where ϵ_t is a white-noise process with variance $\sigma^2 = 36963$ and μ was estimated to be 2070 based on historical data. The training set is obtained by "A2_sales.txt" and is comprised by five years' worth of data for Y_t, which translates to 20 observations of sales.

1) The values of Y_t for t=2019Q1 and 2019Q2 are the ones that we want to predict, along with their 95% prediction intervals. The following sequence of thoughts is drafted on paper:

$$(6) \Leftrightarrow (1 - 1.04B + 0.2B^2 - 0.86B^4 + 0.8944B^5 - 0.172B^6)(Y_t - \mu) = (1 - 0.42B^4)\epsilon_t \tag{7}$$

$$\Leftrightarrow (1 + \Phi_1 B + \Phi_2 B^2 + \Phi_4 B^4 + \Phi_5 B^5 + \Phi_6 B^6)(Y_t - \mu) = (1 + \Theta_4 B^4)\epsilon_t \tag{8}$$

with $\Phi_1 = -1.04, \Phi_2 = 0.2...$
Let be $Z_t = Y_t - \mu$, then we get:

$$Z_t = -\Phi_1 Z_{t-1} - \Phi_2 Z_{t-2} - \Phi_4 Z_{t-4} - \Phi_5 Z_{t-5} - \Phi_6 Z_{t-6} - \epsilon_t + \Theta_4 \epsilon_{t-4} \tag{9}$$

$$Z_{t+1} = -\Phi_1 Z_t - \Phi_2 Z_{t-1} - \Phi_4 Z_{t-3} - \Phi_5 Z_{t-4} - \Phi_6 Z_{t-5} - \epsilon_{t+1} + \Theta_4 \epsilon_{t-3} \tag{10}$$

$$\hat{Z}_{t+1|t} = E[Z_{t+1}|Z_t, Z_{t-1}.....] \tag{11}$$

$$\hat{Z}_{t+1|t} = -\Phi_1 Z_t - \Phi_2 Z_{t-1} - \Phi_4 Z_{t-3} - \Phi_5 Z_{t-4} - \Phi_6 Z_{t-5} - \Theta_4 \epsilon_{t-3} \tag{12}$$

We need ϵ_{t-3},
by making use of (10) and (12) we then take:

$$Z_{t+1} - \hat{Z}_{t+1|t} = \epsilon_{t+1} \tag{13}$$

Then by plugin the (4):

$$\epsilon_{t-3} = Z_{t-3} - \hat{Z}_{t-3|t-4} \tag{14}$$

In order to compute the $\hat{Z}_{t-3|t-4}$, we need ϵ_{t-7}
Then we need $\epsilon_{t-11}, \epsilon_{t-15}, \epsilon_{t-19}\cdots$
We choose to put $\epsilon_{t-15} = 0$
Then we can get $\epsilon_{t-11}, \epsilon_{t-7}$ and ϵ_{t-3}
Finally, the 95% prediction interval is given by $\pm M_{a/2}\sqrt{V[e_{t+1|t}]}$ with :

$$V[e_{t+1|t}] = V[Y_{t+1} - \hat{Y}_{t+1|t}] = V[Z_{t+1} - \hat{Z}_{t+1|t}] = V[\epsilon_{t+1}] = \sigma_\epsilon^2 \tag{15}$$

Let's do the same for Y_{t+2}

$$Z_{t+2} = -\Phi_1 Z_{t+1} - \Phi_2 Z_t - \Phi_4 Z_{t-2} - \Phi_5 Z_{t-3} - \Phi_6 Z_{t-4} + \epsilon_{t+2} + \theta_4 \epsilon_{t-2} \tag{16}$$
$$\hat{Z}_{t+2|t} = E[Z_{t+2}|Z_t, Z_{t-1}....] \tag{17}$$
$$\hat{Z}_{t+2|t} = -\Phi_1 \hat{Z}_{t+1|t} - \Phi_2 Z_t - \Phi_4 Z_{t-2} - \Phi_5 Z_{t-3} - \Phi_6 Z_{t-4} + \theta_4 \epsilon_{t-2} \tag{18}$$

