# MA30085, Time Series - Coursework

Candidate Number 24075

28/03/2024 noon till 17/04/2024 noon

## Description of coursework

For this assignment, you must perform a data analysis task. Data consist of 2 time series randomly selected from the *lts0.Rda* data file (containing 1000 time series). The goal for you is to study carefully both time series, and attempt to model them using the main steps of the Box-Jenkins methodology.

The coursework will be marked based on the statistical reasoning that has led to your conclusions. Thus to carry through the appropriate statistical tasks and reasoning is more important than the conclusions themselves. The highest score achievable is 100 (60 for series 1 and 40 for series 2).

A thorough illustration of this way of proceeding is included in the document prepared for the computational laboratory 3. You can use that document as guidance, but should also feel free to attempt other types of investigation, if necessary.

If you have followed all lectures, studied the related material and followed all computationally laboratories, the time required to analyse both time series turns out to be approximately six to eight hours.

## Preparation - How to obtain your data

### Creation of random seed

The two time series to be analysed are different for each student. Your data are obtained using an integer seed, generated using your unique student number. More specifically, the integer seed to generate the random numbers used for the work consists of the last five digits of your unique, 9-digits, student number. Representing a student number with letters,

student number = $abcdefghi$,

the seed number is

seed number = $efghi$

For example, if your unique student number is 179238011, your seed number is 38011. Or, if your unique student number is 179200810, your seed number is 810, because 00810 is interpreted by R as 810.

### Extraction of time series

The time series for your coursework are extracted randomly from a binary file named *lts0.Rda*. This file **must be** located in the same directory in which you have transferred this R markdown document, *MA30085_coursework.Rmd*.

By running the code in the R chunk below, you will automatically import the two time series, called `tser1` and `tser2`. They are objects of class `ts`. If the operation is successful, you should see both time series displayed in a graphic. Once this is done, you are ready for the analysis.

## Type of time series simulated

The **first time series** (`tser1`) is a simulation of an ARIMA process, $\{X_t,\ t \in \mathbb{Z}\}$, described by the difference equation

$$\phi(B)(1-B)^d X_t = \theta(B) Z_t,$$

where $\phi(\lambda)$ is the AR characteristic polynomial of order $p$, $\theta(\lambda)$ is the MA characteristic polynomial of order $q$, $\{Z_t,\ t \in \mathbb{Z}\}$ a Gaussian white noise process with mean 0 and standard deviation 1, and where $B$ is the backward shift operator. The parameters $p$, $q$ and $d$ of the ARIMA$(p,d,q)$ process are integers with the following range:

$$d = 0, 1, 2 \qquad p = 0, 1, 2, 3 \qquad q = 0, 1, 2, 3.$$

The **second time series** (`tser2`), $Y_t$, has the form

$$Y_t = X_t + m_t,$$

where $X_t$ is an ARIMA$(p,d,q)$ process different from that of the first time series, with

$$d = 0, 1 \qquad p = 0, 1, 2 \qquad q = 0, 1, 2,$$

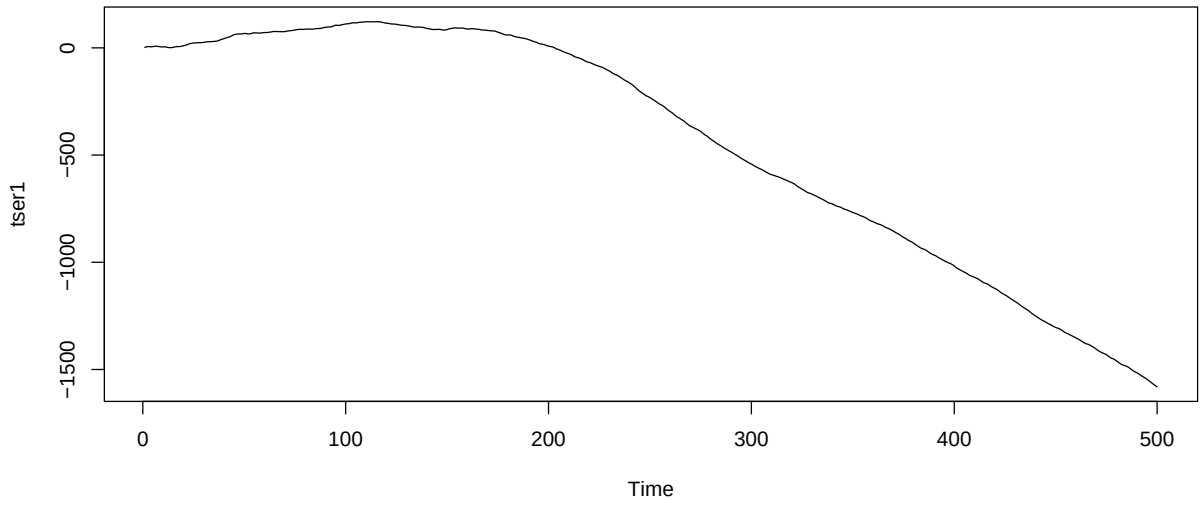
and where $m_t$ is a deterministic polynomial of degree 1 or 2.

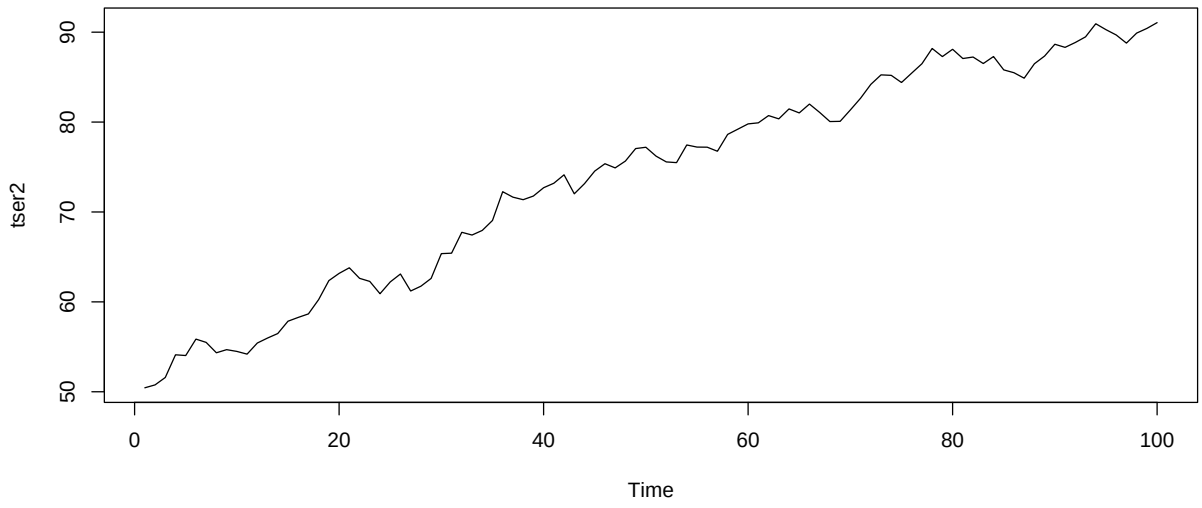
## R chunk for importing data

Below is the R chunk needed to get your work started, as instructed earlier. Once executed, please add all necessary text, code and graphics (please, make sure the size of the graphics window is big enough for me to check details) under the line **SOLUTION**. The final work should be uploaded on Crowdmark as a PDF document.

```
#************************#
#!!! VERY IMPORTANT !!!
#************************#
# Please, replace the "2" inside set.seed() with your
# unique seed. Failure to do so might result in your work
# being penalised

#r interprets 09527 as 9527
set.seed(9527)

#************************#
#!!! VERY IMPORTANT !!!
#
#  DON'T MODIFY THE LINES
#  IN THE REMAINING CODE
#
#************************#
# Loading data
load("lts0.Rda")

# Extracting time series
idx1 <- sample(1:500,size=1)
idx2 <- sample(501:1000,size=1)
tser1 <- lts0[[idx1]]
tser2 <- lts0[[idx2]]
```

```r
# Test you've got the time series in the workspace
par(mfrow=c(2,1))
plot(tser1)
plot(tser2)
```




```r
# Back to one plot per window
par(mfrow=c(1,1))
```

**SOLUTION**

# Time series 1

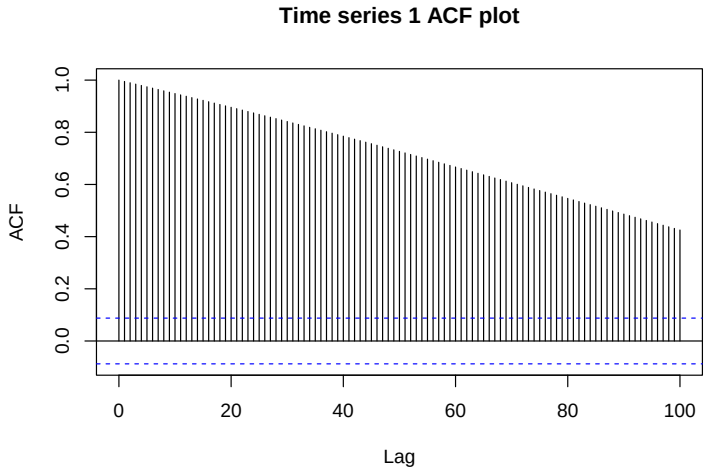
### Initial analysis

We initially notice that `tser1` visually looks non-stationary. Lets verify this is the case by computing the ac.f, Kolmogorov-Smirnov test and Augmented Dickey-Fuller test. For the Kolmogorov-Smirov test, we would fail to reject the null hypothesis at level $\alpha$ if both samples come from a population of the same distribution (stationary). For the Augmented Dickey-Fuller test, if we fail to reject the null hypothesis, then it is considered non-stationary. The function created below called `stationary_tests` computes these all in one go.

```
library(tseries)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
stationary_tests = function(ts_object, k_lags, ts_plot, str_title, n1, n2){
  # print the time series if ts_plot flag is true
  if (ts_plot == TRUE){
    plot(ts_object)
  }
  # plot ACF with the appropriate title given by user in flag 'str_title'
  autocorrelation = acf(ts_object,lag.max=100, plot = FALSE)
  plot(autocorrelation, main = str_title)
  # Kolmogorov-Smirnov test
  print(ks.test(ts_object[1:n1],ts_object[(n1+1):n2]))
  # Augmented Dickey-Fuller Test for different lags in list k_lags
  for (i in k_lags){
    print(adf.test(ts_object, k=i))
  }
}
stationary_tests(tser1, c(7,10,15), ts_plot =FALSE,
                 str_title = 'Time series 1 ACF plot', n1 = 250, n2 = 500)
```



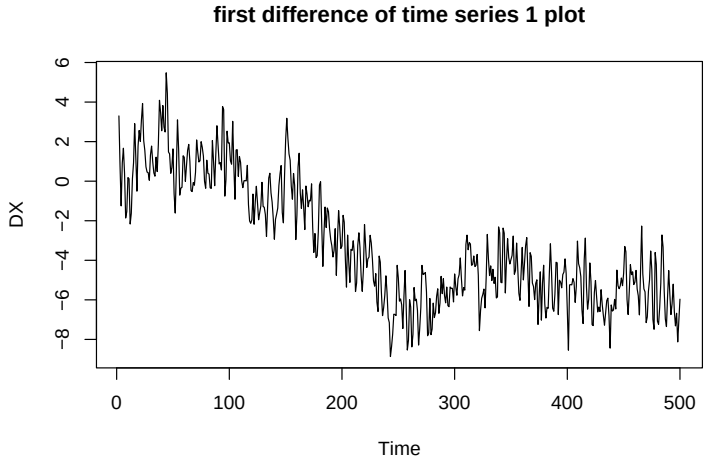
```
##
##  Asymptotic two-sample Kolmogorov-Smirnov test
##
## data:  ts_object[1:n1] and ts_object[(n1 + 1):n2]
## D = 1, p-value < 2.2e-16
## alternative hypothesis: two-sided
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -3.0539, Lag order = 7, p-value = 0.1322
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -3.231, Lag order = 10, p-value = 0.08258
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -3.0431, Lag order = 15, p-value = 0.1368
## alternative hypothesis: stationary
```

- **ac.f:** The series is not decaying fast enough which suggests non-stationary i.e. there are long term correlations
- **Kolmogorov-Smirnov test:** small p-value $<$ 2.2e-16 $<$ 0.05 suggests that the time series is heteroscedastic and non-stationary certinaly at least at the 5% level
- **Augmented Dickey-Fuller Test:** The p-value is above 0.05 for all $k = 7, 10, 15$. Therefore this suggests the series is non-stationary.

Therefore, we can conclude that time series 1 is non-stationary.

