MA30085, Time Series - Coursework

Morgan Rhys Williams

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 computational 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

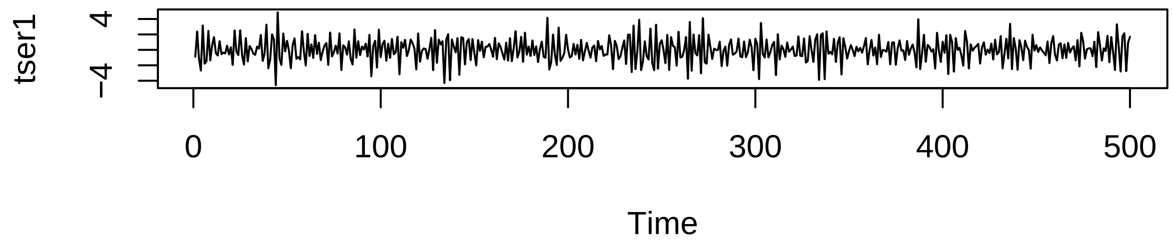
Below is the R chunk needed to get your work started, as instructed earlier. Once executed, please add all necessary text, code and graphics (please, make sure the size of the graphics window is big enough for me to check details) under the line **SOLUTION**. The final work should be uploaded on Crowdmark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

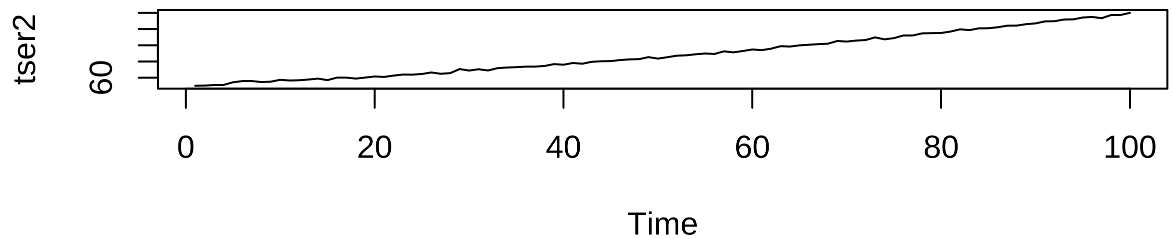
```r
#************************#
#!!! VERY IMPORTANT !!!
#************************#
# Please, replace the "2" inside set.seed() with your
# unique seed. Failure to do so might result in your work
# being penalised
set.seed(77631)

#************************#
#!!! 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]]

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

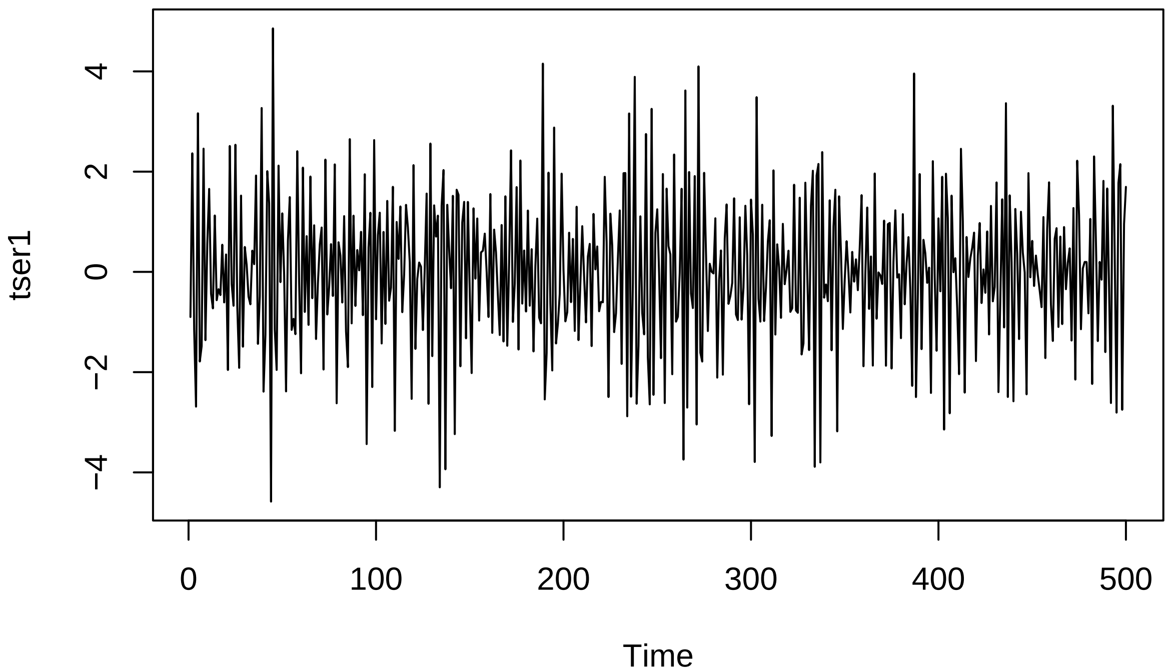





```r
# Back to one plot per window
par(mfrow=c(1,1))
```

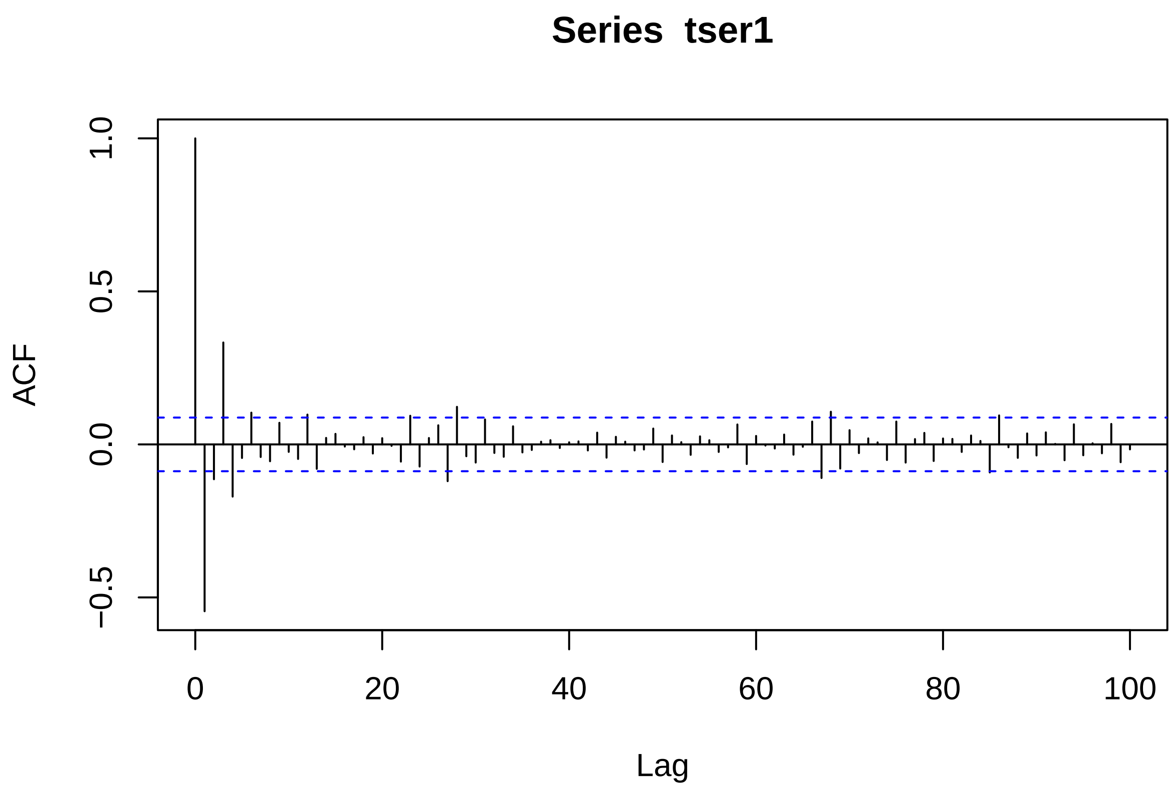
SOLUTION 1

```r
plot(tser1)
```



The visual impression of 'tser1' is of a stationary series. We plot the sample ACF. A stationary series will have a fast (exponential/sinusoidal) decay or a truncation, while a slow decay is typical of a non-stationary series.

```
acf(tser1, lag.max = 100)
```



The sample ACF plot exhibits fast sinusoidal decay, indicative of a stationary series. Conducting a Kolmogorov-Smirnov test will help determine whether or not 'tser1' is stationary and thus follows a Gaussian distribution. The null and alternative hypotheses are as follows:

H_0: the time series is stationary H_1: the time series is non-stationary

```
x1 <- tser1[1:250]
y1 <- tser1[251:500]

ks.test(x1,y1)
```

```
##
##  Asymptotic two-sample Kolmogorov-Smirnov test
##
## data:  x1 and y1
## D = 0.08, p-value = 0.4005
## alternative hypothesis: two-sided
```

A high p-value means we accept the null and thus 'tser1' is a stationary series.

Another test for stationarity we can conduct is the *Augmented Dickey-Fuller test* (ADF test). It targets the presence of one or more unit roots in the series. The null and alternative hypotheses are as follows:

H_0: the time series is non-stationary H_1: the time series is stationary

```
library(tseries)
```

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

```r
adf.test(tser1)
```

```
## Warning in adf.test(tser1): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  tser1
## Dickey-Fuller = -10.676, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```r
adf.test(tser1, k = 10)
```