Now, we need ϵ_{t-2}, so, by making use of (13):

$$\epsilon_{t-2} = Z_{t-2} - \hat{Z}_{t-2|t-3} \tag{19}$$

Now, we need $\epsilon_{t-6}, \epsilon_{t-10}, \epsilon_{t-14}, \epsilon_{t-18}$
We put $\epsilon_{t-14} = 0$
And then we find $\hat{Y}_{t+2|t}$
The 95% prediction interval is given by $\pm M_{a/2}\sqrt{V[e_{t+2|t}]}$

$$e_{t+2|t} = Z_{t+2} - \hat{Z}_{t+2|t} \tag{20}$$

By making use of (16) and (18), we take:

$$e_{t+2|t} = -\Phi_1 (Z_{t+1} - \hat{Z}_{t+1|t}) + \epsilon_{t+2} \tag{21}$$
$$e_{t+2|t} = -\Phi_1 e_{t+1|t} + \epsilon_{t+2} \tag{22}$$
$$V[e_{t+2|t}] = \Phi_1^2 V[e_{t+1|t}] + V[\epsilon_{t+2}] \tag{23}$$
$$V[e_{t+2|t}] = \Phi_1^2 \sigma_\epsilon^2 + \sigma_\epsilon^2 \tag{24}$$
$$V[e_{t+2|t}] = \sigma_\epsilon^2 (1 + \Phi_1^2) \tag{25}$$

Afterwards, the above equations are coded in ”RStudio” (Code in Appendix) and the predictions are computed. Finally, on Table 2 are illustrated the predicted values for the first two quarters of 2019, along with their 95% prediction intervals (p.i.):

Table 2: Predicted values for the first two quarters of 2019

t	**2019Q1**	**2019Q2**
Lower 95% p.i.	1835.154	1743.582
Yt	2211.979	2287.255
Upper 95% p.i.	2588.804	2830.929

2) In Figure 4 are illustrated the predicted and actual values of apartment sales, Y_t, in the capital region of Denmark over time. The model predicts an increase in apartment sales for 2019Q1 and 2019Q2 in relation to the last quarter of 2018.

From the model structure it can be observed that that we have a seasonal ARIMA model structure $(2, 0, 0)\times(1,0,1)_4$, which indicates that the process has a seasonal autoregressive order of 2, no differencing, a non-seasonal moving average order of 0, a seasonal difference of 4, a seasonal autoregressive order of 1, and a seasonal moving average order of 1. Consequently, the data have a strong seasonal component that needs to be taken into account when making predictions.

In terms of the plot, as we move over time we can see that the uncertainty of our predictions is getting higher. That can be observed when taking a deeper look at the arc's chord, which is created from the prediction intervals. Overall, it is expected that the further in future we aim to predict the number of apartment sales, the higher the unexplained part of the model error will become.

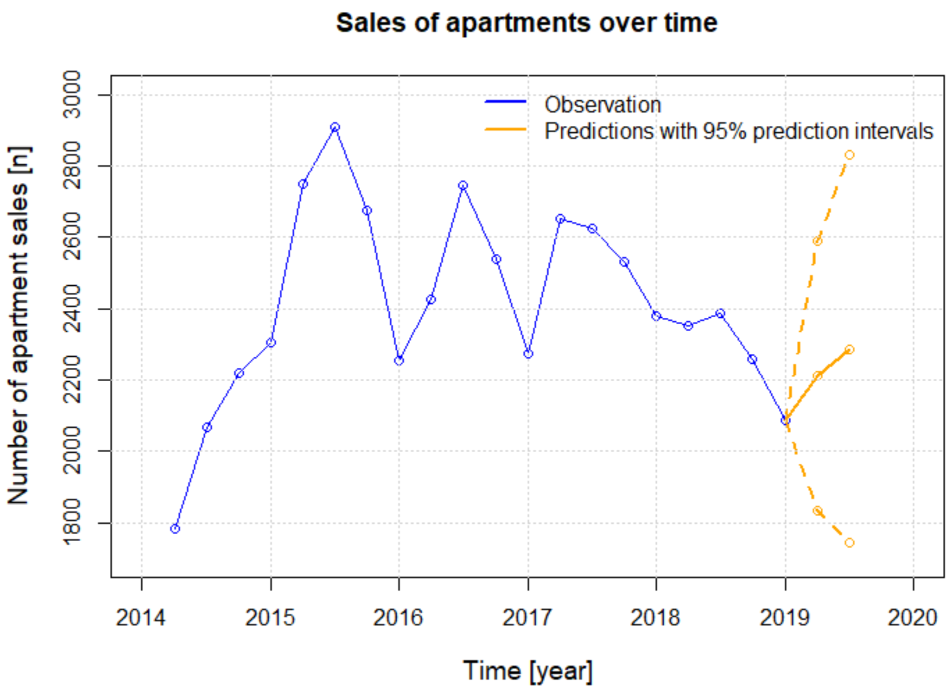

Figure 4: Sales of apartments over time in the capital region of Denmark

Question 2.3: ARMA(2,0) process

In the last section of the assignment we consider an ARMA(2,0) process defined by the following equation:

$$\phi(B)X_t = X_t - 1.5X_{t-1} + \phi_2 X_{t-2} = \epsilon_t \tag{26}$$

Four variations of the process are examined with $\phi_2 \in 0.52, 0.98$ and $\sigma^2 \in 0.1^2, 5^2$, by simulating 300 observations of each of the four processes 100 times (Code in Appendix). The overall aim is to estimate the model parameters based on the simulated sequences.

1) Firstly, we calculate the roots of $\phi(z^{-1}) = 0$ for both values of ϕ_2 as illustrated in the following code of "RStudio":

Listing 2: Question 2.3.1

```
root1 = polyroot(rev(c(1, -1.5, phi2[1])))
(abs(root1))
root2 = polyroot(rev(c(1, -1.5, phi2[2])))
(abs(root2))
```

For $\phi_2 = 0.52$ and $\phi_2 = 0.98$ we respectively find that:

$$roots(\phi(z^{-1})) = \begin{bmatrix}0.5438447 & 0.9561553\end{bmatrix} \text{ and } \begin{bmatrix}0.75 + 0.6461424i & 0.75 - 0.6461424i\end{bmatrix} \tag{27}$$

However, the absolute value of both roots is below 1, indicating that for both ϕ_2 values the roots lie in the unit circle proving that the process is stationary.

2)

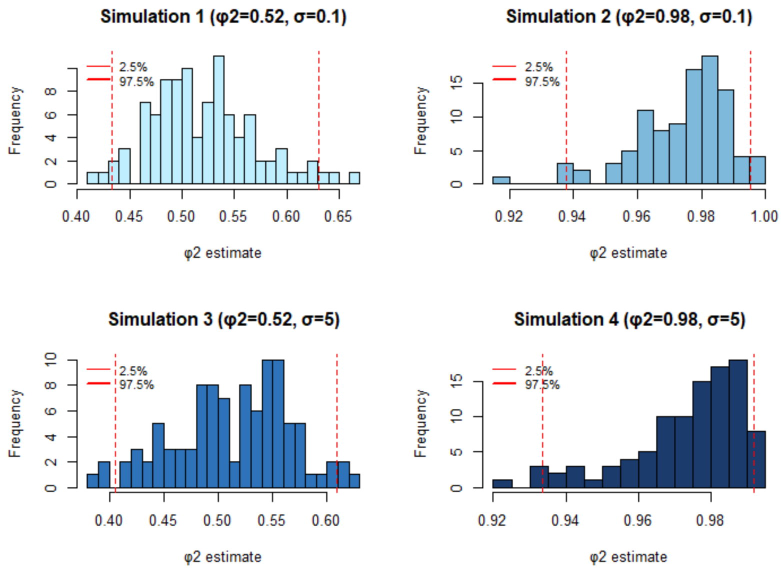

Figure 5: Histogram plot of the estimates of parameter ϕ_2 for each process

3) In the simulation study illustrated in Figure 5, the effects of different values of ϕ_2 on the variance/distribution of estimated ϕ_2 values are investigated for a fixed value of σ. The simulations show that for a specific value of σ, simulations with ϕ_2 = 0.52 produce a wider distribution of estimated ϕ_2 values compared to simulations with ϕ_2 = 0.98.

A wider distribution of estimated ϕ_2 values indicates a greater degree of uncertainty in the estimation of the true ϕ_2 value. This is because the estimated values are more spread out around the true value. In contrast, a narrower distribution of estimated ϕ_2 values indicates a greater degree of precision in the estimation of the true ϕ_2 value. This is because the estimated values are more tightly clustered around the true value.

Therefore, the wider distribution of estimated ϕ_2 values for simulations with ϕ_2 = 0.52 indicates that the true ϕ_2 value is likely to be further away from this value compared to simulations with ϕ_2 = 0.98. In other words, the simulations suggest that the true ϕ_2 value is closer to 0.98 than it is to 0.52. This is an important result because it suggests that if one were to estimate the ARMA(2,0) model using actual data, one should expect a wider distribution of estimated ϕ_2 values if the true value of ϕ_2 is closer to 0.52, and a narrower distribution if the true value is closer to 0.98.