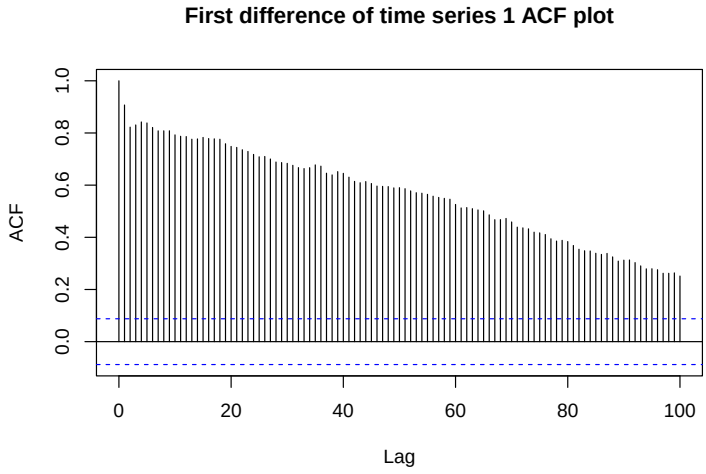
Now lets take the first difference of time series 1.

```
DX = diff(tser1)
plot(DX, main = 'first difference of time series 1 plot')
```




Visually, the series still looks non-stationary. Lets confirm this by computing the ac.f and some statistical tests.

```
stationary_tests(DX, c(7,10,15), ts_plot =FALSE,
                 str_title = 'First difference of time series 1 ACF plot',
                 n1 = 250, n2 = 249)
```




```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  ts_object[1:n1] and ts_object[(n1 + 1):n2]
## D = 0.904, p-value = 0.002193
## alternative hypothesis: two-sided
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -2.9122, Lag order = 7, p-value = 0.1922
```

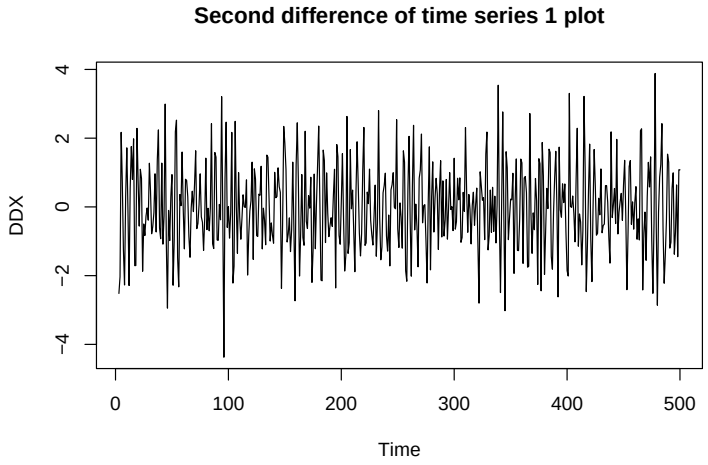
```
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -2.5736, Lag order = 10, p-value = 0.3355
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -2.0293, Lag order = 15, p-value = 0.5659
## alternative hypothesis: stationary
```

- **ac.f:** The series is not decaying fast enough which suggests non-stationary i.e. there are long term correlations
- **Kolmogorov-Smirnov test:** small p-value, certainly at least less than 0.05, also suggests the series is non-stationary, at at least the 5% level.
- **Augmented Dickey-Fuller Test:** The p-value is above 0.05 for all $k = 7, 10, 15$. Therefore this suggests the series is non-stationary.

We can therefore conclude that the first difference of time series 1 is non-stationary

Lets compute the second difference of time series 1.

```
DDX = diff(DX)
plot(DDX, main = 'Second difference of time series 1 plot')
```




The plot visually indicates that the series is stationary. Lets confirm this by computing the ac.f, and some statistical tests.

```
stationary_tests(DDX, c(7,10,15), ts_plot =FALSE,
str_title = 'Second difference of time series 1 ACF plot',
n1 = 249, n2 = 499)
```

```
##
##  Asymptotic two-sample Kolmogorov-Smirnov test
##
## data:  ts_object[1:n1] and ts_object[(n1 + 1):n2]
## D = 0.064257, p-value = 0.6828
## alternative hypothesis: two-sided
```

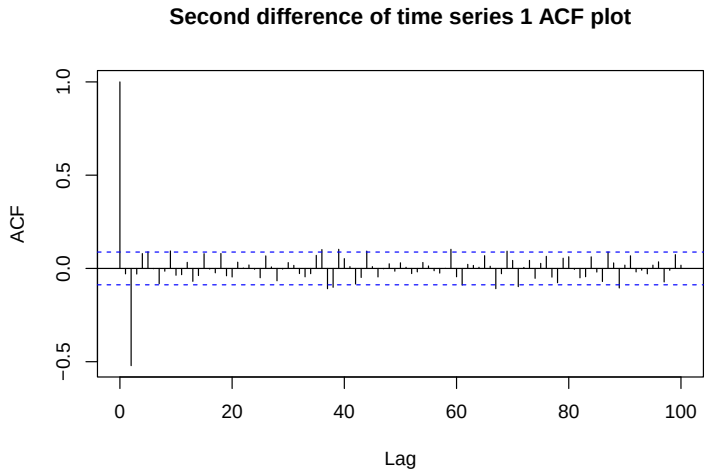
```
## Warning in adf.test(ts_object, k = i): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -11.843, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
## Warning in adf.test(ts_object, k = i): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -8.8434, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
```

```
## Warning in adf.test(ts_object, k = i): p-value smaller than printed p-value
```




```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -8.7078, Lag order = 15, p-value = 0.01
## alternative hypothesis: stationary
```

- **ac.f:** From the ac.f plot, we notice that there are no more of these long term correlations, and the value of the ACF is decreasing rapidly. This behavior certainly indicates a stationary process

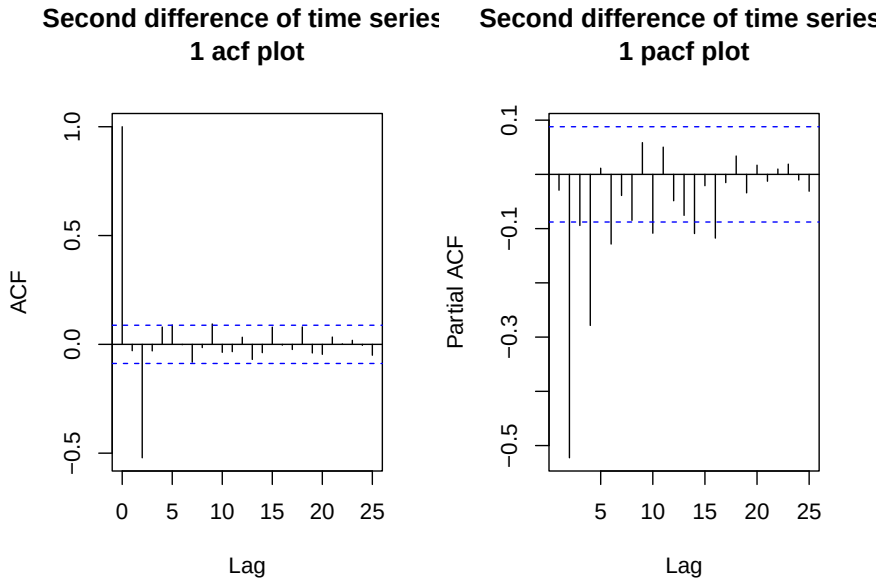
- **Kolmogorov-Smirnov test:** a large p-value suggests the series is stationary, certainly at least at the 5% level (0.05<0.6828).
- **Augmented Dickey-Fuller Test:** The p-value is below 0.05 for various values of the lag ($k = 7, 10, 15$). Therefore, this suggests the series is stationary.

We conclude that the second difference of time series 1 is stationary.

### Determination of the orders of ARMA

We begin by plotting the sample ac.f and the sample partial ac.f for the second difference of time series 1.

```
par(mfrow = c(1,2))
acf(DDX,lag.max=25, main = 'Second difference of time series \n1 acf plot')
pacf(DDX,lag.max=25, main = 'Second difference of time series \n1 pacf plot')
```




Visually, we observe that there is a sharp cutoff in the sample ac.f i.e. values of lag after 2 are all contained within the blue confidence bands ($\pm\frac{2}{\sqrt{500}}$). This means that they are statistically 0. Therefore, the behavior of the sample partial ac.f can be neglected, and we conclude that the second difference is a pure `MA(2)` process. Therefore, we conclude that a `ARIMA(0,2,2)` process can be used to model time series 1 (`tser1`).

### Estimation of model parameters

In the present case, the model `ARIMA(0,2,2)` seems to be the best model with very little doubts. However, in general it is wise to try estimates for models with parameters close to the chosen one. Estimates for models chosen generally have parameters close to the chosen model, or correspond to a slightly different interpretation. The models we will also consider are as follows,

- `ARIMA(1,2,1)`: Possible interpretation being that the sample ac.f is showing a decreasing pattern, rather than a truncation. Then, acknowledging that there is no sharp cutoff in the sample partial ac.f, a mixed `ARMA(p,q)` should be tried. Following the principles of Occam's Razor, we choose to model the second difference as `ARMA(1,1)` first, which corresponds to modelling time series 1 as a `ARIMA(1,2,1)` process.

- `ARIMA(2,2,1)`: Model which has parameters close to `ARIMA(1,2,1)`.
- `ARIMA(1,2,2)`: Model which has parameters close to `ARIMA(0,2,2)` and `ARIMA(1,2,1)`.
- `ARIMA(3,2,1)`: Motivation discussed below.

There does not seem to be a clear cut-off for the sample partial ac.f function. Therefore, we do not consider these AR cases.

We would like the models to satisfy the following criteria:

- 95% confidence interval for each model coefficient excludes 0 (all the model parameters are significant)
- AIC: We would like the AIC to be low to help ensure that the model we select is not overfitting the data
- $\sigma^2 \approx 1$: We would like the variance to be approximately 1 because we know that in the simulation, the variance was always 1.

```
modelA <- arima(tser1,order=c(0,2,2))
print(modelA)
```

```
##
## Call:
## arima(x = tser1, order = c(0, 2, 2))
##
## Coefficients:
##           ma1      ma2
##       -0.0824  -0.6988
## s.e.   0.0324   0.0322
##
## sigma^2 estimated as 1.049:  log likelihood = -719.14,  aic = 1444.28
```

```
# 95% confidence interval for the parameters
modelA$coef -2*sqrt(diag(modelA$var.coef))
```

```
##        ma1        ma2
## -0.1471634 -0.7631471
```

```
modelA$coef +2*sqrt(diag(modelA$var.coef))
```

```
##         ma1         ma2
## -0.01764215 -0.63454569
```

```
modelB <- arima(tser1,order=c(1,2,1))
print(modelB)
```

```
##
## Call:
## arima(x = tser1, order = c(1, 2, 1))
##
## Coefficients:
```

```
##          ar1      ma1
##       0.4663  -0.9010
## s.e.  0.0502   0.0225
##
## sigma^2 estimated as 1.327:  log likelihood = -777.46,  aic = 1560.92
```

```
modelB$coef -2*sqrt(diag(modelB$var.coef))
```

```
##        ar1        ma1
##  0.3658882 -0.9459685
```

```
modelB$coef +2*sqrt(diag(modelB$var.coef))
```

```
##        ar1        ma1
##  0.5666488 -0.8559959
```

```
modelC <- arima(tser1,order=c(2,2,1))
print(modelC)
```

```
##
## Call:
## arima(x = tser1, order = c(2, 2, 1))
##
## Coefficients:
##          ar1      ar2      ma1
##       0.3744  -0.4714  -0.6138
## s.e.  0.0686   0.0462   0.0807
##
## sigma^2 estimated as 1.128:  log likelihood = -737.09,  aic = 1482.18
```

```
modelC$coef -2*sqrt(diag(modelC$var.coef))
```

```
##        ar1        ar2        ma1
##  0.2370819 -0.5638625 -0.7752099
```

```
modelC$coef +2*sqrt(diag(modelC$var.coef))
```

```
##        ar1        ar2        ma1
##  0.5116301 -0.3789797 -0.4523699
```

```
modelD <- arima(tser1,order=c(1,2,2))
print(modelD)
```

```
##
## Call:
## arima(x = tser1, order = c(1, 2, 2))
##
## Coefficients:
##          ar1      ma1      ma2
##       -0.0829  -0.0419  -0.7094
## s.e.   0.0619   0.0424   0.0309
##
## sigma^2 estimated as 1.045:  log likelihood = -718.26,  aic = 1444.51
```

```
modelD$coef -2*sqrt(diag(modelD$var.coef))
```

```
##        ar1        ma1        ma2
## -0.2066324 -0.1266580 -0.7712029
```

```
modelD$coef +2*sqrt(diag(modelD$var.coef))
```

```
##         ar1         ma1         ma2
##  0.04089547  0.04290663 -0.64763780
```

There are several note worthy observations:

- `ARIMA(0,2,2)`: Model has the lowest AIC (1444.28), 95% confidence intervals for coefficients do not include 0, and a variance of $1.049 \approx 1$.
- `ARIMA(1,2,2)`: Although the model has the second lowest AIC, one of the confidence intervals includes 0. Therefore, this model is not a good choice. Models that increase $p$ higher than 1 and/or increase $q$ higher than 2 will also have the same issue (for example, `ARIMA(2,2,2)`,`ARIMA(2,2,3)`,`ARIMA(1,2,3)` etc.).
- `ARIMA(1,2,1)` and `ARIMA(2,2,1)`: Both models have 95% confidence intervals that do not include 0. The AIC of `ARIMA(2,2,1)` is greater than that of `ARIMA(1,2,1)`, $1560.92 > 1482.18$. The variance of `ARIMA(2,2,1)` is closer to 1 than that of `ARIMA(1,2,1)`, $1 \approx 1.128 < 1.327$.

The final bullet point indicates that it is worth testing the model `ARIMA(3,2,1)` to see if it has a lower AIC value and a variance closer to 1, compared to `ARIMA(0,2,2)`.

```
modelE <- arima(tser1,order=c(3,2,1))
print(modelE)
```

```
##
## Call:
## arima(x = tser1, order = c(3, 2, 1))
##
## Coefficients:
##          ar1      ar2     ar3      ma1
##       0.6711  -0.4980  0.2675  -0.8732
## s.e.  0.0649   0.0479  0.0579   0.0425
##
## sigma^2 estimated as 1.091:  log likelihood = -728.79,  aic = 1467.58
```

```
modelE$coef -2*sqrt(diag(modelE$var.coef))
```

```
##        ar1        ar2        ar3        ma1
##  0.5411845 -0.5937823  0.1517404 -0.9581897
```

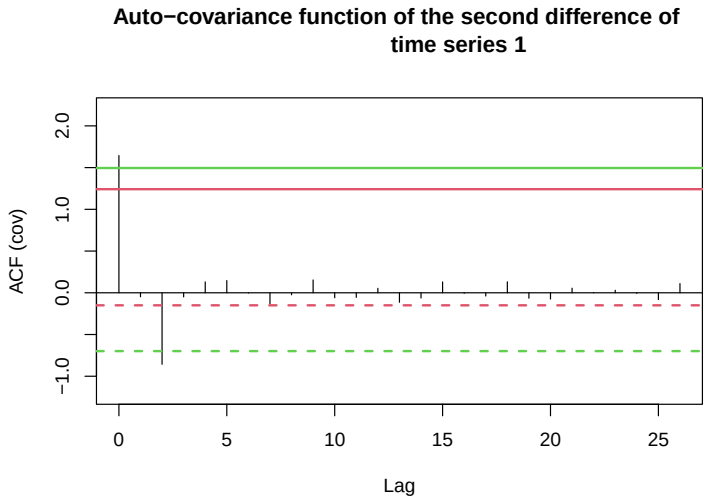
```
modelE$coef +2*sqrt(diag(modelE$var.coef))
```

```
##        ar1        ar2        ar3        ma1
##  0.8009244 -0.4021373  0.3833083 -0.7881918
```

For `ARIMA(3,2,1)`, all 95% confidence intervals still do not include 0, but the AIC is smaller to that of `ARIMA(2,2,1)` ($1467.58 < 1482.18$), and the variance is closer to one than `ARIMA(2,2,1)` ($1 \approx 1.091 < 1.128$). However, we still prefer `ARIMA(0,2,2)` over `ARIMA(3,2,1)`. This is because `ARIMA(0,2,2)` has a lower AIC ($1444.28 < 1467.58$) and a variance closer to 1 ($1 \approx 1.049 < 1.091$). In conclusion, we still believe that the `ARIMA(0,2,2)` model describes the data given here in the best way.