```
## Warning in adf.test(tser1, k = 10): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  tser1
## Dickey-Fuller = -9.8927, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
```

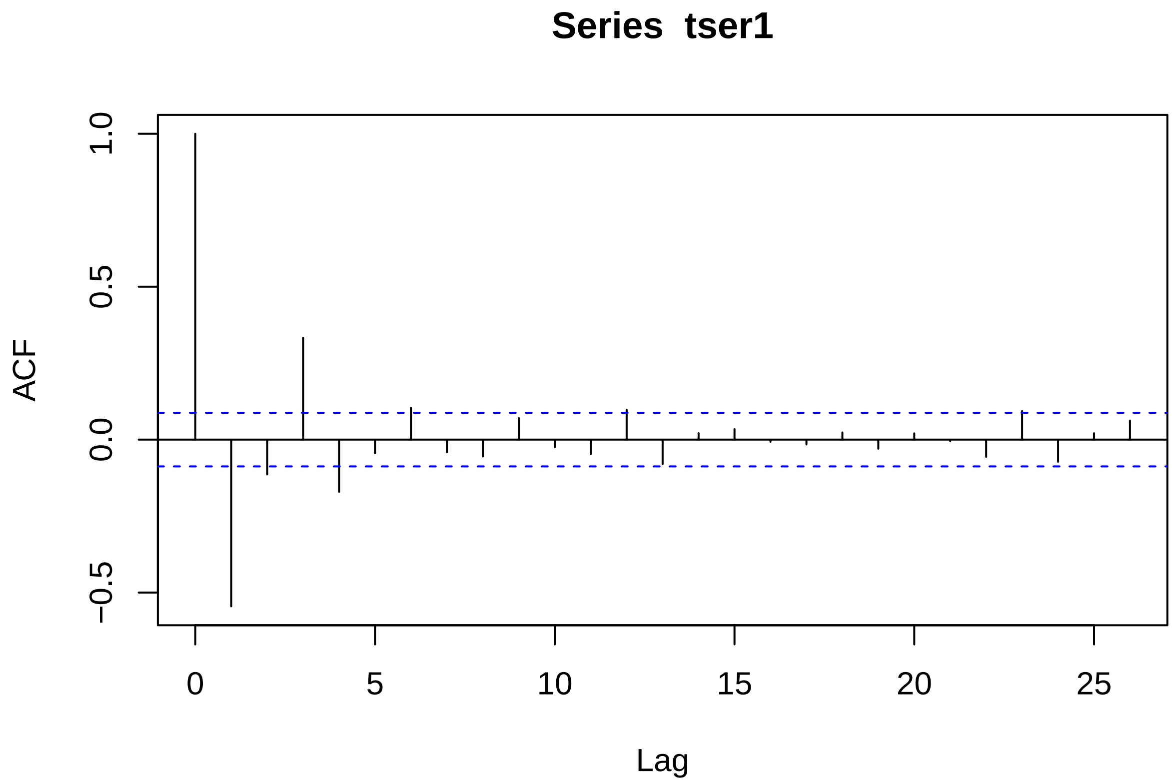
```r
adf.test(tser1, k = 15 )
```

```
## Warning in adf.test(tser1, k = 15): p-value smaller than printed p-value
```

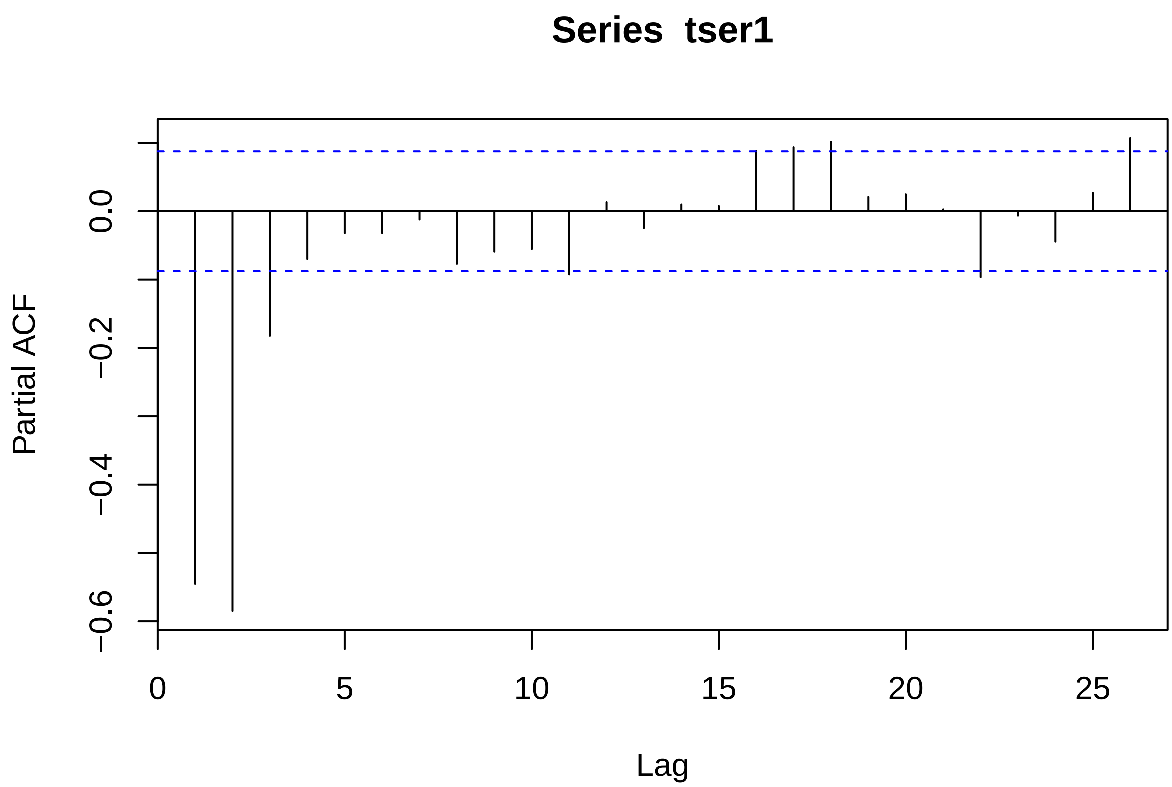
```
##
##  Augmented Dickey-Fuller Test
##
## data:  tser1
## Dickey-Fuller = -6.3368, Lag order = 15, p-value = 0.01
## alternative hypothesis: stationary
```

The test, consistently, returns a small p-value confirming, once again, that the series is stationary. We have ensured that our time series is stationary, and that 'tser1' is an ARIMA(p, d, q) process with $d = 0$. To determine p and q, we use the interplay of the sample ACF and the sample PACF.

```r
acf(tser1)
```




```r
pacf(tser1)
```



Both the ACF and PACF are gradually decreasing. Therefore an ARMA(p, q) model is appropriate for this series. To determine the order p, q, we start with the simplest ARMA model, ARMA(1,1), and compare against other ARMA models of varying orders p, q. An important piece of information coming from these models is whether or not the 95% confidence intervals exclude 0. Confidence intervals excluding 0 means the model has significant parameters and is therefore an appropriate model for the time series data.

```
arma11 <- arima(tser1, order = c(1, 0, 1))
print(arma11)
```

```
##
## Call:
## arima(x = tser1, order = c(1, 0, 1))
##
## Coefficients:
##           ar1      ma1  intercept
##       -0.2544  -0.7410    -0.0040
## s.e.   0.0502   0.0326     0.0101
##
## sigma^2 estimated as 1.171:  log likelihood = -749.54,  aic = 1507.09
```

```
arma11$coef - 2 * sqrt(diag(arma11$var.coef)) #95% Confidence Interval
```

```
##         ar1         ma1   intercept
## -0.35483501 -0.80614288 -0.02411397
```

```
arma11$coef + 2 * sqrt(diag(arma11$var.coef))
```

```
##         ar1         ma1   intercept
```

```
## -0.15394279 -0.67584704  0.01610733
```

All confidence intervals of the ARMA(1,1) model exclude 0, thus all parameters are significant and the model is appropriate.

```r
arma12 <- arima(tser1, order = c( 1, 0, 2))
print(arma12)
```

```
##
## Call:
## arima(x = tser1, order = c(1, 0, 2))
##
## Coefficients:
##          ar1      ma1     ma2  intercept
##       0.0105  -1.0478  0.3173    -0.0038
## s.e.  0.0930   0.0813  0.0760     0.0131
##
## sigma^2 estimated as 1.142:  log likelihood = -743.24,  aic = 1496.49
```

```r
arma12$coef - 2 * sqrt(diag(arma12$var.coef)) #95% Confidence Interval
```

```
##         ar1         ma1         ma2   intercept
## -0.17552732 -1.21032457  0.16538460 -0.02988536
```

```r
arma12$coef + 2 * sqrt(diag(arma12$var.coef))
```

```
##         ar1         ma1         ma2   intercept
##  0.19655370 -0.88531369  0.46924186  0.02233856
```

All confidence intervals of the ARMA(1,2) model do not exclude 0, thus not all parameters are significant and the model is deemed not appropriate.

```r
arma13 <- arima(tser1, order = c( 1, 0, 3))
print(arma13)
```

```
##
## Call:
## arima(x = tser1, order = c(1, 0, 3))
##
## Coefficients:
##           ar1      ma1      ma2     ma3  intercept
##       -0.6059  -0.3434  -0.4508  0.3119    -0.0038
## s.e.   0.0960   0.0885   0.0907  0.0483     0.0152
##
## sigma^2 estimated as 1.106:  log likelihood = -735.17,  aic = 1482.35
```

```r
arma13$coef - 2 * sqrt(diag(arma13$var.coef)) #95% Confidence Interval
```

```
##         ar1         ma1         ma2         ma3   intercept
## -0.79786513 -0.52036036 -0.63213744  0.21531399 -0.03417806
```

```r
arma13$coef + 2 * sqrt(diag(arma13$var.coef))
```

```
##         ar1         ma1         ma2         ma3   intercept
## -0.41402644 -0.16642130 -0.26947194  0.40852039  0.02658146
```

All confidence intervals of the ARMA(1,3) model exclude 0, thus all parameters are significant and the model is appropriate.

```r
arma21 <- arima(tser1, order = c(2, 0, 1))
print(arma21)
```

```
##
## Call:
## arima(x = tser1, order = c(2, 0, 1))
##
## Coefficients:
##           ar1      ar2      ma1  intercept
##       -0.6437  -0.4657  -0.3582     -0.004
## s.e.   0.0648   0.0531   0.0728      0.014
##
## sigma^2 estimated as 1.056:  log likelihood = -723.88,  aic = 1457.76
```

```r
arma21$coef - 2 * sqrt(diag(arma21$var.coef)) #95% Confidence Interval
```

```
##         ar1         ar2         ma1   intercept
## -0.77330448 -0.57201393 -0.50381436 -0.03206146
```

```r
arma21$coef + 2 * sqrt(diag(arma21$var.coef))
```

```
##         ar1         ar2         ma1   intercept
## -0.51410943 -0.35942122 -0.21250139  0.02403225
```

All confidence intervals of the ARMA(2,1) model exclude 0, thus all parameters are significant and the model is appropriate.