4) In Figure 5, we observe that the variance/distribution of estimated ϕ_2 values is similar between simulations with different noise level (σ) values, for a specific true ϕ_2 value. Specifically, for a true value of $\phi_2 = 0.52$, we observe that the estimated values show similar distributions for both $\sigma = 0.1$ and $\sigma = 5$, with a slightly higher estimated mean ϕ_2 value for the simulation with $\sigma = 5$. Similarly, for a true value of $\phi_2 = 0.98$, we observe that the distributions of estimated ϕ_2 values are similar for both $\sigma = 0.1$ and $\sigma = 5$, with a slightly higher estimated mean ϕ_2 value for the simulation with $\sigma = 5$.

These results suggest that the effect of noise level on the estimated ϕ_2 values is relatively weak but observable. The reason for this is that the effect of noise on the ARMA(2,0) model is dependent on the magnitude of the true ϕ_2 value.

For example, for a small true ϕ_2 value, the impact of noise on the estimated ϕ_2 value may be relatively large because the noise can potentially dominate the signal. However, as the true ϕ_2 value increases, the impact of noise on the estimated ϕ_2 value decreases because the signal becomes stronger relative to the noise.

Overall, these results suggest that when estimating ϕ_2 in an AR model, it is important to consider the magnitude of the true ϕ_2 value and its interaction with the noise level in the data. Additionally, it may be useful to perform simulations with different noise levels to investigate the sensitivity of the estimation results to the noise level.

5) The estimated pairs of parameters (ϕ_1, ϕ_2) for the four variations are illustrated in Figure 6. It is evident that the simulations made with ϕ_2 = 0.98 produce estimations for ϕ_1 and ϕ_2 with smaller variances than simulations made with ϕ_2 = 0.52.

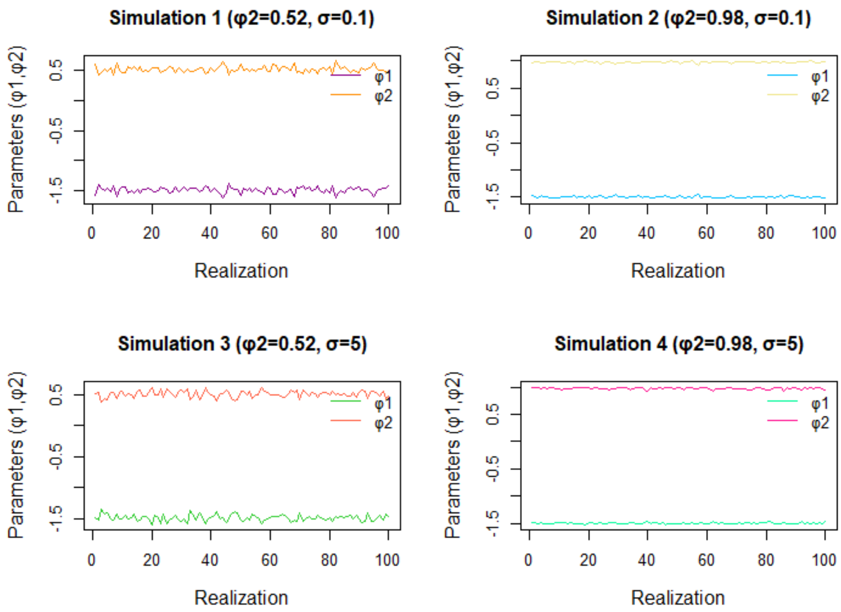

Figure 6: Estimated pairs of parameters (ϕ_1, ϕ_2)

6) A stationary process is one whose statistical properties, such as the mean and variance, do not change over time. In our case, we have a stationary process which allows us to make more accurate predictions. Consequently, the distribution of the estimated values will typically be more stable and less prone to extreme values than in the case of a non-stationary process. This is because the statistical properties of the process are constant over time, so the model can make predictions that are more consistent with the underlying data.

In addition, the distribution of the estimated values will be centered around the true parameter values of the underlying process. This means that the estimated values will have less bias and be more accurate than in the case of a non-stationary process, where the estimated values may be biased due to changes in the underlying statistical properties over time. Overall, having a stationary process can make it easier to fit accurate models and make reliable predictions, which is why it is often desirable in time series analysis.