Since we predicted that the second difference of time series 1 is an `MA(2)` (green) process. We can compute the theoretical values for the auto-covariance using the estimates for the model coefficients. Since the theoretical auto-covariance function for `ARMA(1,1)` (red) is computed in the lecture notes, we choose this model for a comparison.

```
# for ARIMA(0,2,2)
modelA <- arima(tser1,order=c(0,2,2))
ma1_A = modelA$coef[1]
ma2_A = modelA$coef[2]
gamma_0_A = 1 + ma1_A^2 + ma2_A^2
gamma_2_A = ma2_A

# for ARIMA(1,2,1)
modelB <- arima(tser1,order=c(1,2,1))
ar1_B = modelB$coef[1]
ma1_B = modelB$coef[2]
gamma_0_B = (1 + ma1_B^2 + 2*ar1_B*ma1_B)/(1-ar1_B^2)
gamma_2_B = ((1+ar1_B*ma1_B)*(ar1_B + ma1_B))/(1-ar1_B^2)*ar1_B

# plotting on top of each other
autocovariance=acf(diff(diff(tser1)),type="covariance",ylim=c(-1.2,2.2),
                   main = 'Auto-covariance function of the second difference of
                   time series 1')
abline(h=gamma_0_A,lwd=2,col=3)
abline(h=gamma_2_A,lwd=2,col=3, lty = 2)
abline(h=gamma_0_B,lwd=2,col=2)
abline(h=gamma_2_B,lwd=2,col=2, lty = 2)
```




The dotted lines represent the predicted values of the autocovariance function at lag 2. Due to there only being 500 observations and that we are using estimated coefficients, the coloured lines do not quite line up with the peaks. It is clear that autocovariance values from the second difference of time series 1 are modeled

fairly well for a fitted `MA(2)` process (green). In comparison, `ARMA(1,1)` failed to model the autocovariance values from the second difference of time series 1 well. This discovery provides extra evidence for our belief that `ARIMA(0,2,2)` model describes the data given here in the best way.

### Agreement with data

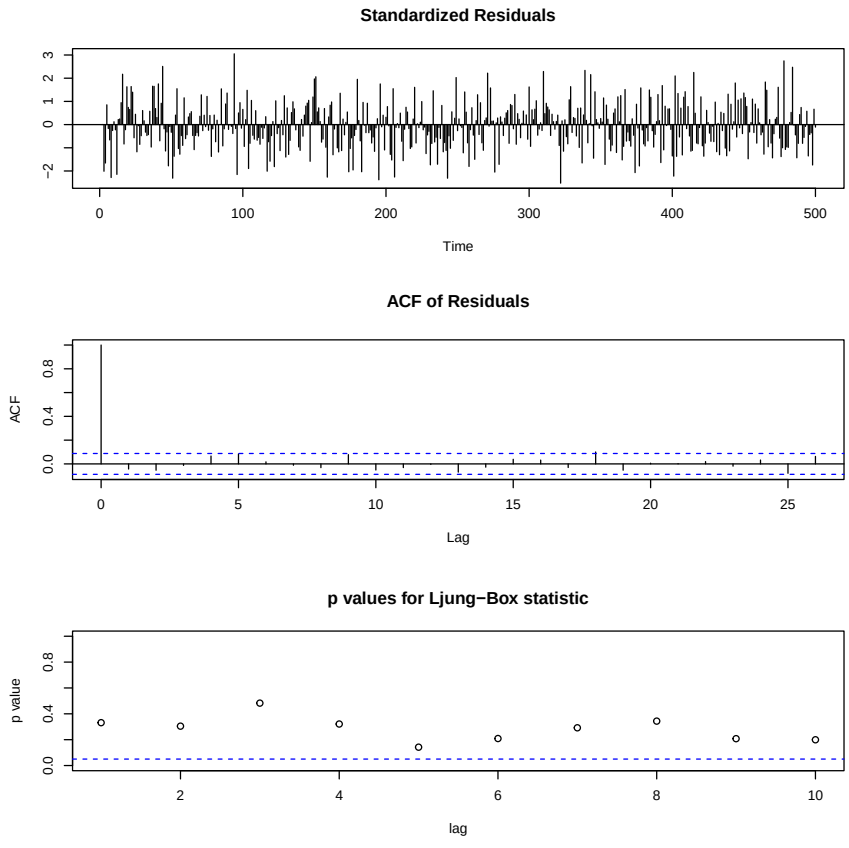
For a good model fit, the sequence of residuals should behave like a realisation of white noise

- The mean of the residuals should be close to 0.
- The spread of the residuals around the mean is constant over time
- Autocorrelations between residuals are negligible

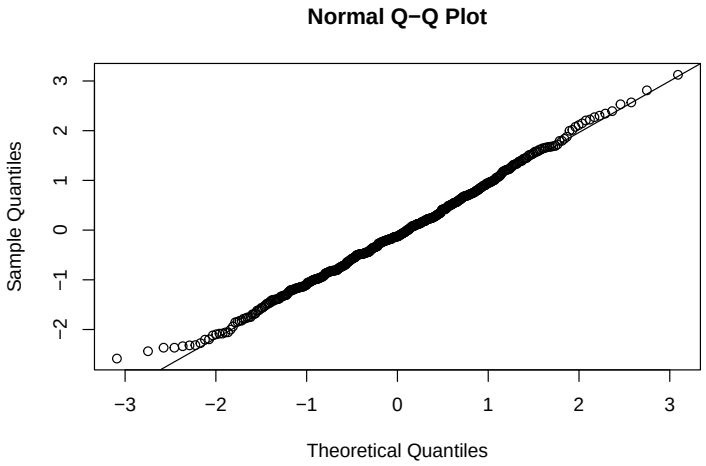
The Ljung-Box statistic can be used to test whether or not the autocorrelation function of a stationary process is zero. If we fail to reject the null hypothesis at level $\alpha$, then the autocorrelation function is 0 at level $\alpha$, and the model fit is adequate.

We now carry out model verification for `ARIMA(0,2,2)`

```
tsdiag(modelA)
```



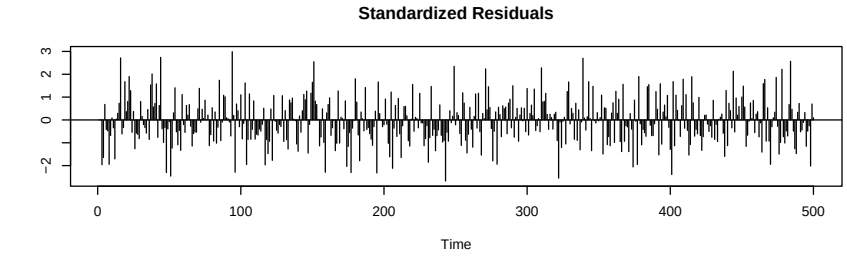

```
qqnorm(modelA$residuals)
qqline(modelA$residuals)
```

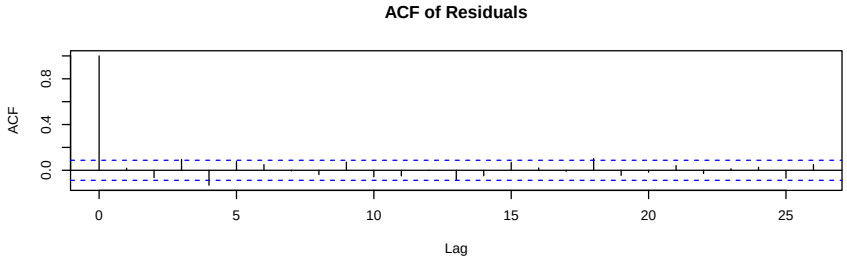


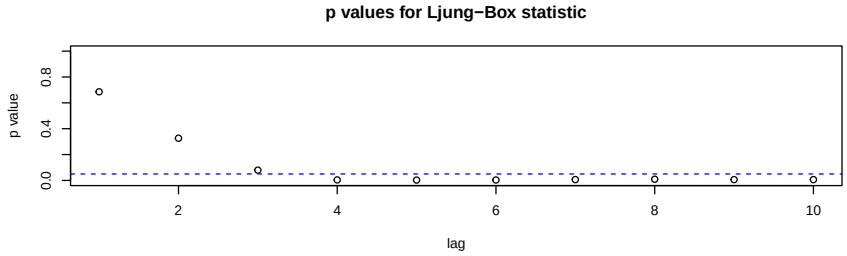

The standardized residuals and the sample ac.f for the residuals are certainly compatible with that of white noise. We notice in the second plot that there is one lag spike above the blue confidence interval, but this does not effect the goodness of fit of the model. This is because it is a single observation that could be due to the so called '5% chance'. We also notice that the normal Q-Q plot of the residuals shows that our assumption of normality is very reasonable. All the p-values for the Ljung-Box statistic are outside the blue confidence interval (the test $\alpha$ value). Therefore, we fail to reject the null hypothesis, so the model fit is adequate.

We also carry out model verification for `ARIMA(3,2,1)`.

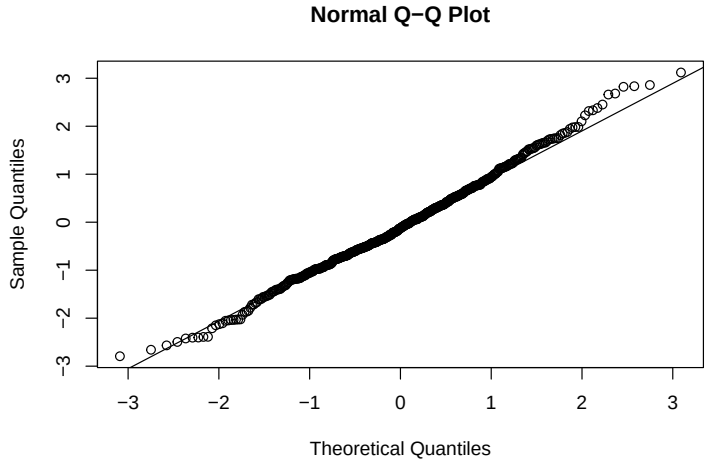
```
tsdiag(modelE)
```




```
qqnorm(modelE$residuals)
qqline(modelE$residuals)
```

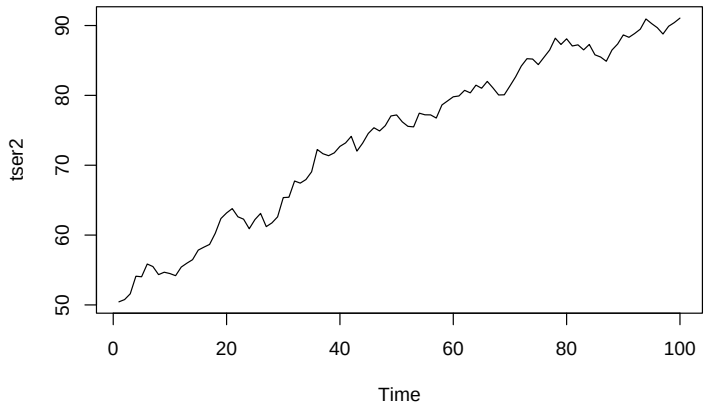



We immediately observe that the p-values for the Ljung-Box statistic are small. This implies that the test statistic is larger than the corresponding critical value. Therefore, the ac.f of the residuals do not correspond to white noise. We reject the null hypothesis, the fit is not adequate. In conclusion, we are confident that `ARIMA(0,2,2)` describes the data in the best way.