```r
arma22 <- arima(tser1, order = c( 2, 0, 2))
print(arma22)
```

```
##
## Call:
## arima(x = tser1, order = c(2, 0, 2))
##
## Coefficients:
##           ar1      ar2      ma1      ma2  intercept
##       -0.7118  -0.4671  -0.2882  -0.0835    -0.0040
## s.e.   0.1019   0.0509   0.1078   0.1026     0.0133
##
## sigma^2 estimated as 1.055:  log likelihood = -723.55,  aic = 1459.09
```

```r
arma22$coef - 2 * sqrt(diag(arma22$var.coef)) #95% Confidence Interval
```

```
##         ar1         ar2         ma1         ma2   intercept
## -0.91552787 -0.56887582 -0.50391361 -0.28867887 -0.03058415
```

```r
arma22$coef + 2 * sqrt(diag(arma22$var.coef))
```

```
##         ar1         ar2         ma1         ma2   intercept
## -0.50810752 -0.36530758 -0.07252454  0.12167376  0.02255574
```

All confidence intervals of the ARMA(2,2) model do not exclude 0, thus not all parameters are significant and the model is deemed not appropriate.

```r
arma23 <- arima(tser1, order = c( 2, 0, 3))
print(arma23)
```

```
##
## Call:
```

```
## arima(x = tser1, order = c(2, 0, 3))
##
## Coefficients:
##           ar1      ar2      ma1     ma2      ma3  intercept
##       -0.7042  -0.5655  -0.3005  0.0244  -0.1274    -0.0041
## s.e.   0.1072   0.0891   0.1196  0.1351   0.1023     0.0121
##
## sigma^2 estimated as 1.052:  log likelihood = -722.81,  aic = 1459.62
```

```
arma23$coef - 2 * sqrt(diag(arma23$var.coef)) #95% Confidence Interval
```

```
##         ar1         ar2         ma1         ma2         ma3   intercept
## -0.91852338 -0.74369102 -0.53970918 -0.24590903 -0.33189891 -0.02829593
```

```
arma23$coef + 2 * sqrt(diag(arma23$var.coef))
```

```
##         ar1         ar2         ma1         ma2         ma3   intercept
## -0.48979863 -0.38731667 -0.06122918  0.29465837  0.07716209  0.02010818
```

All confidence intervals of the ARMA(2,3) model do not exclude 0, thus not all parameters are significant and the model is deemed not appropriate.

```
arma31 <- arima(tser1, order = c( 3, 0, 1))
print(arma31)
```

```
##
## Call:
## arima(x = tser1, order = c(3, 0, 1))
##
## Coefficients:
##           ar1      ar2    ar3      ma1  intercept
##       -0.2297  -0.0867  0.283  -0.7624    -0.0042
## s.e.   0.1030   0.0989  0.081   0.0861     0.0106
##
## sigma^2 estimated as 1.049:  log likelihood = -722.24,  aic = 1456.47
```

```
arma31$coef - 2 * sqrt(diag(arma31$var.coef)) #95% Confidence Interval
```

```
##         ar1         ar2         ar3         ma1   intercept
## -0.43569220 -0.28441836  0.12113369 -0.93449325 -0.02541223
```

```
arma31$coef + 2 * sqrt(diag(arma31$var.coef))
```

```
##         ar1         ar2         ar3         ma1   intercept
## -0.02380680  0.11102438  0.44495955 -0.59027397  0.01696929
```

All confidence intervals of the ARMA(3,1) model do not exclude 0, thus not all parameters are significant and the model is deemed not appropriate.

```
arma32 <- arima(tser1, order = c( 3, 0, 2))
print(arma32)
```

```
##
## Call:
## arima(x = tser1, order = c(3, 0, 2))
##
## Coefficients:
##           ar1     ar2     ar3      ma1     ma2  intercept
##       -0.0081  0.0046  0.3459  -0.9961  0.1463    -0.0043
```

```
## s.e.   0.1686  0.0893  0.0707   0.1715  0.1095     0.0105
##
## sigma^2 estimated as 1.046:  log likelihood = -721.46,  aic = 1456.91
```

```r
arma32$coef - 2 * sqrt(diag(arma32$var.coef)) #95% Confidence Interval
```

```
##         ar1         ar2         ar3         ma1         ma2   intercept
## -0.34525668 -0.17389720  0.20456197 -1.33917759 -0.07271096 -0.02537801
```

```r
arma32$coef + 2 * sqrt(diag(arma32$var.coef))
```

```
##         ar1         ar2         ar3         ma1         ma2   intercept
##  0.32896131  0.18316281  0.48720743 -0.65302574  0.36540111  0.01670013
```

All confidence intervals of the ARMA(3,2) model do not exclude 0, thus not all parameters are significant and the model is deemed not appropriate.

```r
arma33 <- arima(tser1, order = c( 3, 0, 3))
print(arma33)
```

```
##
## Call:
## arima(x = tser1, order = c(3, 0, 3))
##
## Coefficients:
##           ar1      ar2     ar3      ma1     ma2      ma3  intercept
##       -0.0024  -0.0777  0.3250  -1.0057  0.2474  -0.0708    -0.0044
## s.e.   0.1880   0.1452  0.0803   0.1927  0.1843   0.0947     0.0104
##
## sigma^2 estimated as 1.045:  log likelihood = -721.19,  aic = 1458.38
```

```r
arma33$coef - 2 * sqrt(diag(arma33$var.coef)) #95% Confidence Interval
```

```
##         ar1         ar2         ar3         ma1         ma2         ma3
## -0.37839866 -0.36802857  0.16443687 -1.39120428 -0.12122326 -0.26022873
##   intercept
## -0.02518776
```

```r
arma33$coef + 2 * sqrt(diag(arma33$var.coef))
```

```
##        ar1        ar2        ar3        ma1        ma2        ma3  intercept
##  0.3736059  0.2126073  0.4856341 -0.6202378  0.6161231  0.1186151  0.0164843
```

All confidence intervals of the ARMA(3,3) model do not exclude 0, thus not all parameters are significant and the model is deemed not appropriate. The estimates of the data variance for each model are close to its correct value 1, with ARIMA$(3, 0, 3)$ being closest.

Comparing AIC of the candidate models.

```r
AIC_arma11 <- AIC(arma11)
AIC_arma12 <- AIC(arma12)
AIC_arma13 <- AIC(arma13)
AIC_arma21 <- AIC(arma21)
AIC_arma22 <- AIC(arma22)
AIC_arma23 <- AIC(arma23)
AIC_arma31 <- AIC(arma31)
AIC_arma32 <- AIC(arma32)
AIC_arma33 <- AIC(arma33)
AIC_values <- c(AIC_arma11, AIC_arma12, AIC_arma13, AIC_arma21, AIC_arma22, AIC_arma23, AIC_arma31, AIC_
```

```r
Candidates <- c('ARMA(1,1)', 'ARMA(1,2)', 'ARMA(1,3)', 'ARMA(2,1)', 'ARMA(2,2)', 'ARMA(2,3)', 'ARMA(3,1)
IC_table <- data.frame(Candidates, AIC_values)
sorted_AIC <- IC_table[order(AIC_values),]
sorted_AIC
```

```
##   Candidates AIC_values
## 7  ARMA(3,1)   1456.474
## 8  ARMA(3,2)   1456.911
## 4  ARMA(2,1)   1457.762
## 9  ARMA(3,3)   1458.377
## 5  ARMA(2,2)   1459.092
## 6  ARMA(2,3)   1459.623
## 3  ARMA(1,3)   1482.347
## 2  ARMA(1,2)   1496.486
## 1  ARMA(1,1)   1507.089
```

```r
subset(sorted_AIC, Candidates %in% c('ARMA(1,1)', 'ARMA(1,3)', 'ARMA(2,1)'))
```