Code

Listing 3: Question 2.1.3

```
# Q2.1.3: Simulate 10 realizations of the process with 200
    observations each and plot them
n <- 200 # Number of observations
r <- 10 # Number of realization
Y.all <- matrix(NA,ncol=n, nrow=r)
for (j in 1:r){
    Y <- arima.sim(list(ar=0.8,ma=c(0.8,-0.5)),n, sd = 0.4)
    Y.all[j,] <- Y
}
matplot(t(Y.all),lty=1, type = "l", col=rainbow(11),
        xlab = "Time [t]", ylab = "Observation [Xt]",
        main="Stochastic process (10 realizations)")
```

Listing 4: Question 2.1.4

```
# Q2.1.4: Estimate the autocorrelation function (ACF) for each
    realization and plot them
par(mfrow=c(1,1))
ACF.all <- matrix(NA,ncol=24, nrow=r)
colors <- rainbow(r)
for (j in 1:r){
  ACF <- acf(Y.all[j,], xlab="Lag", ylab="ACF", plot=FALSE)
  ACF.all[j,] <- ACF$acf
}
ACF.lags <- ACF$lag
matplot(ACF.lags, t(ACF.all), type="l", col=colors, xlab="Lag",
    ylab="ACF", main="Autocorrelation function for each realization
    ")
for (j in 1:r){
  text(ACF.lags, ACF.all[j,]+0.05, j, pos=1, col=colors[j])
}
abline(h=c(2/sqrt(n),-2/sqrt(n)), lty=2)
```

Listing 5: Question 2.1.5

```
# Q2.1.5: Estimate the partial autocorrelation function (PACF) for
    each realization and plot them
par(mfrow=c(1,1))
PACF.all <- matrix(NA, ncol=23, nrow=r)
colors <- rainbow(r)
for (j in 1:r){
```

```
  PACF <- pacf(Y.all[j,], xlab="Lag", ylab="PACF", plot=FALSE)
  PACF.all[j, 1:min(23, length(PACF$acf))] <- PACF$acf[1:min(23, length(PACF$acf))]
}
PACF.lags <- PACF$lag
matplot(PACF.lags, t(PACF.all), type="b", pch=19, col=colors, xlab="Lag", ylab="PACF", main="Partial autocorrelation function for each realization")
abline(h=c(2/sqrt(n),-2/sqrt(n)), lty=2)
legend("topright", legend=paste("Realization", 1:r), col=colors, pch=19, lty=1, ncol=2)
```

Listing 6: Question 2.1.6

```
# Q2.1.6: Calculate the variance of each of the realizations
var.all <- rep(NA, r)
for (j in 1:r){
  var.all[j] <- var(Y.all[j,])
}
(var.all)
(mean(var.all))
```

Listing 7: Question 2.2.1

```
# Q2.2.1: Predict the values of Yt for t=2019Q1 and 2019Q2, along with 95% prediction intervals
# Load the data
data <- read.table("A2_sales.txt", header = TRUE)
t <- seq(2014.25,2019,by=0.25)
y <- data$Sales

# Model constants
phi <- c(-1.04, 0.2, -0.86, 0.8944, -0.172)
theta <- -0.42
sigma2 <- 36963
mu <- 2070

# Center data
z <- y - mu

estimate <- function(t, z, eps_t_minus_4) {
  -phi[1]*z[t-1] - phi[2]*z[t-2] - phi[3]*z[t-4] - phi[4]*z[t-5] - phi[5]*z[t-6] + theta*eps_t_minus_4
}