# Time series 2

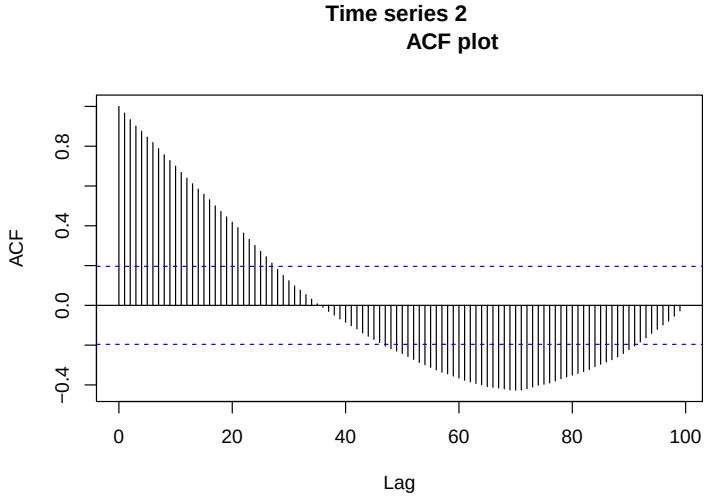
## Initial analysis

```
plot(tser2)
```




Visually, there is a clear trend so time series 2 does not look stationary. We can confirm this by computing the following stationary tests.

```
stationary_tests(tser2, c(7,10,15), ts_plot =FALSE, str_title = 'Time series 2
                 ACF plot', n1=50, n2 =100 )
```




```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  ts_object[1:n1] and ts_object[(n1 + 1):n2]
## D = 0.94, p-value = 2.776e-15
```

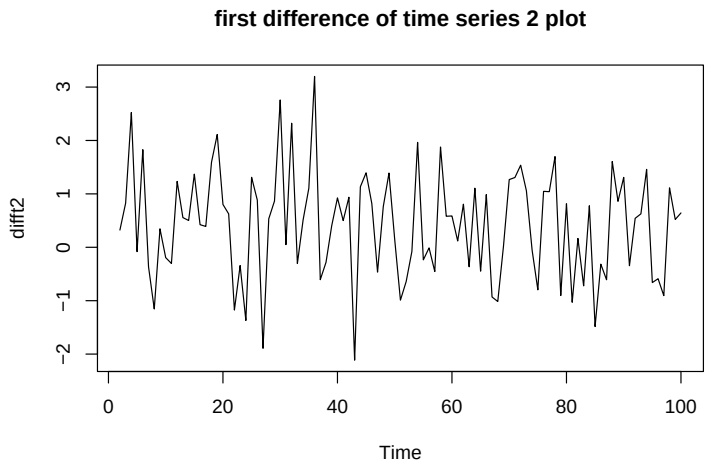
```
## alternative hypothesis: two-sided
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -1.6225, Lag order = 7, p-value = 0.733
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -1.1984, Lag order = 10, p-value = 0.9033
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -0.57441, Lag order = 15, p-value = 0.977
## alternative hypothesis: stationary
```

- **ac.f:** The series does not decay fast enough, and there is a clear trend. This suggests non-stationarity.
- **Kolmogorov-Smirnov test:** small p-value, certainly at least less than $2.776e-15 < 0.05$, also suggests the series is non-stationary, at at least the 5% level.
- **Augmented Dickey-Fuller Test:** The p-value far above 0.05 for all $k = 7, 10, 15$. Therefore this suggests the series is non-stationary.

Therefore, we can conclude that `tser21` is non-stationary. Hence, we have sufficient evidence to rule out a polynomial of the form $m_t = b$, where $b \in \mathbb{R}$.

Now, we compute the first difference of time series 2 (`tser2`).

```
difft2 = diff(tser2)
plot(difft2, main = 'first difference of time series 2 plot')
```



The first difference of time series 2 looks visually stationary. Lets confirm this by computing the usual stationary tests.

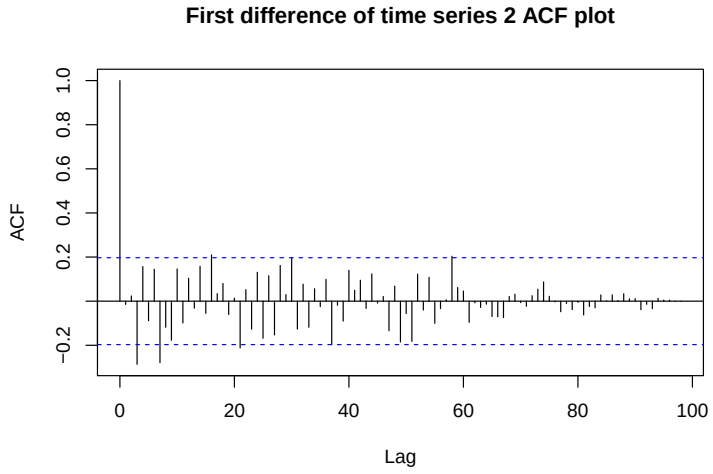
```
stationary_tests(difft2, c(7,10,15), ts_plot =FALSE,
                 str_title = 'First difference of time series 2 ACF plot',
                 n1=50, n2 = 99)
```

```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  ts_object[1:n1] and ts_object[(n1 + 1):n2]
## D = 0.1498, p-value = 0.5687
## alternative hypothesis: two-sided
```

```
## Warning in adf.test(ts_object, k = i): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -5.2362, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
## Warning in adf.test(ts_object, k = i): p-value smaller than printed p-value
```




```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -5.2748, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
```

```
## data:  ts_object
## Dickey-Fuller = -3.1234, Lag order = 15, p-value = 0.1115
## alternative hypothesis: stationary
```

- **ac.f:** From the ac.f plot, we notice that there are no more of these long term correlations or spikes in lags, and the value of the ACF is decreases rapidly. This behavior certainly indicates a stationary process
- **Kolmogorov-Smirnov test:** a large p-value suggests the series is stationary, certainly far beyond the 5% level ($0.05<0.5687$).
- **Augmented Dickey-Fuller Test:** The p-value is below 0.05 for various values of the lag ($k = 7, 10, 15$). Therefore, this suggests the series is stationary.

Therefore, we can conclude that the first difference of time series 2 is stationary. Therefore, we can conclude that the polynomial trend is most likely of degree 1 and of the form $m_t = a + bt$. For the polynomial trend to be of degree 2, we would have expected the first difference of time series 2 to be non-stationary, which clearly is not the case.

We present two possible approaches here to compute the appropriate reconstruction for $X_t$. By computing two different methods, this should help us consolidate our findings.

- **Linear regression:** Calculate the polynomial coefficients $a$ and $b$ using classical statistical inference. One issue with linear regression is that we assume that the errors are i.i.d normally distributed, which may not be the case here.
- **Differencing and inverse differencing:** This approach was motivated from the polynomial trends lecture, were it was discussed that $k$ lag 1 difference operations could be used to remove polynomial trends of degree $k$.

## Linear regression

We compute the following linear regression model with the aim to compute significant estimates to the polynomial coefficients.

```
data = data.frame(t = 1:100, values = tser2)
# Fit a linear regression
linear_regression = lm(values ~ t, data = data)
summary(linear_regression)
```

```
##
## Call:
## lm(formula = values ~ t, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.1904 -2.0935  0.3548  1.7349  4.2750
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 53.228768   0.441732  120.50   <2e-16 ***
## t            0.409750   0.007594   53.96   <2e-16 ***
## ---
```

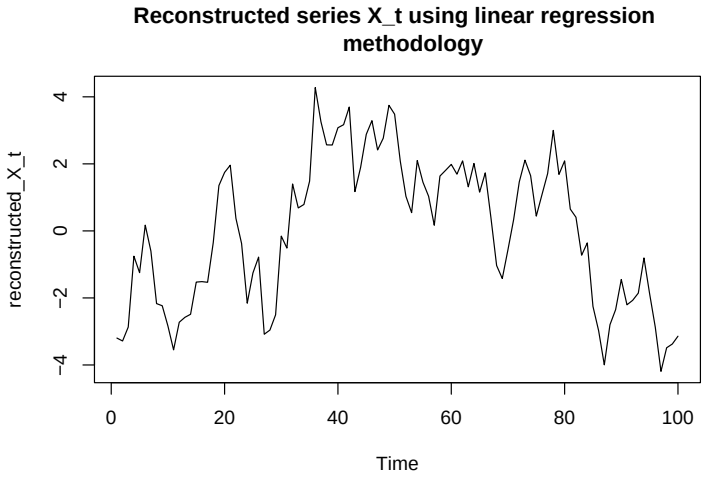
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.192 on 98 degrees of freedom
## Multiple R-squared:  0.9674, Adjusted R-squared:  0.9671
## F-statistic:  2911 on 1 and 98 DF,  p-value: < 2.2e-16
```

We deduce that both the intercept $b$ and the gradient $a$ are significant. Therefore, our deterministic polynomial of order 1 is

$$m_t = 0.409750t + 53.638519$$

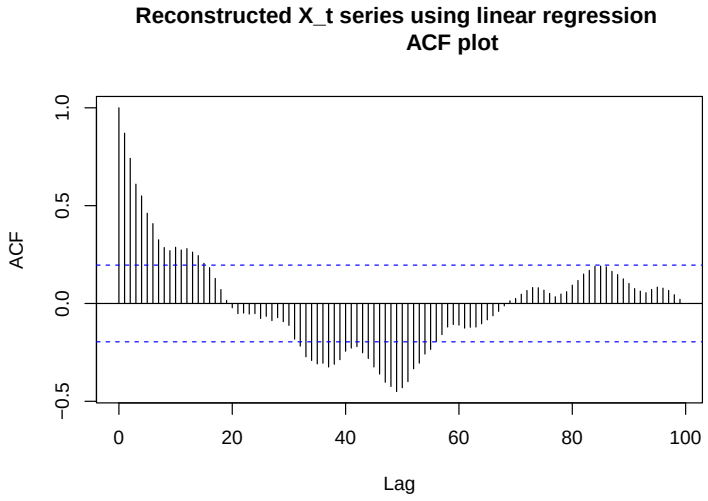
We then subtract this polynomial from `tser2` to obtain a reconsruction of the series $X_t$.

```
predicted_values = predict(linear_regression, newdata = data.frame(t = 1:100))
reconstructed_X_t = tser2 - predicted_values
plot(reconstructed_X_t, main = 'Reconstructed series X_t using linear regression
     methodology')
```




We observe that the reconstructed series does not look visually stationary. We can confirm this by computing the stationary tests

```
stationary_tests(reconstructed_X_t, c(7,10,15), ts_plot =FALSE,
                 str_title = 'Reconstructed X_t series using linear regression
                 ACF plot', n1=20, n2 = 40)
```

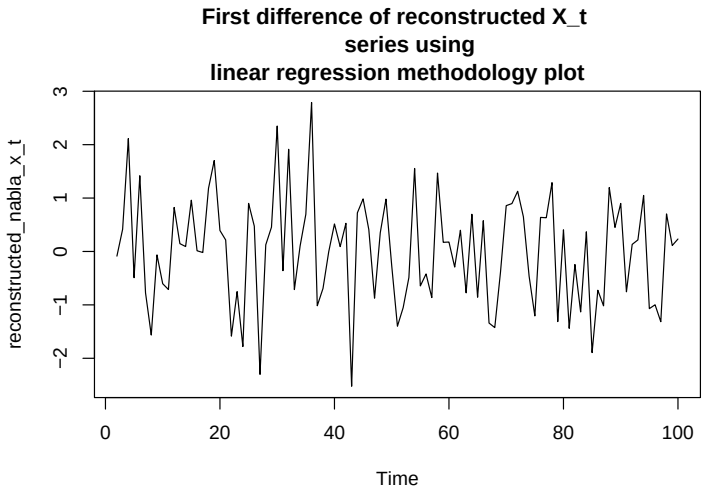



```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  ts_object[1:n1] and ts_object[(n1 + 1):n2]
## D = 0.5, p-value = 0.0123
## alternative hypothesis: two-sided
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -1.6225, Lag order = 7, p-value = 0.733
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -1.1984, Lag order = 10, p-value = 0.9033
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -0.57441, Lag order = 15, p-value = 0.977
## alternative hypothesis: stationary
```

- **ac.f:** The series does not decay fast enough, and there is a clear trend. This suggests non-stationarity.
- **Kolmogorov-Smirnov test:** Multiple of these tests where conducted. Note that due to the 'cyclic' behavior observed this is necessary (discussed in a problem on the week 3 lab sheet). We only include one test and omit the rest (presentation purposes), but confirm that most of them had a small p-value, at least less than 0.05. This also suggests the series is non-stationary, at at least the 5% level.
- **Augmented Dickey-Fuller Test:** The p-value far above 0.05 for all $k = 7, 10, 15$. Therefore this suggests the series is non-stationary.

We therefore compute a second difference of the reconstructed series

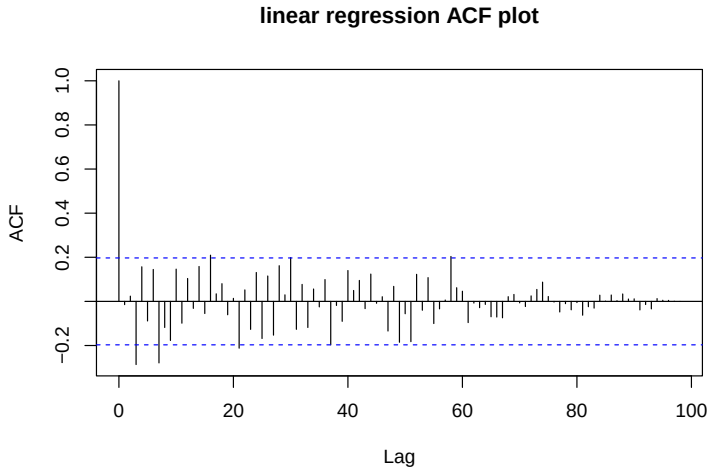
```
reconstructed_nabla_x_t = diff(reconstructed_X_t)
plot(reconstructed_nabla_x_t, main = 'First difference of reconstructed X_t
     series using \nlinear regression methodology plot')
```




This series certainly looks more visually stationary. Lets compute the usual stationary tests,

```
stationary_tests(reconstructed_nabla_x_t, c(7,10,15), ts_plot =FALSE,
                 str_title = 'First difference of reconstructed X_t series using
                 \nlinear regression ACF plot', n1=50, n2 = 99)
```

```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  ts_object[1:n1] and ts_object[(n1 + 1):n2]
## D = 0.1498, p-value = 0.5687
## alternative hypothesis: two-sided

## Warning in adf.test(ts_object, k = i): p-value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -5.2362, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary

## Warning in adf.test(ts_object, k = i): p-value smaller than printed p-value
```




```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -5.2748, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -3.1234, Lag order = 15, p-value = 0.1115
## alternative hypothesis: stationary
```

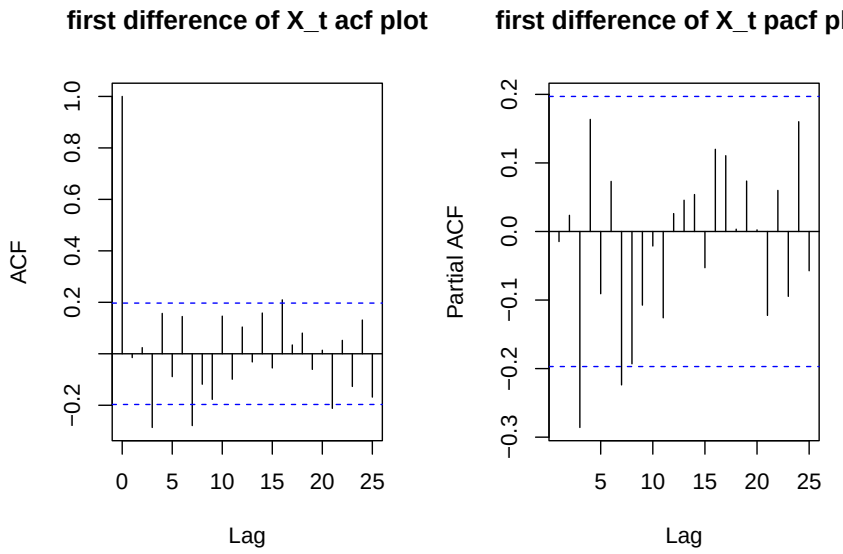
- **ac.f:** From the ac.f plot, we notice that there are no more of these long term correlations or spikes in lags, and the value of the ACF is decreases rapidly. This behavior certainly indicates a stationary process
- **Kolmogorov-Smirnov test:** a large p-value suggests the series is stationary, certainly far beyond the 5% level (0.05<0.5687).
- **Augmented Dickey-Fuller Test:** The p-value is below 0.05 for various values of the lag ($k = 7, 10, 15$). Therefore, this suggests the series is stationary.

Therefore, we can conclude that the second difference of of the reconstructed time series is stationary.

### Estimation of model parameters

We begin by plotting the sample ac.f and the sample partial ac.f for the first difference of the reconstructed $\nabla X_t$

```
par(mfrow = c(1,2))
acf(reconstructed_nabla_x_t,lag.max=25, main = 'first difference of X_t acf plot')
pacf(reconstructed_nabla_x_t,lag.max=25, main = 'first difference of X_t pacf plot')
```




We notice that there does not seem to be a cut off in lag in the sample ac.f or sample partial ac.f. functions respectively in the required range for $p$ and $q$. If the lags at 1 and 2 for the sample ac.f and sample partial ac.f were close to the blue confidence intervals, then we could argue a cut off since there are only 100 simulations. Therefore, both the correlogram suggest that a mixed `ARMA(p,q)` process should be investigated.

We compute model fits for `ARIMA(1,1,1)`, `ARIMA(1,1,2)`,`ARIMA(2,1,1)` and `ARIMA(2,1,2)`.

```
modelA <- arima(reconstructed_X_t, order=c(1,1,1), method = "ML")
```

```
## Warning in arima(reconstructed_X_t, order = c(1, 1, 1), method = "ML"): possible
## convergence problem: optim gave code = 1
```

```
print(modelA)
```

```
##
## Call:
## arima(x = reconstructed_X_t, order = c(1, 1, 1), method = "ML")
##
## Coefficients:
##           ar1     ma1
##       -0.9991  0.9841
## s.e.   0.0046  0.0383
##
## sigma^2 estimated as 0.9162:  log likelihood = -136.92,  aic = 279.83
```

```
modelA$coef -2*sqrt(diag(modelA$var.coef))
```

```
##        ar1        ma1
## -1.0083502  0.9075883
```

```
modelA$coef +2*sqrt(diag(modelA$var.coef))
```

```
##        ar1        ma1
## -0.9897994  1.0607077
```

```
modelB <- arima(reconstructed_X_t,order=c(1,1,2), method = "ML" )
print(modelB)
```

```
##
## Call:
## arima(x = reconstructed_X_t, order = c(1, 1, 2), method = "ML")
##
## Coefficients:
##           ar1     ma1     ma2
##       -0.9978  1.1371  0.1627
## s.e.   0.0068  0.1046  0.1039
##
## sigma^2 estimated as 0.8976:  log likelihood = -135.83,  aic = 279.66
```

```
modelB$coef -2*sqrt(diag(modelB$var.coef))
```

```
##         ar1         ma1         ma2
## -1.01139087  0.92786435 -0.04510173
```

```
modelB$coef +2*sqrt(diag(modelB$var.coef))
```

```
##        ar1        ma1        ma2
## -0.9841847  1.3462657  0.3704636
```

```
# ARMA(2,1)
modelC <- arima(reconstructed_X_t,order=c(2,1,1))
print(modelC)
```

```
##
## Call:
## arima(x = reconstructed_X_t, order = c(2, 1, 1))
##
## Coefficients:
##          ar1      ar2      ma1
##       0.8363  -0.0742  -0.8784
## s.e.  0.1373   0.1039   0.0977
##
## sigma^2 estimated as 0.9995:  log likelihood = -140.53,  aic = 289.06
```

```
modelC$coef -2*sqrt(diag(modelC$var.coef))
```

```
##        ar1        ar2        ma1
##  0.5616509 -0.2820046 -1.0737017
```

```
modelC$coef +2*sqrt(diag(modelC$var.coef))
```

```
##        ar1        ar2        ma1
##  1.1109705  0.1335407 -0.6830060
```

```r
# ARMA(2,2)
modelD <- arima(reconstructed_X_t,order=c(2,1,2))
```

```
## Warning in arima(reconstructed_X_t, order = c(2, 1, 2)): possible convergence
## problem: optim gave code = 1
```

```r
print(modelD)
```

```
##
## Call:
## arima(x = reconstructed_X_t, order = c(2, 1, 2))
##
## Coefficients:
##           ar1     ar2      ma1      ma2
##       -0.0661  0.9329  -0.0112  -0.9880
## s.e.   0.0468  0.0465   0.0544   0.0542
##
## sigma^2 estimated as 0.8852:  log likelihood = -136.02,  aic = 282.05
```

```r
modelD$coef -2*sqrt(diag(modelD$var.coef))
```

```
##        ar1        ar2        ma1        ma2
## -0.1596520  0.8399988 -0.1200006 -1.0963515
```

```r
modelD$coef +2*sqrt(diag(modelD$var.coef))
```

```
##         ar1         ar2         ma1         ma2
##  0.02746685  1.02581783  0.09762076 -0.87969138
```

It is very important for the models to have significant parameters. The only model tested that does not have at least 1 parameter that is not significant is `ARIMA(1,1,1)`. We therefore choose to test some models with $q$ and $p$ equal to zero. Unfortunately, `AR(1)`, `AR(2)`, `MA(1)` and `MA(2)` had the same problem. `ARIMA(1,1,1)` has the second lowest AIC out of all the models tested, and the variance is closer to 1 than the majority of the models tested. We also choose to investigate `ARIMA(0,1,0)`.

```r
modelE <- arima(reconstructed_X_t,order=c(0,1,0))
print(modelE)
```

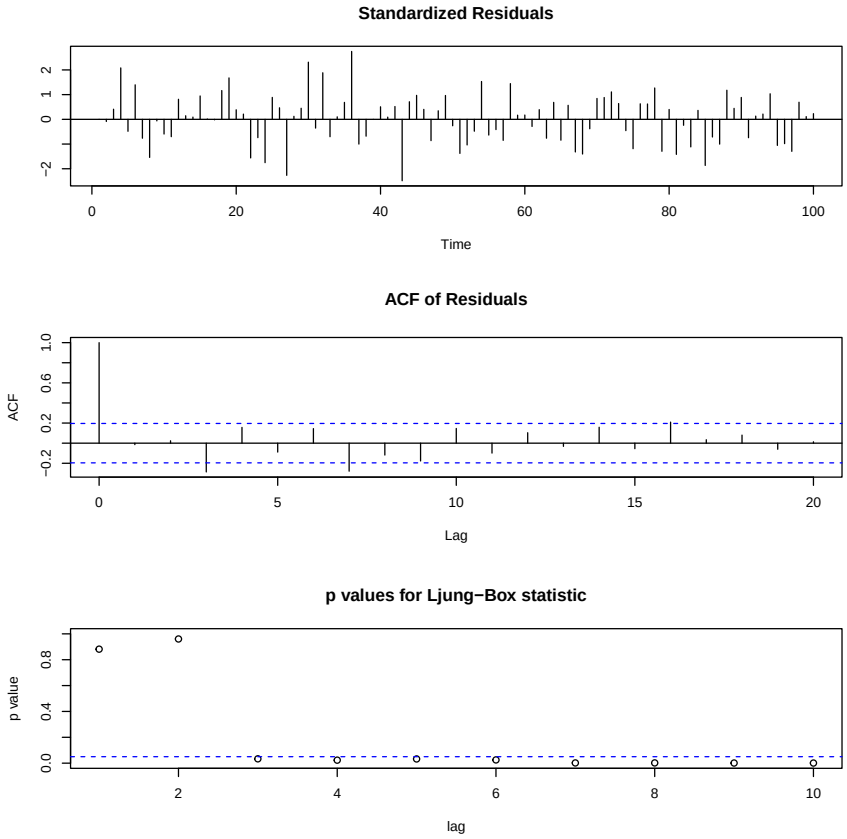
```
##
## Call:
## arima(x = reconstructed_X_t, order = c(0, 1, 0))
##
##
## sigma^2 estimated as 1.031:  log likelihood = -141.99,  aic = 285.98
```

We note that the variance is closer to 1 for `ARIMA(0,1,0)` than `ARIMA(1,1,1)`. However, the AIC is the fourth lowest out of all the models tested.

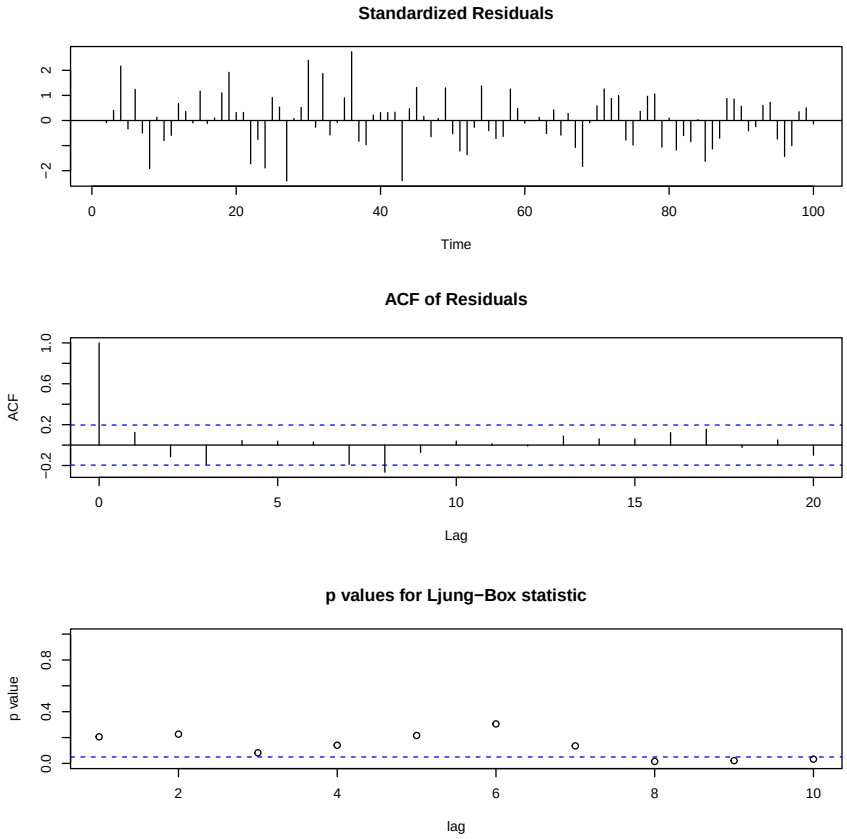
### Agreement with data

We check the model diagnostics for `ARIMA(0,1,0)` and `ARIMA(1,1,1)`

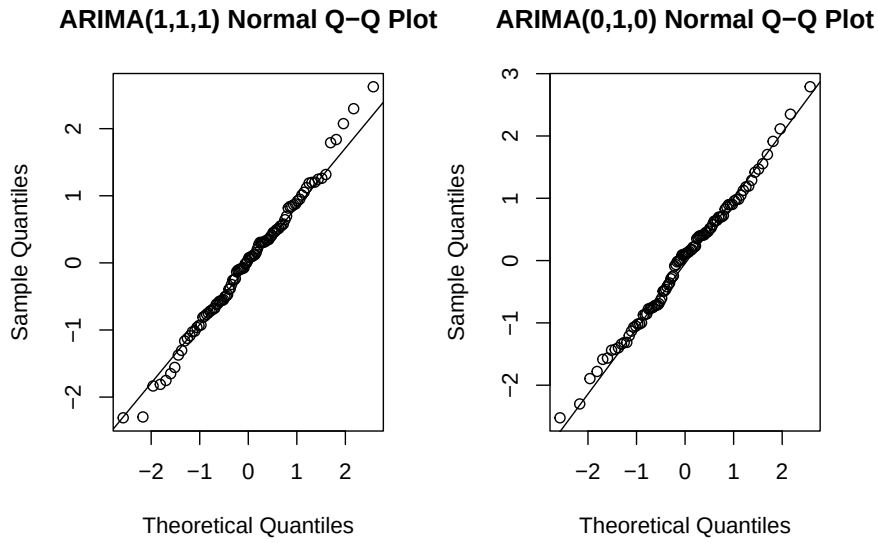
```
tsdiag(modelE)
```




```
tsdiag(modelA)
```




```
par(mfrow = c(1,2))
qqnorm(modelA$residuals, main='ARIMA(1,1,1) Normal Q-Q Plot')
qqline(modelA$residuals)
qqnorm(modelE$residuals, main='ARIMA(0,1,0) Normal Q-Q Plot')
qqline(modelE$residuals)
```



There are many p-values below the blue confidence interval for `ARIMA(0,1,0)`. This implies the model fit is inadequate. We note that `ARIMA(1,1,1)` also has some significant p-values for the Ljung-Box statistic. Although the standardized residuals and the sample ac.f for the residuals are compatible with that of white noise (the one spike for the sample ac.f for the residuals could be due to the so called '5% chance' so this is good enough).

We investigate the fit for the various models tried above and discover that all the models with $d = 1$ have the same issue apart from `ARIMA(1,1,2)`, which has no significant p-value for the Ljung-Box statistic. Since `ARIMA(1,1,2)` has one in-significant parameter, if that parameter was significant at a higher alpha level, we might consider this model. It would also be worth seeing if the p-values of the Ljung-Box statistic are just below 5% for `ARIMA(1,1,1)`.