```
##   Candidates AIC_values
## 4  ARMA(2,1)   1457.762
## 3  ARMA(1,3)   1482.347
## 1  ARMA(1,1)   1507.089
```

According to AIC, ARIMA$(3,0,1)$ is the best model for these data. However, excluding the non-appropriate models discovered earlier, we see that it is ARIMA$(2,0,1)$ that is the best model for these data with an AIC of 1457.762.

The model found is described by the following equation which includes also a mean, μ:

$$X_t - \widehat{\mu} = \sum_{i=1}^{p} \widehat{\alpha}_i (X_{t-i} - \widehat{\mu}) + Z_t + \sum_{i=1}^{q} \widehat{\beta}_i Z_{t-i},$$

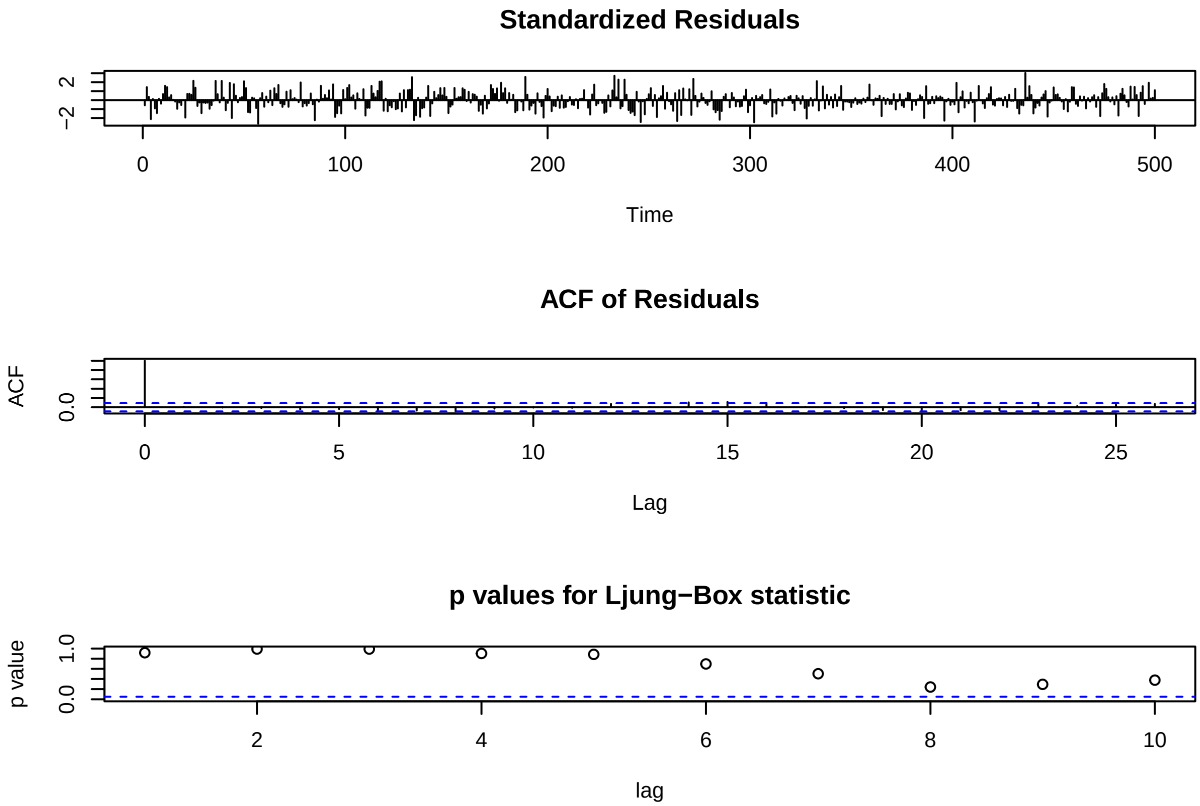
In our case, with $\widehat{\mu} = -0.004$ which has 95% confidence interval $(-0.0321, 0.0240)$,

$$X_t + 0.004 = \widehat{\alpha}_1 (X_{t-1} + 0.004) + \widehat{\alpha}_2 (X_{t-2} + 0.004) + Z_t + \widehat{\beta}_1 Z_{t-1}$$

Estimated Coefficients (with 95% Confidence Intervals)
$\widehat{\alpha}_1 = -0.6437 \quad (-0.7733, -0.5141)$
$\widehat{\alpha}_2 = -0.4657 \quad (-0.5720, -0.3594)$
$\widehat{\beta}_1 = -0.3582 \quad (-0.5038, -0.2125)$

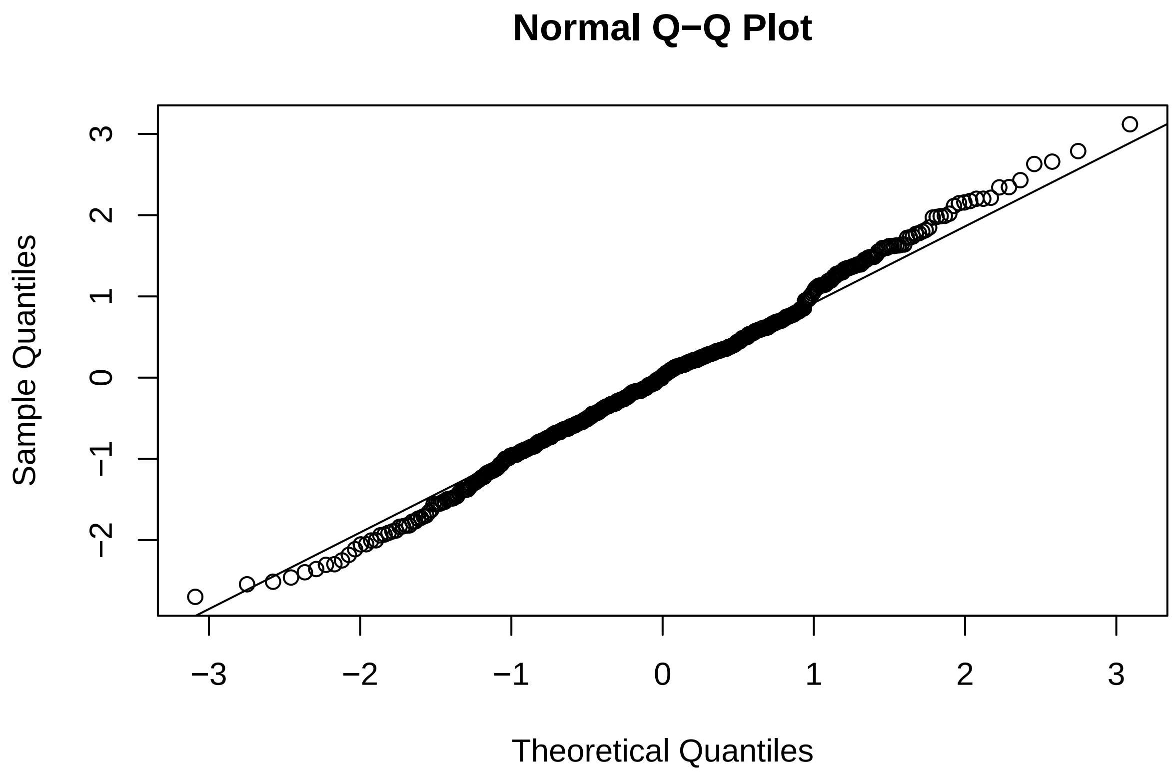
The verification of the goodness of fit of the model chosen is based on the residuals. A plot of the residuals time series and its sample ACF should be compatible with that of white noise, i.e. a Gaussian distribution. A plot of the p-values of the Ljung-Box statistic over a range of lags will also show whether the residuals are compatible with being white noise.

```r
tsdiag(arma21)
```



The ACF of the residuals is truncated at lag 0, indicative of a Gaussian process. The p-values of the Ljung-Box test for ARIMA$(2, 0, 1)$ are clearly compatible with the residuals being white noise.

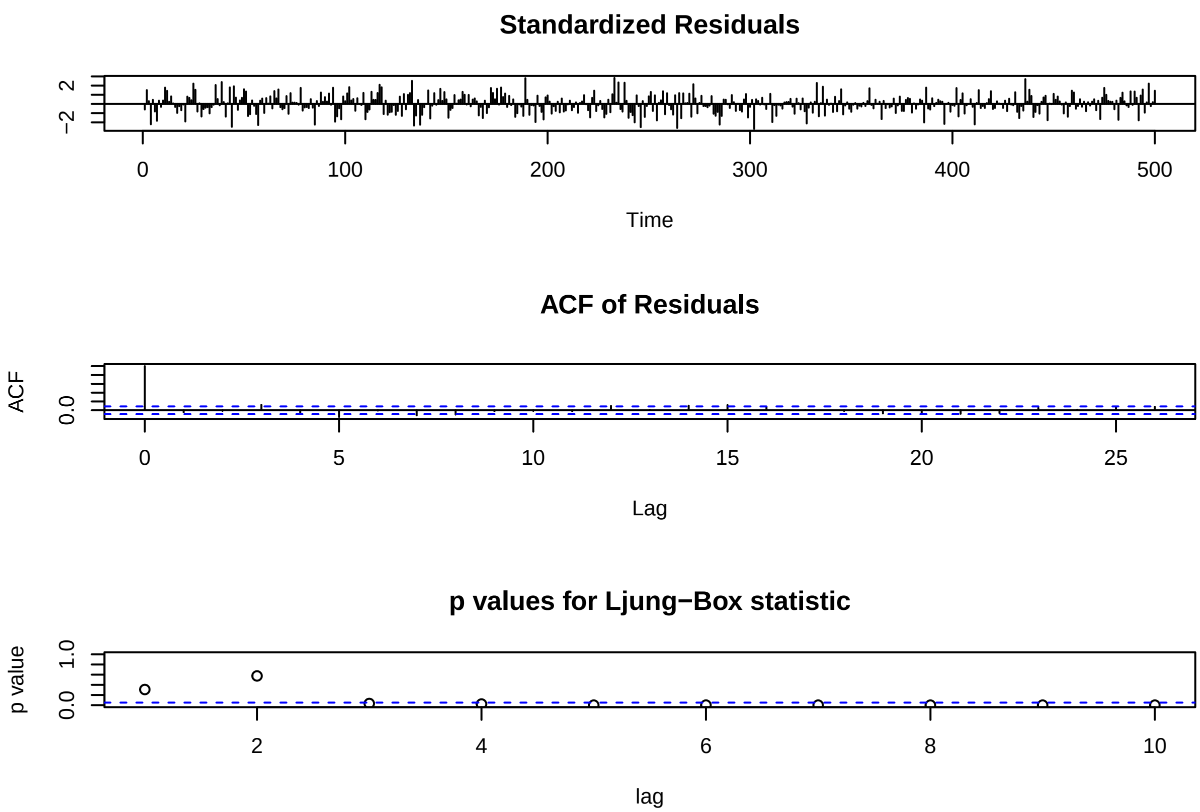
```
residuals21 <- residuals(arma21)
qqnorm(residuals21)
qqline(residuals21)
```



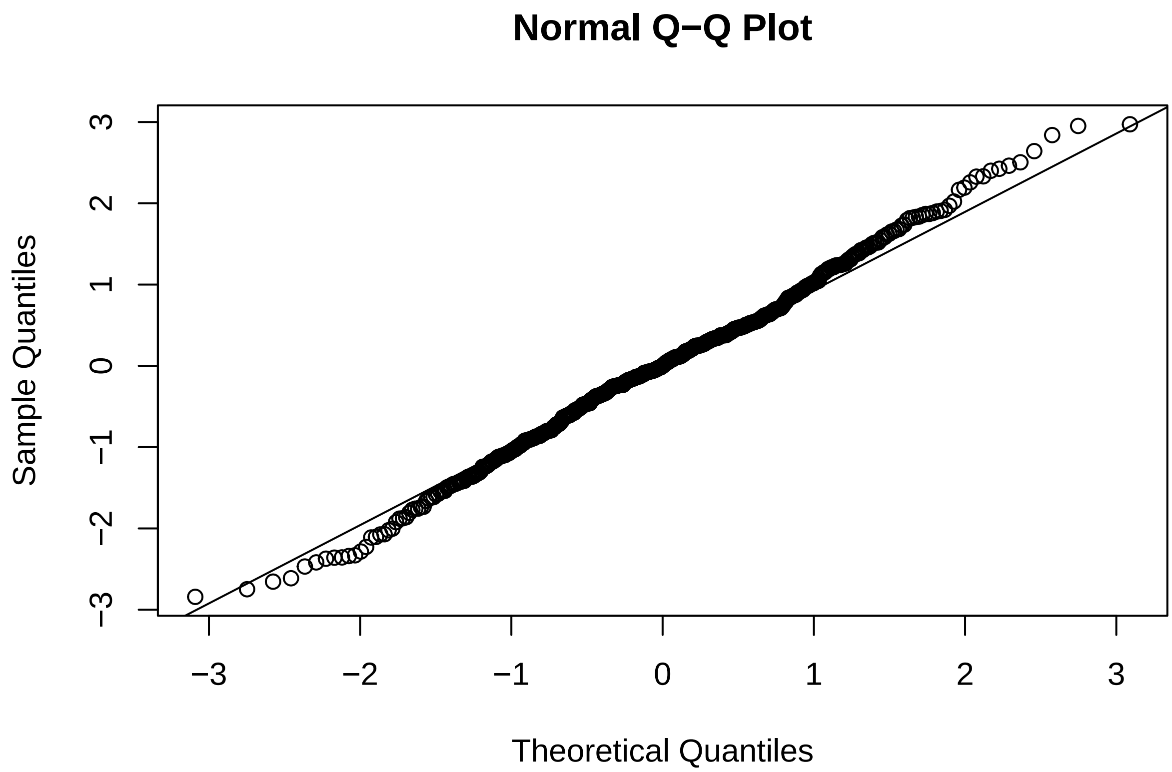
A quantile-quantile plot of the residuals also supports this claim.

Let us try the same with the other models, ARIMA$(1, 0, 3)$ and ARIMA$(1, 0, 1)$.