```

```
# Estimate Yt for t = 2019Q1 using epsilon5 = 0
epsilon5 <- 0
z9_hat <- estimate(9, z, epsilon5)
epsilon9 <- z[9] - z9_hat
z13_hat <- estimate(13, z, epsilon9)
epsilon13 <- z[13] - z13_hat
z17_hat <- estimate(17, z, epsilon13)
epsilon17 <- z[17] - z17_hat
z21_hat <- estimate(21, z, epsilon17)
y21_hat <- z21_hat + mu
(y21_hat)

# Estimate Yt for t = 2019Q2 using epsilon6 = 0
epsilon6 <- 0
z10_hat <- estimate(10, z, epsilon6)
epsilon10 <- z[10] - z10_hat
z14_hat <- estimate(14, z, epsilon10)
epsilon14 <- z[14] - z14_hat
z18_hat <- estimate(18, z, epsilon14)
epsilon18 <- z[18] - z18_hat
z22_hat <- -phi[1]*z21_hat - phi[2]*z[22-2] - phi[3]*z[22-4] - phi[4]*z[22-5] - phi[5]*z[22-6] + theta*epsilon18
y22_hat <- z22_hat + mu
(y22_hat)

# 95% prediction intervals
uncertainity_y21 <- 1.96*sqrt(sigma2)
uncertainity_y22 <- 1.96*sqrt(sigma2)*sqrt(1+phi[1]^2)

# Store data
tpred <- c(2019, 2019.25, 2019.50)
ypred <-  matrix(y[20],ncol=3, nrow=3)
ypred[2,] <- c(y[20], y21_hat, y22_hat)
ypred[1,2] <- ypred[2,2]-uncertainity_y21
ypred[3,2] <- ypred[2,2]+uncertainity_y21
ypred[1,3] <- ypred[2,3]-uncertainity_y22
ypred[3,3] <- ypred[2,3]+uncertainity_y22
```

Listing 8: Question 2.2.2

```
# Q2.2.2: Plot the actual and the predicted values
plot(t, y, type = "l", col="blue",
     xlab="Time [year]", ylab="Number of apartment sales [n]",
     xlim=c(2014,2020), ylim=c(1700,3000), cex.lab=1.2)
```

```
points(t, y, col="blue")
matlines(tpred, t(ypred), type="l",lty=c(2,1,2),lwd=2, col="#
    FFA500")
matpoints(tpred[2:3], t(ypred[1:3,2:3]), pch = 1, col="#FFA500")
legend("topright", c("Observation", "Predictions with 95%
    prediction intervals")
        , col=c("blue", "#FFA500"), lty=1, bty='n', lwd=2)
title("Sales of apartments over time")
grid()
```

Listing 9: Question 2.3

```
# Simulate 300 observations of each of the four processes 100
    times
phi2 <- c(0.52, 0.98)
sigma <- c(0.1, 5)
n <- 300 # Number of observations
r <- 100 # Number of realizations
sim1 <- replicate(r,arima.sim(model = list(ar=c(1.5, -phi2[1]),
    order=c(2,0,0)), n, sd=sigma[1]))
sim2 <- replicate(r,arima.sim(model = list(ar=c(1.5, -phi2[2]),
    order=c(2,0,0)), n, sd=sigma[1]))
sim3 <- replicate(r,arima.sim(model = list(ar=c(1.5, -phi2[1]),
    order=c(2,0,0)), n, sd=sigma[2]))
sim4 <- replicate(r,arima.sim(model = list(ar=c(1.5, -phi2[2]),
    order=c(2,0,0)), n, sd=sigma[2]))
```