```
modelB$coef -qnorm(0.925)*sqrt(diag(modelB$var.coef))
```

```
##         ar1         ma1         ma2
## -1.00757882  0.98648955  0.01312609
```

```
modelB$coef +qnorm(0.925)*sqrt(diag(modelB$var.coef))
```

```
##        ar1        ma1        ma2
## -0.9879967  1.2876405  0.3122357
```

```
# Box.test function found by ?tsdiag() command
for (i in c(8,9,10)){
  box_test = Box.test(residuals(modelA), lag = i, type = "Ljung-Box")
  print(box_test$p.value)
}
```

```
## [1] 0.01541229
## [1] 0.02147383
## [1] 0.03282682
```

The model parameter associated with `ma2` are not significant at 10%, but are significant at 15%. Two of the significant p-values are quite far below the 5% level. For both `ARIMA(1,1,1)` and `ARIMA(1,1,2)`, the standardized residuals and the sample ac.f for the residuals are certainly compatible with that of white noise, and the normality assumption is about okay.

Therefore, there is no clear answer to the model that should be picked in this situation. However, we judge that in this case, if we absolutely had to pick a model, we would pick `ARIMA(1,1,1)` with $m_t = 0.409750t + 53.638519$. This is because there are very few observations in `tser2`. Therefore, it is possible that we could be seeing three significant p-values of the Ljung box statistic due to random fluctuations and not a significant auto correlation. We also note that the AIC is approximately the same as the model with the lowest AIC (`ARIMA(1,1,2)`), the variance is closer to 1 than `ARIMA(1,1,2)`, and it is the only model with significant parameters.

We now try an alternative approach, with the hope of more certainty

## Differencing and inverse differencing

We now choose to investigate the second approach. First, we must decide what the value fo the $d$ parameter is. This is done by contradiction.

**Recomendation:** We encourage the reader to read the simulation case studies in the appendix for more details

Initially, we assume for possible contradiction that the series $X_t$ is stationary with mean 0 and standard deviation 1. This is a sensible assumption based on two lectures: information spoken about time series 2 in lecture 16 (Box Jenkins method lecture); and spoken advice when applying corollary 1.1 in the polynomial trends lecture.

$$Y_t = X_t + at + b$$

We now use corollary 1.1 to compute the first difference of time series 2,

$$\nabla Y_t = \nabla X_t + a$$

The mean of the series $\nabla Y_t$ is $a$. Therefore, rearranging and writing in components

$$\nabla Y_t - a = [y_1 - y_0 - a, ..., y_n - y_{n-1} - a] = [x_1 - x_0, ..., x_n - x_{n-1}]$$

We can then take the cumulative sum using the `cumsum` function in R

$$\text{cumsum}([x_1 - x_0, ..., x_n - x_{n-1}]) = [x_1 - x_0, ..., x_n - x_0]$$

Note that this time series has $n$ entries. This is effectively indefinite 'integration' with the constant of integration being $-x_0$. Therefore, this time series will certainly have the same structure as the original time series $X_t$, just with one less entry, and shifted by $-x_0$ on the y-axis. The function `inverse_diff` computes this (more more details see the appendix).

```
# function to compute the reverse difference operation
# with capability to estimate 'a' if 'n' simulations is low to produce a better
# reconstruction
inverse_diff = function(difft, eps, precision){
  if (precision == FALSE){
    val = mean(difft) + eps
  } else{
    val = round(mean(difft),precision) + eps
  }
  original_series <- cumsum(difft-val)
  orginal_series = ts(original_series)
  return(orginal_series)
}
# just taking the mean is okay for this series, permuting a by +- \eps,
# and changing the precision doesnt help
reconstructed_plus_constant = inverse_diff(diff(tser2), 0, FALSE)
```

Since we assumed that the mean of $X_t$ is 0, we can compute the component $x_0$ to produce a reconstruction of the series $X_t$

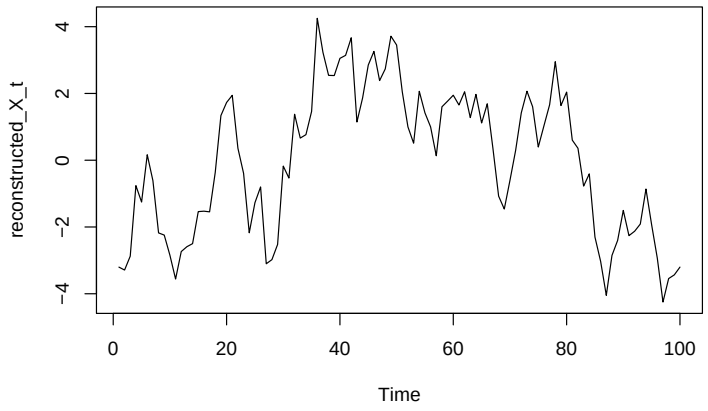
$$\text{mean}([x_1 - x_0, ..., x_n - x_0]) = -x_0$$

Therefore, we can then compute the reconstruction for the series $X_t$ starting at $t = 1$,

$$[x_1 - x_0, ..., x_n - x_0] + [x_0, ..., x_0] = [x_1, ..., x_n]$$

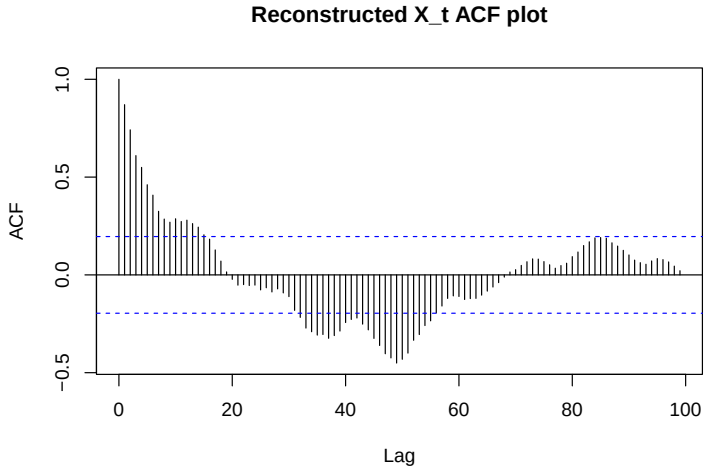
and then place the computed $x_0$ before the entry $x_1$ in the time series $[x_1, ..., x_n]$. Therefore we have a fully reconstructed $X_t$ of original length $n + 1$. The following code does this to produce the reconstructed series `reconstructed_X_t`.

```r
# computing x_0
x_0 = (-1)*mean(reconstructed_plus_constant)
# computing the reconstructed Arima(1,0,1) series
reconstructed_X_t = as.ts(c(x_0, reconstructed_plus_constant+x_0))
# plotting for visual inspection
plot(reconstructed_X_t)
```




Visually we are not convinced that this is stationary. This implies a contraction, and hence $X_t$ must not be stationary, and therefore $d = 1$. We can confirm this by computing the normal stationary tests,

```r
stationary_tests(reconstructed_X_t, c(7,10,15), ts_plot =FALSE,
                 str_title = 'Reconstructed X_t ACF plot',
                 n1=20, n2 = 40)
```




```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  ts_object[1:n1] and ts_object[(n1 + 1):n2]
## D = 0.5, p-value = 0.0123
```

```
## alternative hypothesis: two-sided
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -1.6225, Lag order = 7, p-value = 0.733
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -1.1984, Lag order = 10, p-value = 0.9033
## alternative hypothesis: stationary
##
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_object
## Dickey-Fuller = -0.57441, Lag order = 15, p-value = 0.977
## alternative hypothesis: stationary
```

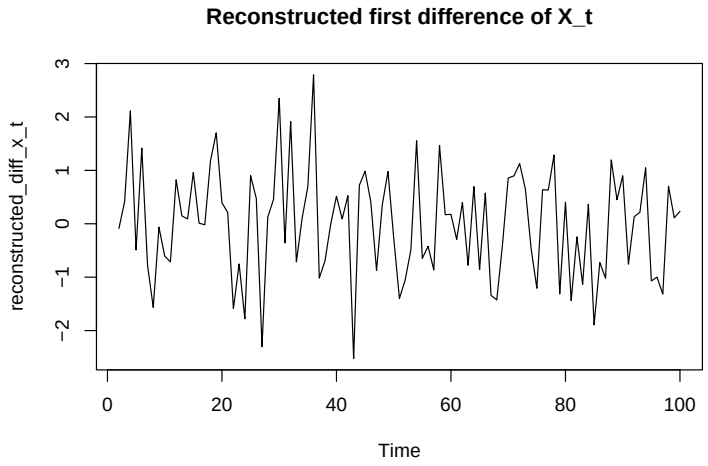
- **ac.f:** The series does not decay fast enough, and there is a clear trend. This suggests non-stationarity.
- **Kolmogorov-Smirnov test:** Multiple of these tests where conducted. Note that due to the 'cyclic' behavior observed this is necessary (discussed in a problem on the week 3 lab sheet). We only include one test and omit the rest (presentation purposes), but confirm that most of them had a small p-value, at least less than 0.05. This also suggests the series is non-stationary, at at least the 5% level.
- **Augmented Dickey-Fuller Test:** The p-value far above 0.05 for all $k = 7, 10, 15$. Therefore this suggests the series is non-stationary.

Therefore we confirm that we have a contraction, $d \neq 0$. Therefore, we assume that $\nabla X_t$ is a stationary series with mean 0. With this assumption, we observe that the mean of $\nabla Y_t$ is $a$, and hence we have a reconstruction for the series $\nabla X_t$ as follows,

$$\nabla X_t = \nabla Y_t - a$$

We observe that the series $\nabla X_t$ must be stationary, since we are only shifting the values on the y-axis by $a$, and we conformed that $\nabla Y_t$ was stationary when deducing that the polynomial trend must be of degree 1, $m_t = at + b$.

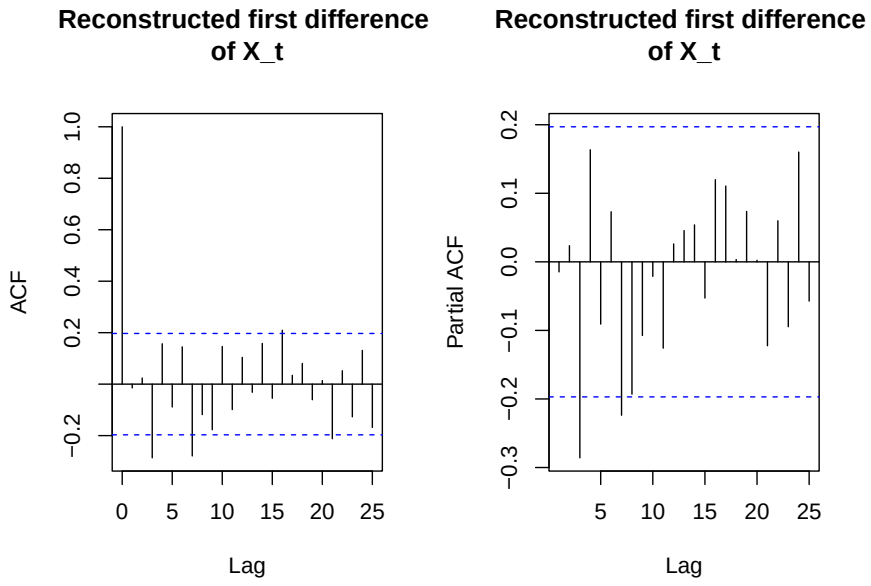
```
reconstructed_diff_x_t = diff(tser2) - mean(diff(tser2))
plot(reconstructed_diff_x_t, main = 'Reconstructed first difference of X_t')
```




### Estimation of model parameters q and p

We begin by plotting the sample ac.f and the sample partial ac.f for the first difference of the reconstructed $\nabla X_t$

```
par(mfrow = c(1,2))
acf(reconstructed_diff_x_t,lag.max=25, main = 'Reconstructed first difference \nof X_t')
pacf(reconstructed_diff_x_t,lag.max=25, main = 'Reconstructed first difference \nof X_t')
```




We notice that there does not seem to be a cut off in lag either of the correlograms in the range for $p$ and $q$ specified. If the lags at 1 and 2 for the sample ac.f and sample partial ac.f were close to the blue confidence intervals, then we could argue a cut off since there are only 100 simulations. Therefore, both the correlogram suggest that a mixed `ARMA(p,q)` process should be investigated. We use Occam's Razor and begin our investigations with the model `ARIMA(1,1,1)`.

We also test the models `ARIMA(1,1,2)`, `ARIMA(2,1,2)` and `ARIMA(2,1,1)`

```r
# d=0 since we do not have the series $X_t$, but know the series $\nabla X_t$
modelF <- arima(reconstructed_diff_x_t, order=c(1,0,1), include.mean=FALSE, method = "ML")
```

```
## Warning in arima(reconstructed_diff_x_t, order = c(1, 0, 1), include.mean =
## FALSE, : possible convergence problem: optim gave code = 1
```

```r
print(modelF)
```

```
##
## Call:
## arima(x = reconstructed_diff_x_t, order = c(1, 0, 1), include.mean = FALSE,
##     method = "ML")
##
## Coefficients:
##           ar1     ma1
##       -0.9990  0.9833
## s.e.   0.0049  0.0381
##
## sigma^2 estimated as 0.9167:  log likelihood = -136.92,  aic = 279.83
```