```
tsdiag(arma13)
```

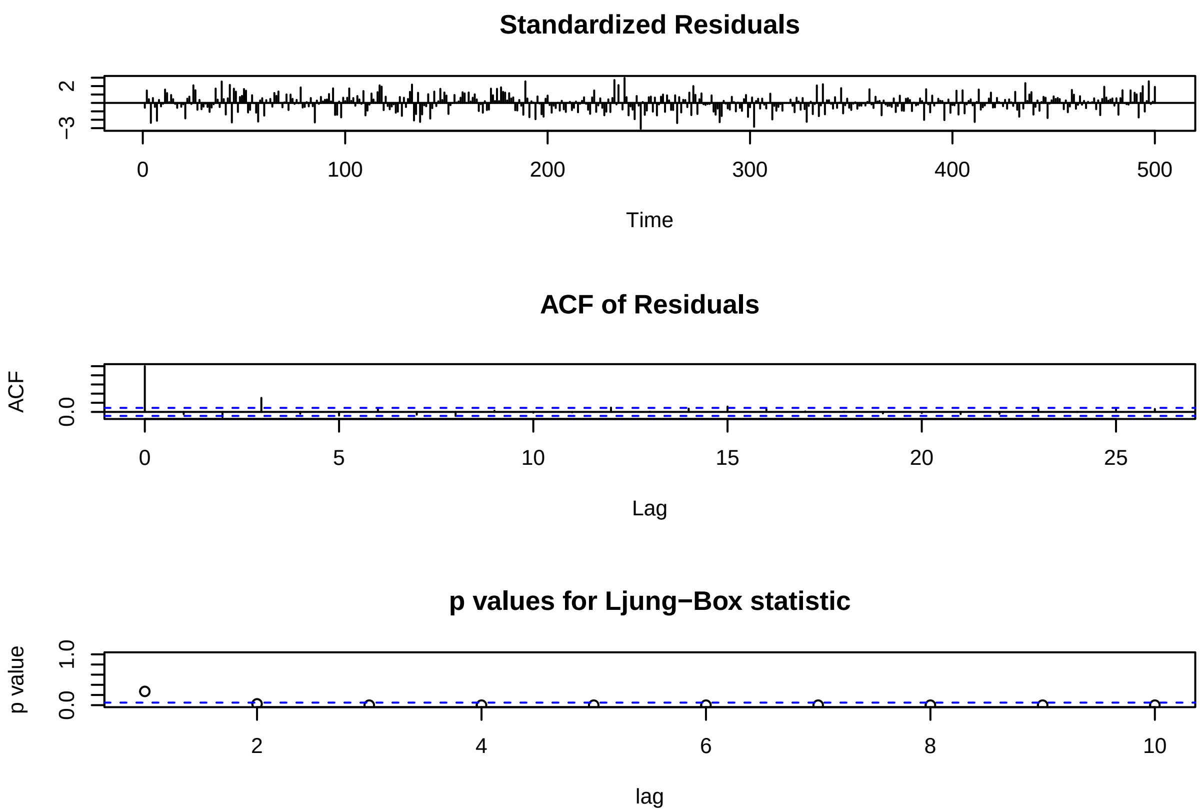



```r
residuals13 <- residuals(arma13)
qqnorm(residuals13)
qqline(residuals13)
```

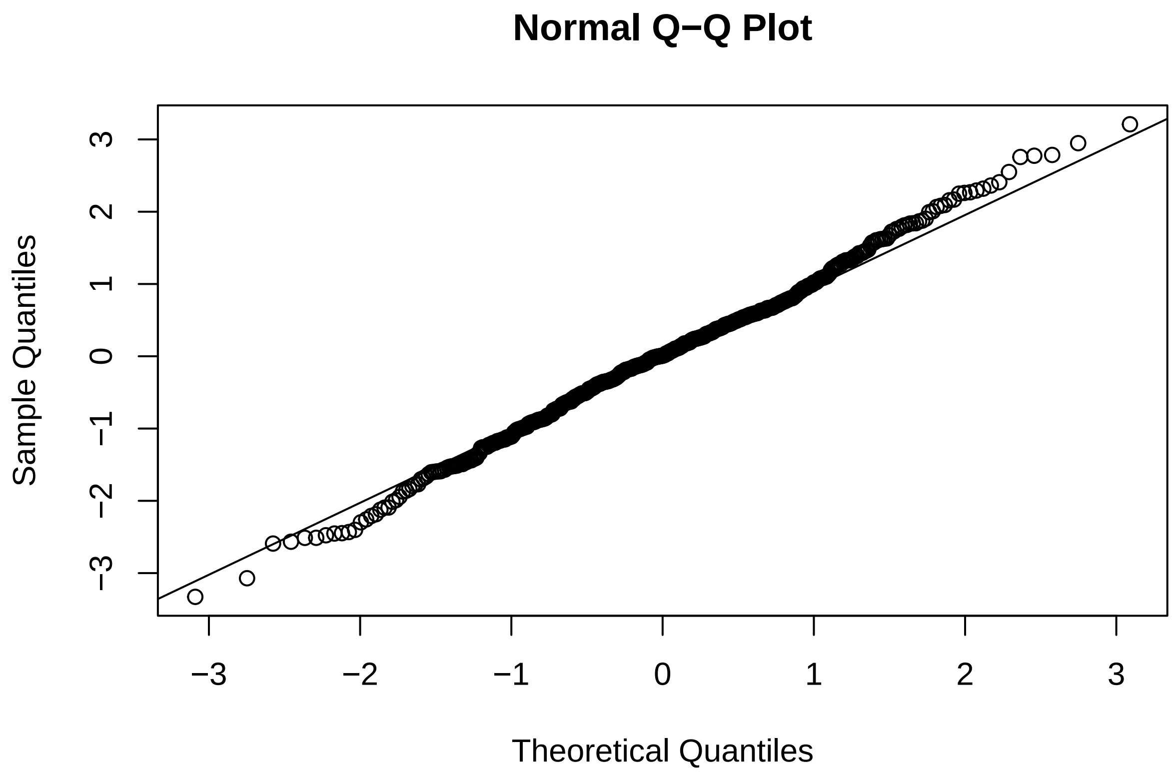


The ACF of the residuals also has a spike at lag 0 but includes significant spikes at other lags, namely 4, 5, 6 etc. and is therefore not indicative of a Gaussian process. The p-values of the Ljung-Box test for ARIMA$(1, 0, 3)$ are clearly not compatible with the residuals being white noise, despite the QQ plot looking somewhat appropriate.