Listing 10: Question 2.3.2

```
# Q2.3.2: For each process, make a histogram plot of the estimates
    of parameter phi2 and indicate the 95% quantiles
nbin <- 20
par(mfrow=c(2,2))

# sim1
param.all1 <- matrix(NA,ncol=2, nrow=r)
for (i in 1:r){
    X <- matrix(NA,ncol=2, nrow=(n-2))
    X[,1] <- -sim1[2:(n-1),i]
    X[,2] <- -sim1[1:(n-2),i]
    Y <- sim1[3:n,i]
    param <- solve(t(X) %*% X) %*% t(X) %*% Y
    param.all1[i,] <- param
}
```

```r
hist(param.all1[,2], breaks=nbin, main="Simulation 1 (phi2=0.52, sigma=0.1)", xlab="phi2 estimate", col='#BFEFFF')
abline(v=quantile(param.all1[,2], 0.025), lty=2, col='red')
abline(v=quantile(param.all1[,2], 0.975), lty=2, col='red')
quantiles <- quantile(param.all1[,2], c(0.025, 0.5, 0.975))
legend("topleft", legend=c("2.5%", "97.5%"),
       col=c("red", "red"), lwd=c(1, 2, 1),
       lty=c(1, 1, 1), bty="n", cex=0.8)

# sim2
param.all2 <- matrix(NA, ncol=2, nrow=r)
for (i in 1:r){
    X <- matrix(NA, ncol=2, nrow=(n-2))
    X[,1] <- -sim2[2:(n-1),i]
    X[,2] <- -sim2[1:(n-2),i]
    Y <- sim2[3:n,i]
    param <- solve(t(X) %*% X) %*% t(X) %*% Y
    param.all2[i,] <- param
}
hist(param.all2[,2], breaks=nbin, main="Simulation 2 (phi2=0.98, sigma=0.1)", xlab="phi2 estimate", col='#7EB8DA')
abline(v=quantile(param.all2[,2], 0.025), lty=2, col='red')
abline(v=quantile(param.all2[,2], 0.975), lty=2, col='red')
quantiles <- quantile(param.all2[,2], c(0.025, 0.5, 0.975))
legend("topleft", legend=c("2.5%", "97.5%"),
       col=c("red", "red"), lwd=c(1, 2, 1),
       lty=c(1, 1, 1), bty="n", cex=0.8)

# sim3
param.all3 <- matrix(NA, ncol=2, nrow=r)
for (i in 1:r){
    X <- matrix(NA, ncol=2, nrow=(n-2))
    X[,1] <- -sim3[2:(n-1),i]
    X[,2] <- -sim3[1:(n-2),i]
    Y <- sim3[3:n,i]
    param <- solve(t(X) %*% X) %*% t(X) %*% Y
    param.all3[i,] <- param
}
hist(param.all3[,2], breaks=nbin, main="Simulation 3 (phi2=0.52, sigma=5)", xlab="phi2 estimate", col='#2E73B9')
abline(v=quantile(param.all3[,2], 0.025), lty=2, col='red')
abline(v=quantile(param.all3[,2], 0.975), lty=2, col='red')
quantiles <- quantile(param.all3[,2], c(0.025, 0.5, 0.975))
legend("topleft", legend=c("2.5%", "97.5%"),
```

```
        col=c("red", "red"), lwd=c(1, 2, 1),
        lty=c(1, 1, 1), bty="n", cex=0.8)

# sim4
param.all4 <- matrix(NA, ncol=2, nrow=r)
for (i in 1:r){
    X <- matrix(NA, ncol=2, nrow=(n-2))
    X[,1] <- -sim4[2:(n-1),i]
    X[,2] <- -sim4[1:(n-2),i]
    Y <- sim4[3:n,i]
    param <- solve(t(X) %*% X) %*% t(X) %*% Y
    param.all4[i,] <- param
}
hist(param.all4[,2], breaks=nbin, main="Simulation 4 (phi2=0.98,
   sigma=5)", xlab="phi2 estimate", col='#1A3B6C')
abline(v=quantile(param.all4[,2], 0.025), lty=2, col='red')
abline(v=quantile(param.all4[,2], 0.975), lty=2, col='red')
quantiles <- quantile(param.all4[,2], c(0.025, 0.5, 0.975))
legend("topleft", legend=c("2.5%", "97.5%"),
        col=c("red", "red"), lwd=c(1, 2, 1),
        lty=c(1, 1, 1), bty="n", cex=0.8)
```

Listing 11: Question 2.3.5

```
# Q2.3.5: Plot all the estimated pairs of parameters (phi1, phi2)
   for the four variations
par(mfrow=c(2,2))
# set up colors
colors1 <- c("#8B008B", "#FF8C00")  # Dark Magenta and Dark Orange
colors2 <- c("#00BFFF", "#F0E68C")  # Deep Sky Blue and Khaki
colors3 <- c("#32CD32", "#FF6347")  # Lime Green and Tomato
colors4 <- c("#00FA9A", "#FF1493")  # Medium Spring Green and Deep
     PinÎş

for (i in 1:4) {
  if (i == 1) {
    matplot(param.all1, lty=1, type="l", main="Simulation 1 (Ď2
       =0.52, sigma=0.1)",
            xlab="Realization", ylab='Parameters (phi1, phi2)',
               col=colors1, cex.lab=1.2)
    legend("topright", legend=c("phi1", "phi2"), lty=1, col=
       colors1, bty='n')
  } else if (i == 2) {
    matplot(param.all2, lty=1, type="l", main="Simulation 2 (phi2
```

```
=0.98, sigma=0.1)",
            xlab="Realization", ylab='Parameters (phi1,phi2)', col
              =colors2, cex.lab=1.2)
    legend("topright", legend=c("phi1", "phi2"), lty=1, col=
      colors2, bty='n')
  } else if (i == 3) {
    matplot(param.all3, lty=1, type="l", main="Simulation 3 (phi2
      =0.52, sigma=5)",
            xlab="Realization", ylab='Parameters (phi1,phi2)', col
              =colors3, cex.lab=1.2)
    legend("topright", legend=c("phi1", "phi2"), lty=1, col=
      colors3, bty='n')
  } else {
    matplot(param.all4, lty=1, type="l", main="Simulation 4 (phi2
      =0.98, sigma=5)",
            xlab="Realization", ylab='Parameters (phi1,phi2)', col
              =colors4, cex.lab=1.2)
    legend("topright", legend=c("phi1", "phi2"), lty=1, col=
      colors4, bty='n')
  }
}
```