```r
modelF$coef -2*sqrt(diag(modelF$var.coef))
```

```
##        ar1        ma1
## -1.0087589  0.9071117
```

```r
modelF$coef +2*sqrt(diag(modelF$var.coef))
```

```
##        ar1        ma1
## -0.9891724  1.0594129
```

```r
modelG <- arima(reconstructed_diff_x_t, order=c(1,0,2), include.mean=FALSE, method = "ML")
print(modelG)
```

```
##
## Call:
## arima(x = reconstructed_diff_x_t, order = c(1, 0, 2), include.mean = FALSE,
##     method = "ML")
##
## Coefficients:
##           ar1     ma1     ma2
##       -0.9978  1.1370  0.1627
## s.e.   0.0068  0.1046  0.1039
##
## sigma^2 estimated as 0.8976:  log likelihood = -135.83,  aic = 279.66
```

```r
modelG$coef -2*sqrt(diag(modelG$var.coef))
```

```
##         ar1         ma1         ma2
## -1.01142675  0.92783733 -0.04502788
```

```r
modelG$coef +2*sqrt(diag(modelG$var.coef))
```

```
##        ar1        ma1        ma2
## -0.9841167  1.3462491  0.3704853
```

```r
modelA <- arima(reconstructed_diff_x_t, order=c(2,0,2), include.mean=FALSE)
```

```
## Warning in arima(reconstructed_diff_x_t, order = c(2, 0, 2), include.mean =
## FALSE): possible convergence problem: optim gave code = 1
```

```r
print(modelA)
```

```
##
## Call:
## arima(x = reconstructed_diff_x_t, order = c(2, 0, 2), include.mean = FALSE)
##
## Coefficients:
##          ar1     ar2      ma1      ma2
##       -0.0649  0.9338  -0.0131  -0.9867
## s.e.   0.0472  0.0468   0.0563   0.0560
##
## sigma^2 estimated as 0.8857:  log likelihood = -136.03,  aic = 282.05
```

```r
modelA$coef -2*sqrt(diag(modelA$var.coef))
```

```
##        ar1        ar2        ma1        ma2
## -0.1594106  0.8401838 -0.1257659 -1.0987818
```

```r
modelA$coef +2*sqrt(diag(modelA$var.coef))
```

```
##         ar1         ar2         ma1         ma2
##  0.02956217  1.02741930  0.09955214 -0.87468027
```

```r
modelB <- arima(reconstructed_diff_x_t,order=c(2,0,1), include.mean=FALSE)
print(modelB)
```

```
##
## Call:
## arima(x = reconstructed_diff_x_t, order = c(2, 0, 1), include.mean = FALSE)
##
## Coefficients:
##          ar1      ar2      ma1
##       0.8363  -0.0742  -0.8784
## s.e.  0.1374   0.1039   0.0977
##
## sigma^2 estimated as 0.9996:  log likelihood = -140.53,  aic = 289.06
```

```r
modelB$coef -2*sqrt(diag(modelB$var.coef))
```

```
##        ar1        ar2        ma1
##  0.5616220 -0.2819828 -1.0737680
```

```r
modelB$coef +2*sqrt(diag(modelB$var.coef))
```

```
##        ar1        ar2        ma1
##  1.1110283  0.1335603 -0.6829440
```

We observe essentially the same results as the linear regression case, and hence form similar conclusions.

Only one of the models above have significant parameters (`ARIMA(1,1,1)`). All other models with $d = 1$ have at least one insignificant parameter (same as linear regression case). The model with the lowest AIC is `ARIMA(1,1,2)`. Although the AIC is only marginally higher for `ARIMA(1,1,1)`, and the variance is closer to 1 than `ARIMA(1,1,2)`. We also test `ARIMA(0,1,0)`

```r
#d=0 since we are fitting the first difference below.
modelC <- arima(reconstructed_diff_x_t,order=c(0,0,0), include.mean=FALSE)
print(modelC)
```

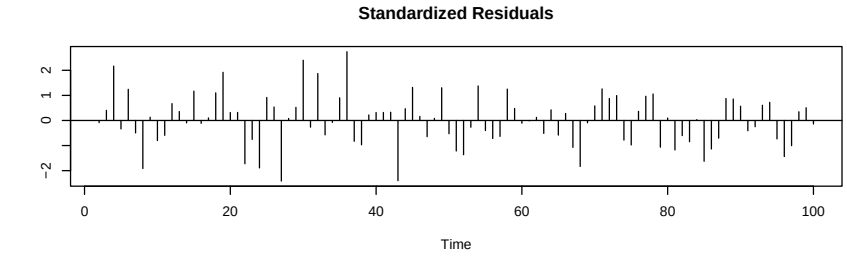
```
##
## Call:
## arima(x = reconstructed_diff_x_t, order = c(0, 0, 0), include.mean = FALSE)
##
##
## sigma^2 estimated as 1.031:  log likelihood = -141.99,  aic = 285.98
```

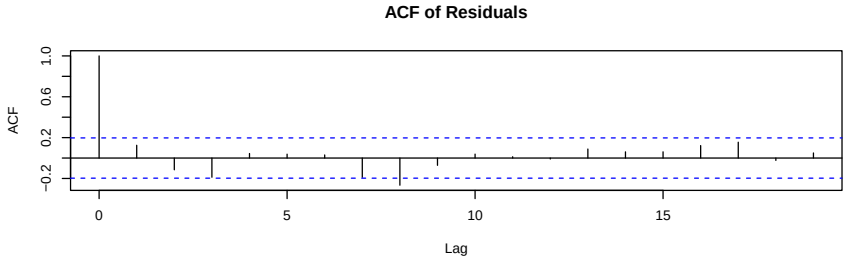
`ARIMA(0,1,0)` has the fourth lowest AIC value, but a variance closer to 1 than `ARIMA(1,1,1)` and `ARIMA(1,1,2)`.

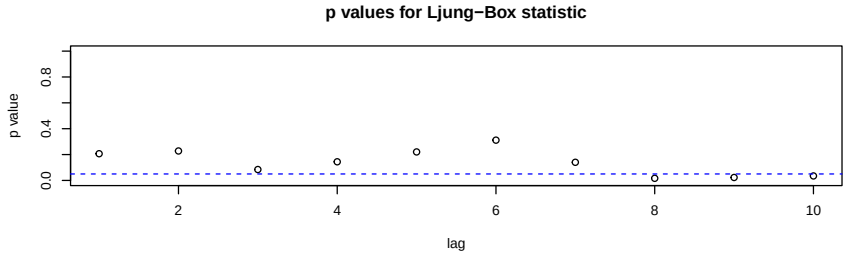
### Agreement with data

We compute model diagnostics for `ARIMA(1,1,1)` and `ARIMA(0,1,0)`.

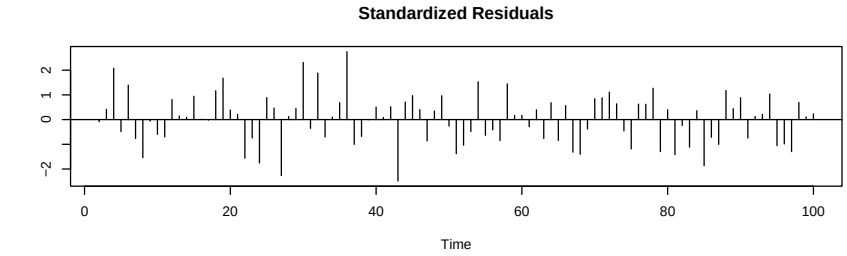
```r
# ARIMA(2,1,2)
tsdiag(modelF)
```

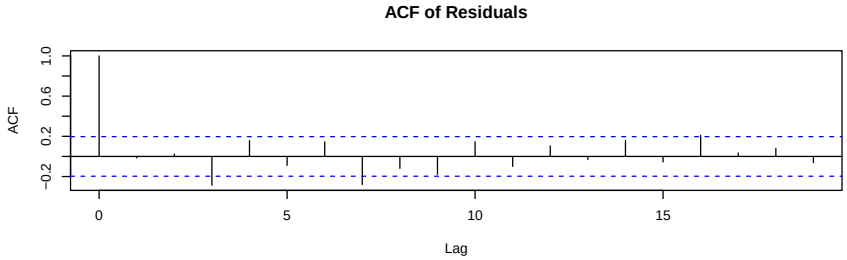


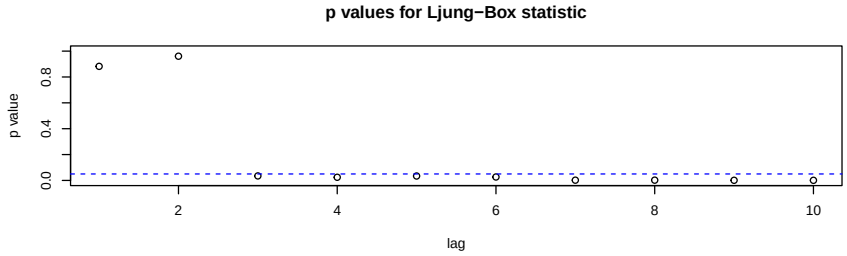



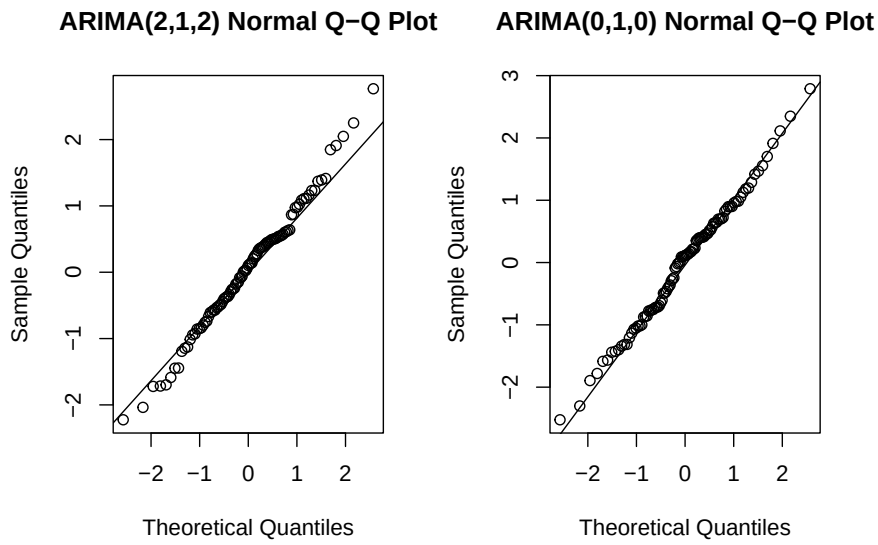

```
# ARIMA(0,1,0)
tsdiag(modelC)
```



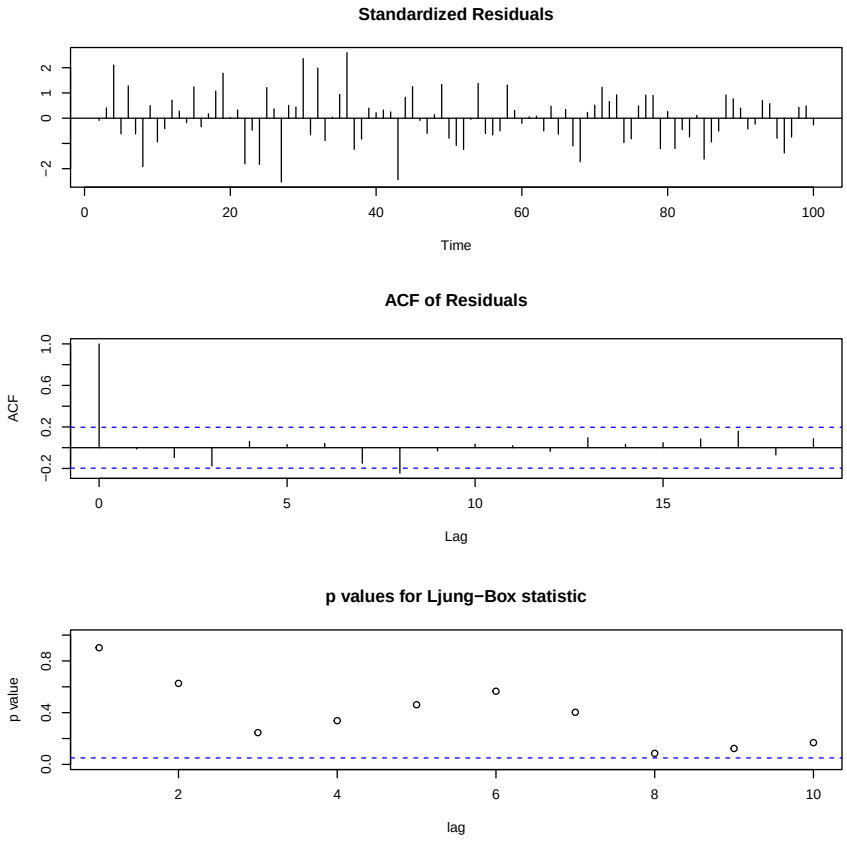

```
par(mfrow = c(1,2))
qqnorm(modelA$residuals, main='ARIMA(2,1,2) Normal Q-Q Plot')
qqline(modelA$residuals)
qqnorm(modelC$residuals, main='ARIMA(0,1,0) Normal Q-Q Plot')
qqline(modelC$residuals)
```



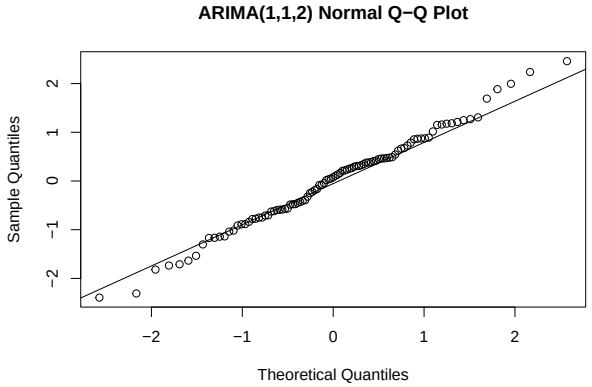
We exclude `ARIMA(0,1,0)` since the p-values for lags 4 and onwards are significant. We also observed that `ARIMA(1,1,1)` also has some significant p-values for the Ljung-Box statistic at lags $8, 9, 10$. For `ARIMA(1,1,1)`, the standardized residuals and the sample ac.f for the residuals are certainly compatible with that of white noise (only one spike for sample ac.f for the residuals outside confidence interval could just be explained by the so called '5% chance'). The normality assumption is reasonable.

We discover that all other models with $d = 1$ have the some insignificant p-values except for `ARIMA(1,1,2)`, which has no significant p-values for the Ljung-Box statistic.