```
tsdiag(arma11)
```




```r
residuals11 <- residuals(arma11)
qqnorm(residuals11)
qqline(residuals11)
```



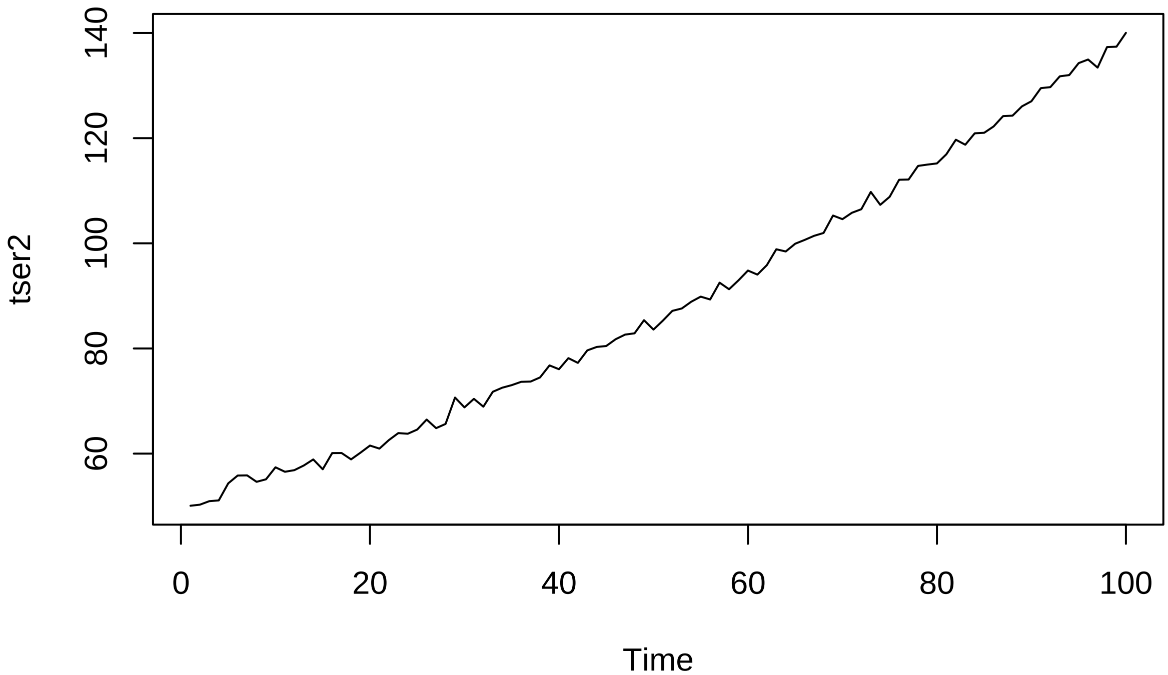
The ACF of the residuals also has a spike at lag 0 but includes significant spikes at other lags, namely 2 & 3, and is therefore not indicative of a Gaussian process.The p-values of the Ljung-Box test for ARIMA$(1, 0, 1)$ are clearly not compatible with the residuals being white noise, despite the QQ plot looking somewhat appropriate. Therefore, we can be satisfied that the ARIMA$(2, 0, 1)$ model describes the data given here in the best way.

Hence our model is given by:

$$X_t + 0.004 = -0.6437(X_{t-1} + 0.004) - 0.4657(X_{t-2} + 0.004) + Z_t - 0.3582Z_{t-1}.$$

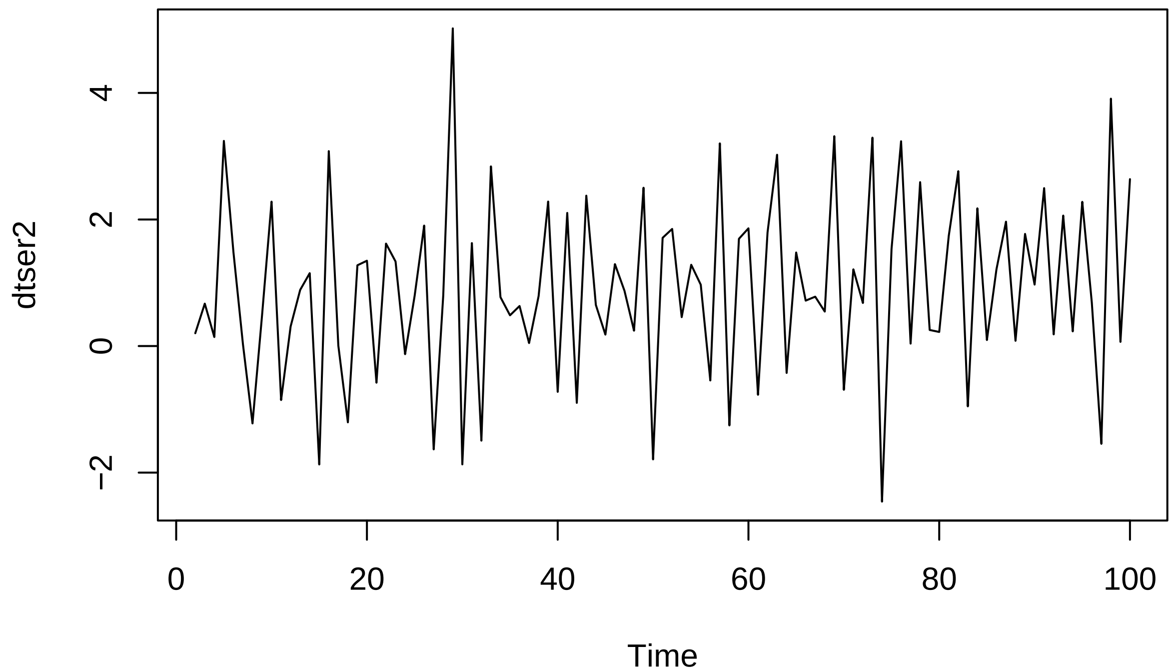
2

```
plot(tser2)
```



We know that 'tser2' is a non-stationary series with a deterministic polynomial trend. To determine the degree of the polynomial trend we difference the series until we have a stationary process. If the differenced series is stationary the polynomial will be of degree 1. If the once differenced series is not stationary but the twice differenced series is then the polynomial trend is of degree 2.

```
dtser2 <- diff(tser2)
plot(dtser2)
```



Visually the differenced series looks stationary. We will conduct KS and ADF tests to verify this.

```
x2 <- dtser2[1:50]
y2 <- dtser2[51:99]
ks.test(x2, y2)
```

```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  x2 and y2
## D = 0.18816, p-value = 0.3043
## alternative hypothesis: two-sided
```

A high p-value means we accept the null and thus X_t is a stationary series according to the KS test.

```
adf.test(dtser2)
```

```
## Warning in adf.test(dtser2): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  dtser2
## Dickey-Fuller = -8.1182, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

```
adf.test(dtser2, k = 10)
```

```
##
```

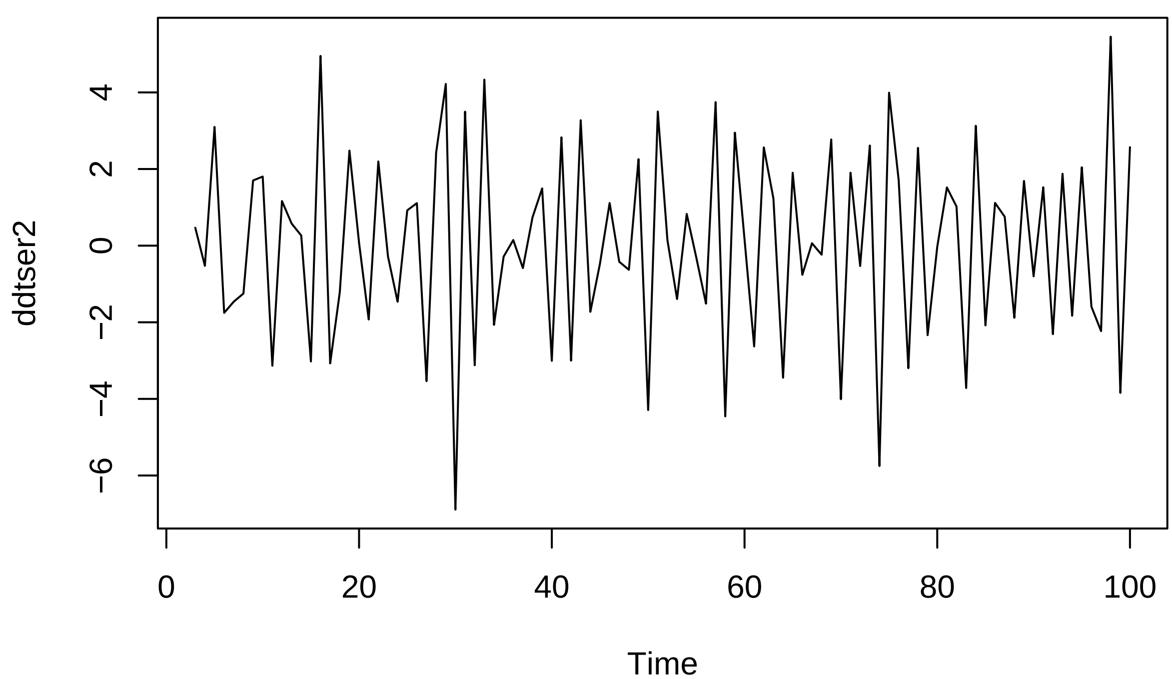
```
##  Augmented Dickey-Fuller Test
##
## data:  dtser2
## Dickey-Fuller = -3.2362, Lag order = 10, p-value = 0.08587
## alternative hypothesis: stationary
```

```
adf.test(dtser2, k = 15)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  dtser2
## Dickey-Fuller = -2.9957, Lag order = 15, p-value = 0.1644
## alternative hypothesis: stationary
```

The ADF test returns a small p-value but only for lag order 4 and not for higher lag orders indicating non-stationarity over higher orders. This could also potentially be caused by the small sample size of 100 for this time series. We will difference the series again and conduct the same tests to determine whether or not the twice differenced series is stationary and therefore that the polynomial trend is of degree 2.

```
ddtser2 <- diff(dtser2)
plot(ddtser2)
```



Visually this twice-differncec series looks stationary and the KS and ADF tests support this.

```
x3 <- ddtser2[1:49]
y3 <- ddtser2[50:98]
ks.test(x3, y3)
```

```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  x3 and y3
## D = 0.081633, p-value = 0.9973
## alternative hypothesis: two-sided
```

A high p-value means we accept the null and thus X_t is a stationary series according to the KS test.

```
adf.test(ddtser2)
```

```
## Warning in adf.test(ddtser2): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ddtser2
## Dickey-Fuller = -10.932, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

```
adf.test(ddtser2, k = 10)
```

```
## Warning in adf.test(ddtser2, k = 10): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ddtser2
## Dickey-Fuller = -5.4694, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
```

```
adf.test(ddtser2, k = 15)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ddtser2
## Dickey-Fuller = -3.8201, Lag order = 15, p-value = 0.02088
## alternative hypothesis: stationary
```

The test, consistently, returns small p-values confirming, once again, that this series is stationary.The equations below shows what differencing has done to our model, where ∇ is the difference operator.

$$Y_t = X_t + at^2 + bt + c\nabla Y_t = \nabla X_t + 2at - a + b\nabla^2 Y_t = \nabla^2 X_t + 2a$$

To determine the coefficients a, b, c we run a linear regression of the time series against varying degrees of time up to degree 3 to verfiy our claim that the polynomial trend is of degree 2. We should expect significant coefficients up to degree 2 by the information discovered above.