Extra Plots

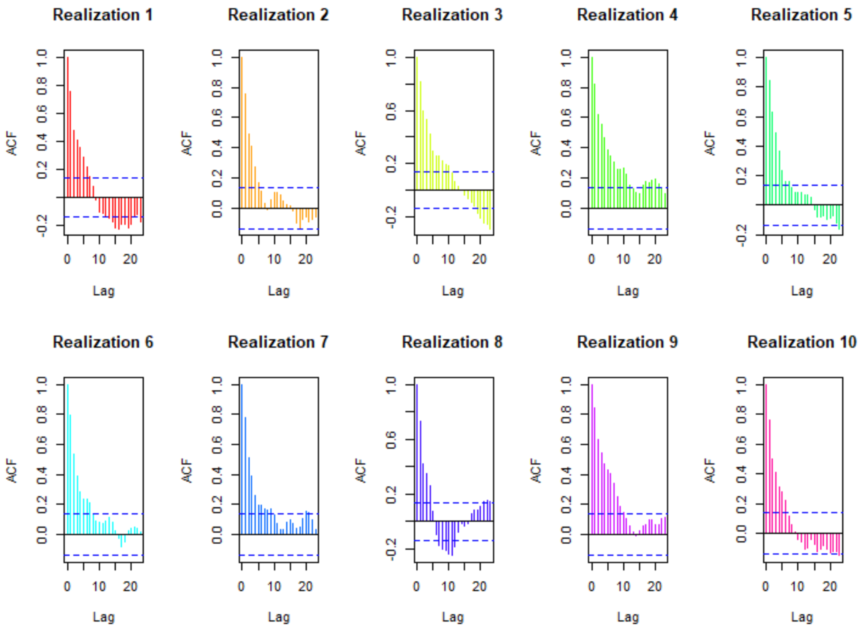

Figure 7: Autocorrelation function (ACF) for each realization (separate)

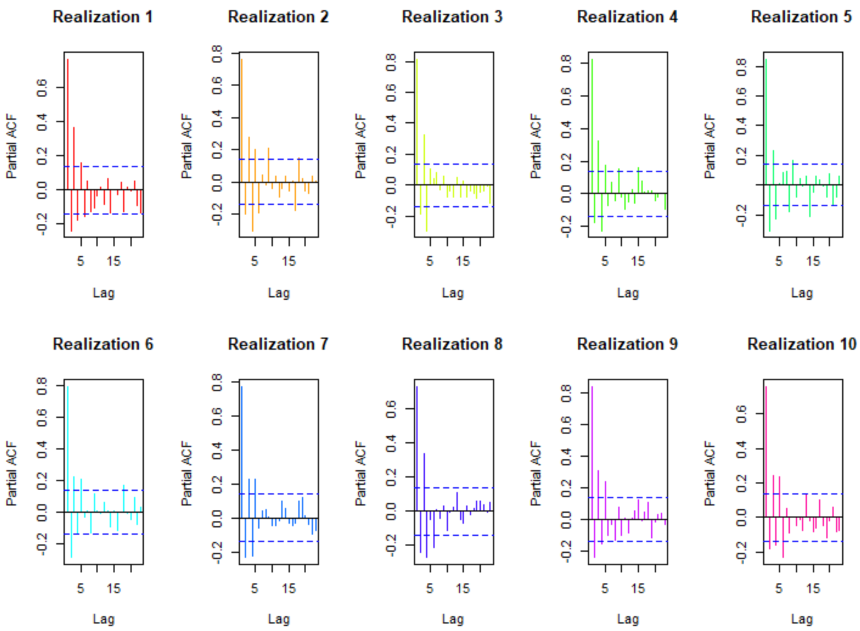

Figure 8: Partial autocorrelation function (PACF) for each realization (separate)