```
tsdiag(modelG)
```




```
qqnorm(modelG$residuals, main='ARIMA(1,1,2) Normal Q-Q Plot')
qqline(modelG$residuals)
```




For `ARIMA(1,1,2)`, the standardized residuals and the sample ac.f for the residuals are certainly compatible with that of white noise. Only one spike for sample ac.f for the residuals is outside confidence interval. This could be explained by random fluctuations due to low number of samples in `tser2`. The normality assumption also seems reasonable.

Similar to the case in linear regression, since `ARIMA(1,1,2)` has one in-significant parameter, if that parameter was significant at a higher alpha level, we might consider this model. It would also be worth seeing if the p-values of the Ljung-Box statistic are just below 5% for `ARIMA(1,1,1)`.

```
modelG$coef -qnorm(0.925)*sqrt(diag(modelG$var.coef))
```

```
##         ar1         ma1         ma2
## -1.00760014  0.98646398  0.01319263
```

```
modelG$coef +qnorm(0.925)*sqrt(diag(modelG$var.coef))
```

```
##        ar1        ma1        ma2
## -0.9879433  1.2876224  0.3122648
```

```
# Box.test function found by ?tsdiag() command
for (i in c(8,9,10)){
  box_test = Box.test(residuals(modelF), lag = i, type = "Ljung-Box")
  print(box_test$p.value)
}
```

```
## [1] 0.0163034
## [1] 0.02271482
## [1] 0.03464689
```

The model parameter associated with `ma2` are not significant at 10%, but are significant at 15%. Two of the significant p-values are quite par below the 5% level. Two of the three p-values are quite far away from the 5% level.

We unfortunately reach exactly the same conclusions as for linear regression: there is no clear answer to the model that should be picked in this situation. If we absolutely had to pick a model, we would pick `ARIMA(1,1,1)` with a polynomial trend of degree 1 (same as linear regression). The AIC is approximately the same as the model with the lowest AIC (`ARIMA(1,1,2)`), the variance is closer to 1 than `ARIMA(1,1,2)`, and it is the only model with significant parameters. We could consider the three significant p-values from Ljung box-test as realizations from random fluctuations and not a significant autocorrelation function due to the small number of observations in `tser2`.

# Appendix

## Time series 2: A mock simulation study

### Case d = 0,

We simulate an `ARIMA(1,0,1)` process, `arima101`

$$X_t = \alpha X_{t-1} + Z_t + \beta Z_{t-1}$$

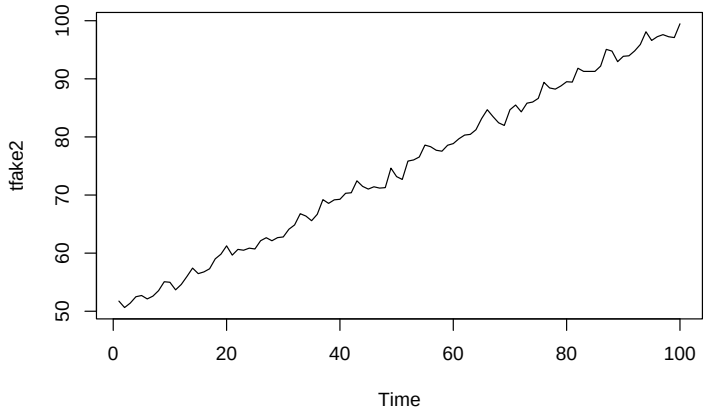
and is stationary with standard deviation equal to 1 and mean equal to 0. This selection is motivated by lecture 16 (Box Jenkins Lecture), where it was stated that $\mu$ will be 0 for the stationary process.

We then generate a fake series in the form of time series 2 `tfake2`,

$$Y_t = X_t + a + bt$$

These series are generated using the function `tfake2_sim`. The task is as follows, given only the series $Y_t$, reconstruct the series $X_t$.

```
nsim=100
d=0
tfake2_sim = function(nsim, d, print_model = FALSE){
  # computes the t
  t= 0:(nsim-1+d)
  # computes a simulation of arima values, where the coefficients are selected
  # as in lab 3
  x3 <- arima.sim(n=nsim,model=list(ma=c(0.193),ar=c(0.064),order=c(1,d,1)),sd=1)
  # does the user want to see a fitted model output?
  if (print_model == TRUE){
    modelA <- arima(x3,order=c(1,d,1), include.mean = FALSE)
    print(modelA)
  }
  # makes the fake time series by adding polynomial trend
  tfake2 = x3 + 0.5*t + 50 #+ 18*t^2
  # returns the required data in a list
  return(list(x3,tfake2,modelA))
}
output = tfake2_sim(nsim, d, print_model = FALSE)
arima101 = output[[1]]
tfake2 = output[[2]]
modelA = output[[3]]
plot(tfake2)
```




```
print(paste0('this is the mean of ARIMA(1,0,1): ',mean(arima101)))
```

```
## [1] "this is the mean of ARIMA(1,0,1): -0.15209509381682"
```

We are only doing 100 simulations here so the sample mean is not exactly 0, if you increase `nsim` you will observe the sample mean gets closer to 0.

By corollary 1.1 in the lecture notes, the first difference is

$$\nabla Y_t = a + \nabla X_t$$

and since $X_t$ is stationary, the mean of $\nabla Y_t$ is $a$. Therefore, rearranging and writing in components

$$\nabla Y_t - a = [y_1 - y_0 - a, ..., y_n - y_{n-1} - a] = [x_1 - x_0, ..., x_n - x_{n-1}]$$

We can then take the cumulative sum,

$$cumsum([x_1 - x_0, ..., x_n - x_{n-1}]) = [x_1 - x_0, ..., x_n - x_0]$$

This is a vector of $n$ components (the original length was $n + 1$), where we have effectively computed an indefinite integral (the $-x_0$ is the additional constant at the end of the summation integral...). This was hinted at in the polynomial trends lecture, and hence was the motivation. Since `nsim` is small, computing the mean of $\nabla Y_t$ is not very accurate. Of course, this is 'solved' for larger and larger simulations (the reader can test this if they like by increasing the `nsim` parameter). Empirically for low simulations, it was found that the best to estimate `a` by trying a few values, inside the confidence interval, to minimize the reconstruction error. This can be done even if you have little knowledge about the series $X_t$, since we assumed it is stationary, therefore has no trend, so the gradient of the line of best fit must be equal to 0, with the intercept being the mean. Therefore, this could be set up as an optimization problem to calculate the best `a`.

Therefore we note that the `eps` and `precision` are not necessary for large `nsim`, but do help for low `nsim`.

```
inverse_diff = function(difft, eps, precision){
  if (precision == FALSE){
    val = mean(difft) + eps
  } else{
    val = round(mean(difft),precision) + eps
```

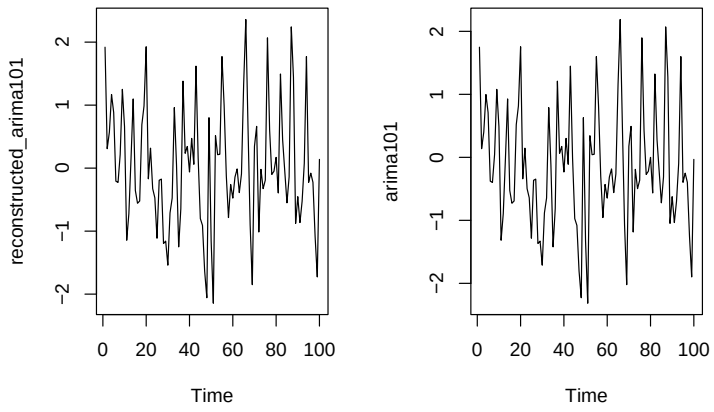
```
  }
  original_series <- cumsum(difft-val)
  orginal_series = ts(original_series)
  return(orginal_series)
}
reconstructed_plus_constant = inverse_diff(diff(tfake2), 0, 1)
```

We can compute $x_0$ by remembering that we assumed the mean of the stationary process $X_t$ is 0.

$$mean([x_1 - x_0, ..., x_n - x_0]) = -x_0$$

Therefore, we can compute the constructed series,

```
# computing x_0
x_0 = (-1)*mean(reconstructed_plus_constant)
# computing the reconstructed Arima(1,0,1) series
reconstructed_arima101 = as.ts(c(x_0, reconstructed_plus_constant+x_0))
# plotting for visual inspection
par(mfrow = c(1,2))
plot(reconstructed_arima101)
plot(arima101)
```




```
print('pure difference between simulations')
```

```
## [1] "pure difference between simulations"
```

```
summary(arima101 - reconstructed_arima101)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.1713 -0.1713 -0.1713 -0.1713 -0.1713 -0.1713
```

```
print('difference between simulations with shifted mean')
```

```
## [1] "difference between simulations with shifted mean"
```

```
summary(arima101 - mean(arima101) - reconstructed_arima101 + mean(reconstructed_arima101))
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -9.676e-15 -3.050e-15  2.047e-16 -8.330e-19  2.994e-15  9.697e-15
```

```
print(paste0('b value: ', tfake2[1] - x_0, ' which is approximately ', round(tfake2[1] - x_0, 0)))
```

```
## [1] "b value: 49.8287073905372 which is approximately 50"
```

We notice that they are statistically structurally identical, with differences coming from series shifts in the y-axis due to the low simulation value `nsim` (i.e. increasing `nsim` forces the mean of `arima101` to be closer to 0).

### Case d = 1,

We simulate an `ARIMA(1,1,1)` process $X_t$, stored in the time series object `arima111`. We now formulate the series $Y_t$ as,

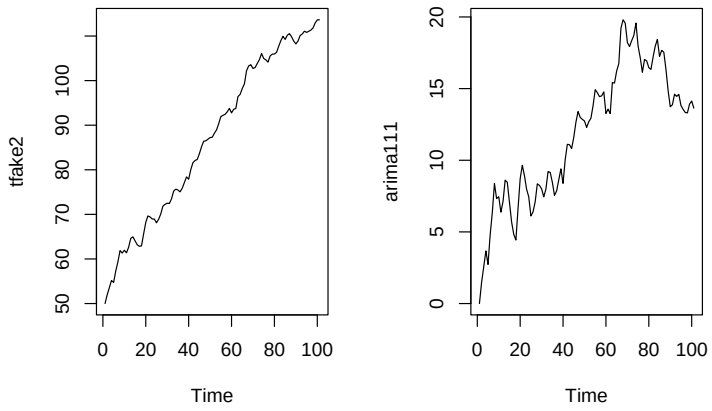
$$Y_t = X_t + at + b$$

As before, `nsim` simulations of $Y_t$ are produced in time series object `tfake2`. Since $d > 0$, this is non-stationary, but the first difference is stationary with mean 0 and standard deviation of 1. We compute the first difference using corollary 1.1.

$$\nabla Y_t = a + \nabla X_t$$

Our aim this time is to reconstruct $\nabla X_t$ to sufficient accuracy, only using $Y_t$.

```
nsim=100
d=1
output = tfake2_sim(nsim, d, print_model = FALSE)
arima111 = output[[1]]
tfake2 = output[[2]]
modelA = output[[3]]
par(mfrow = c(1,2))
plot(tfake2)
plot(arima111)
```



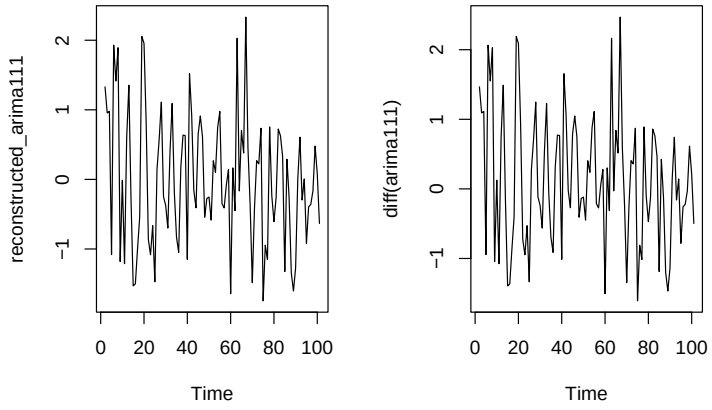

```
print(paste0('this is the mean of the first difference of ARIMA(1,1,1): ',mean(diff(arima101))))
```

```
## [1] "this is the mean of the first difference of ARIMA(1,1,1): -0.0179872099156389"
```

Since the mean of the series $\nabla Y_t$ is $a$, we can reconstruct the series $\nabla X_t$ as

$$\nabla X_t = \nabla Y_t - a$$

```
# compute the reconstruction
reconstructed_arima111= diff(tfake2) - mean(diff(tfake2))
# visual comparison between reconstructed and original
par(mfrow = c(1,2))
plot(reconstructed_arima111)
plot(diff(arima111))
```




```
print('pure difference between simulations')
```

```
## [1] "pure difference between simulations"
```

```
summary(reconstructed_arima111 - diff(arima111))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.1364 -0.1364 -0.1364 -0.1364 -0.1364 -0.1364
```

```
print('difference between simulations with shifted mean')
```

```
## [1] "difference between simulations with shifted mean"
```

```
summary(reconstructed_arima111 - mean(reconstructed_arima111)
        - diff(arima111) + mean(diff(arima111)) )
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -1.069e-14 -4.427e-15  2.776e-17  1.693e-17  3.580e-15  1.446e-14
```

We notice that they are statistically structurally identical, with differences coming from series shifts in the y-axis due to the low simulation value `nsim` (i.e. increasing `nsim` forces the mean of the difference of time series `arima111` to be closer to 0).