```
time_index <- seq_along(tser2) #values of t
lm <- lm(tser2 ~ time_index) #linear regression
summary(lm)
```

```
##
## Call:
## lm(formula = tser2 ~ time_index)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -4.8480 -2.6284 -0.7009  2.5365  6.9903
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 43.44349    0.63702   68.20   <2e-16 ***
## time_index   0.89983    0.01095   82.17   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.161 on 98 degrees of freedom
## Multiple R-squared:  0.9857, Adjusted R-squared:  0.9855
## F-statistic:  6751 on 1 and 98 DF,  p-value: < 2.2e-16
```

```r
qm <- lm(tser2 ~ time_index + I(time_index^2)) #quadratic regression
summary(qm)
```

```
##
## Call:
## lm(formula = tser2 ~ time_index + I(time_index^2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.6217 -0.5862  0.0185  0.5761  2.6055
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     5.030e+01  3.024e-01  166.34   <2e-16 ***
## time_index      4.967e-01  1.382e-02   35.94   <2e-16 ***
## I(time_index^2) 3.992e-03  1.326e-04   30.11   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9878 on 97 degrees of freedom
## Multiple R-squared:  0.9986, Adjusted R-squared:  0.9986
## F-statistic: 3.502e+04 on 2 and 97 DF,  p-value: < 2.2e-16
```

```r
cm <- lm(tser2 ~ time_index + I(time_index^2) + I(time_index^3)) #cubic regression
summary(cm)
```

```
##
## Call:
## lm(formula = tser2 ~ time_index + I(time_index^2) + I(time_index^3))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.41905 -0.60304  0.00412  0.65817  2.75222
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      5.065e+01  4.092e-01 123.771  < 2e-16 ***
## time_index       4.562e-01  3.491e-02  13.068  < 2e-16 ***
## I(time_index^2)  4.988e-03  8.011e-04   6.227 1.25e-08 ***
## I(time_index^3) -6.576e-06  5.215e-06  -1.261     0.21
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

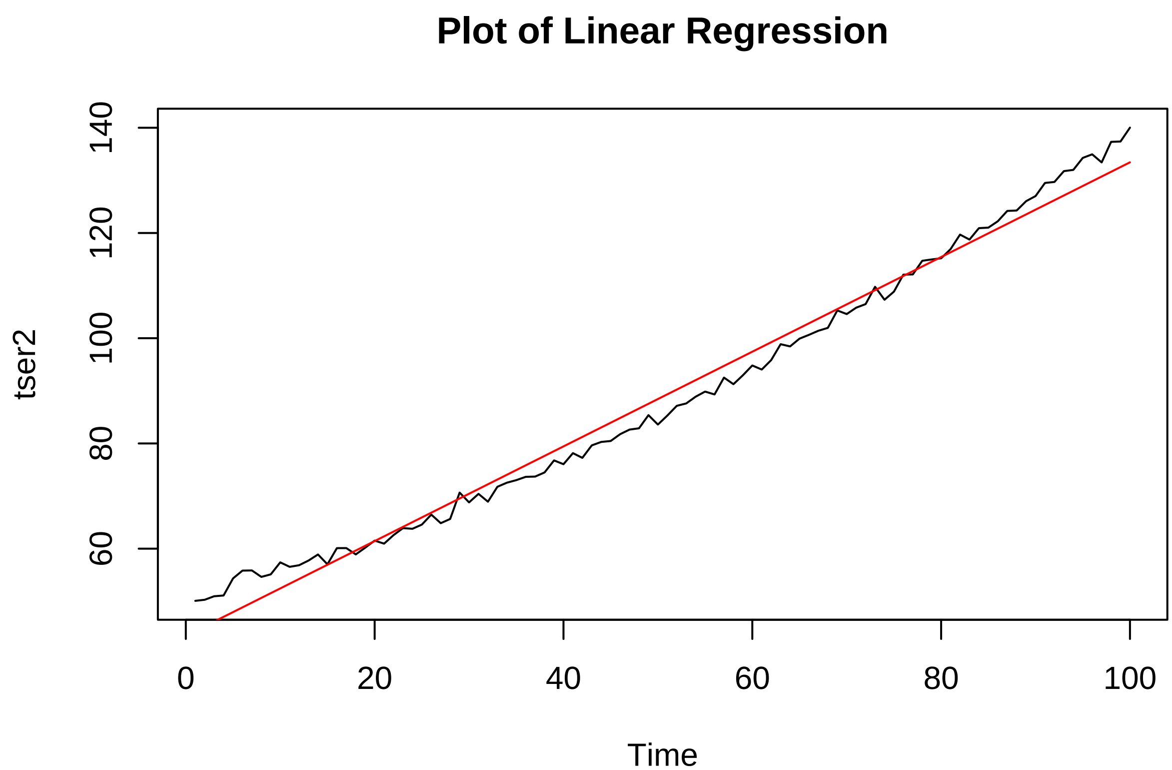
```
##
## Residual standard error: 0.9849 on 96 degrees of freedom
## Multiple R-squared:  0.9986, Adjusted R-squared:  0.9986
## F-statistic: 2.349e+04 on 3 and 96 DF,  p-value: < 2.2e-16
```

All coefficients are significant except the degree 3 time index coefficient of the cubic model, confirming that our trend is of degree 2. The coefficient a for the quadratic term of our quadratic regression appears significant despite being very small which we will look into. The plots below help visualise the estimated regressions.

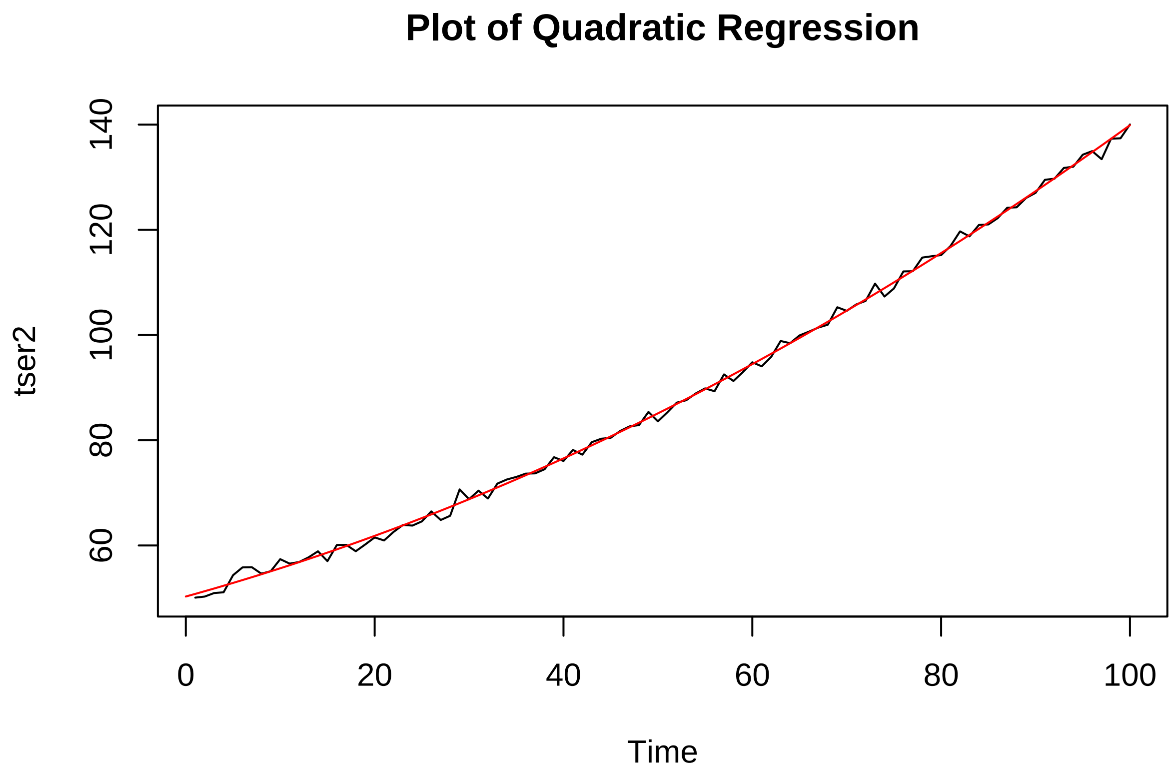
```
x <- seq(0,100, by = 1)
2*qm$coefficients[3]
```

```
## I(time_index^2)
##     0.007983174
```

```
y = lm$coefficients[2] * x + lm$coefficients[1]
plot(tser2, main = 'Plot of Linear Regression')
lines(x,y, col='red')
```

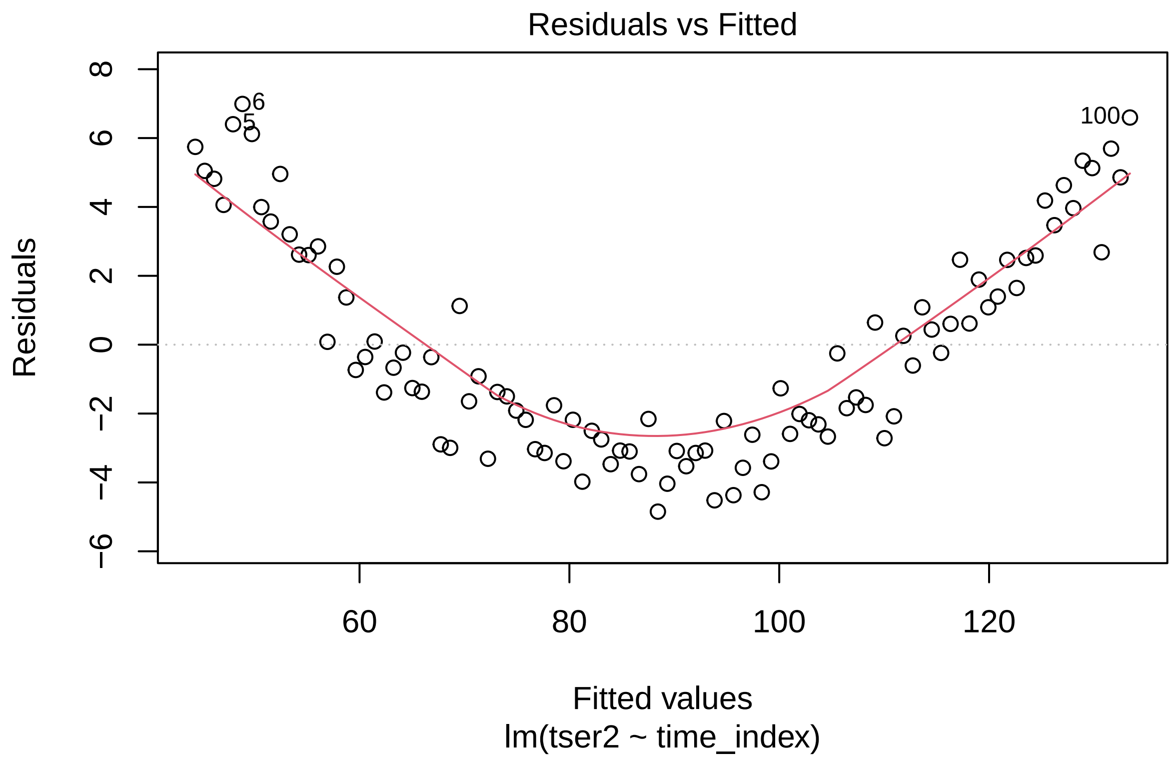


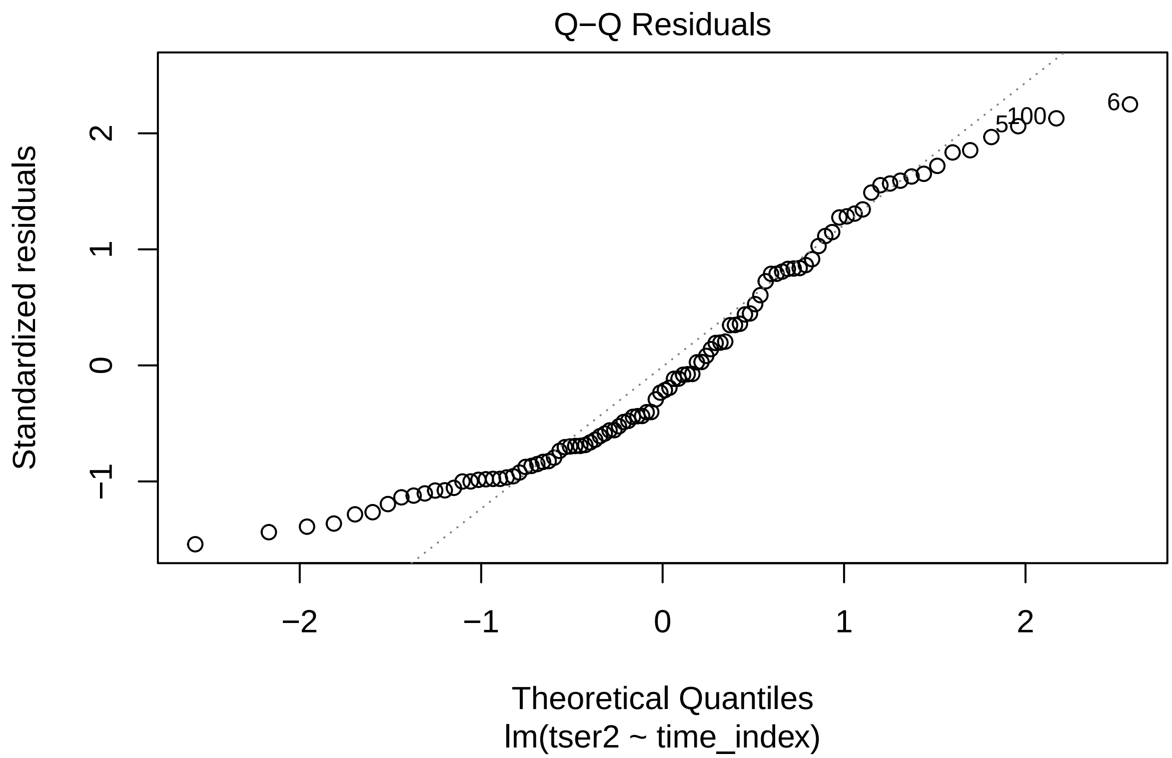

```
y_quad = qm$coefficients[3] * x^2 + qm$coefficients[2] * x + qm$coefficients[1]
plot(tser2, main = 'Plot of Quadratic Regression')
lines(x, y_quad, col='red')
```



Both regressions appear appropriate, with a stronger visual relationship with the quadratic model. A look at the diagnostic plots for each regression will provide more information about the normality of the regression residuals.

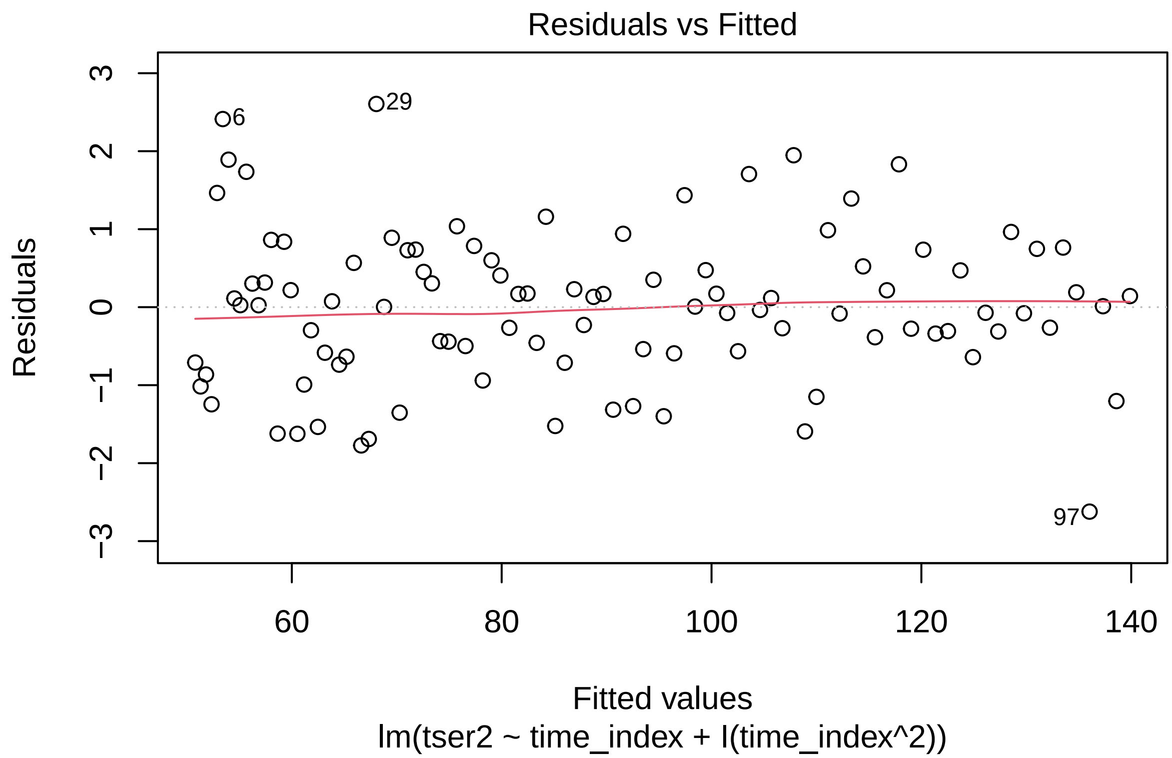
```
plot(lm,1:2)
```


Residuals vs Fitted
Residuals
Fitted values
lm(tser2 ~ time_index)
8
6
4
2
0
−2
−4
−6
60
80
100
120
6
5
100

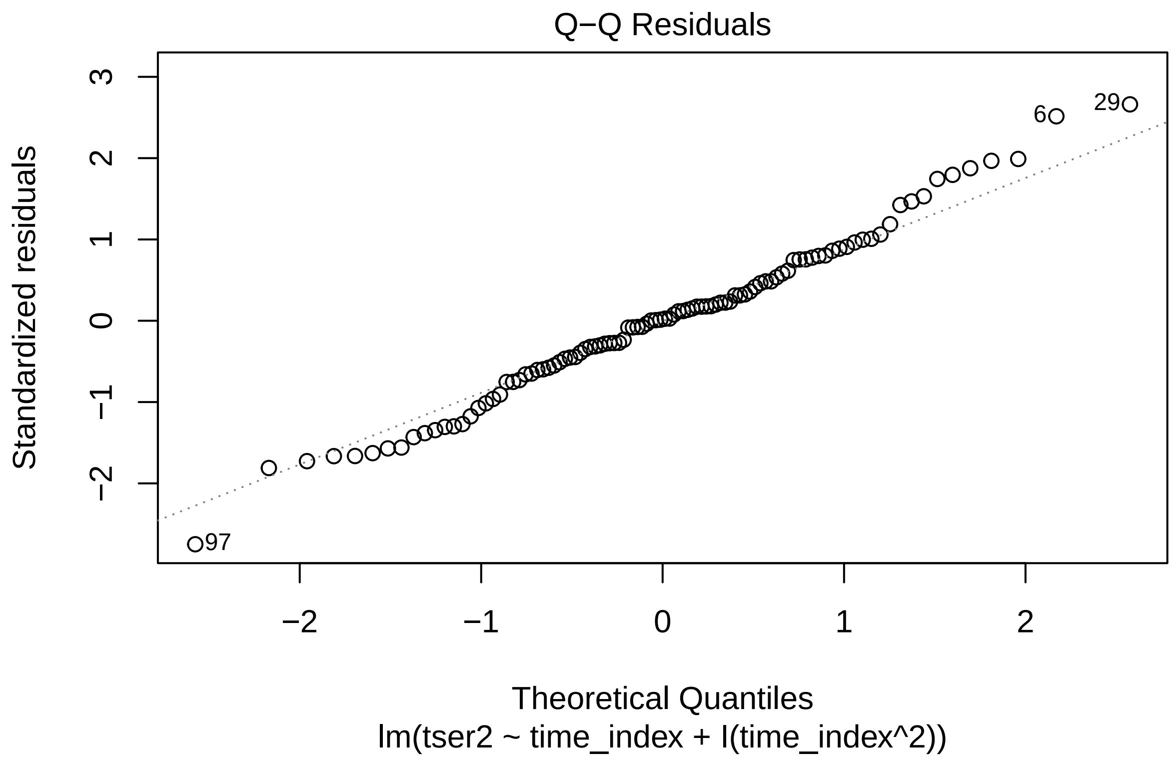


The diagnostic plots for the residuals of the linear regression are not distributed normally and exhibit a non-linear residual. This invalidates the assumption of normality.

```r
plot(qm,1:2)
```



Residuals vs Fitted
Residuals
Fitted values
lm(tser2 ~ time_index + I(time_index^2))
3
2
1
0
−1
−2
−3
60
80
100
120
140
6
29
97



The diagnostic plots for the residuals of the quadratic regression show much clearer signs of normality. Therefore we can conclude that the trend polynomial is quadratic in nature.

Our model is therefore:

$$Y_t = X_t + \widehat{a}t^2 + \widehat{b}t + \widehat{c},$$

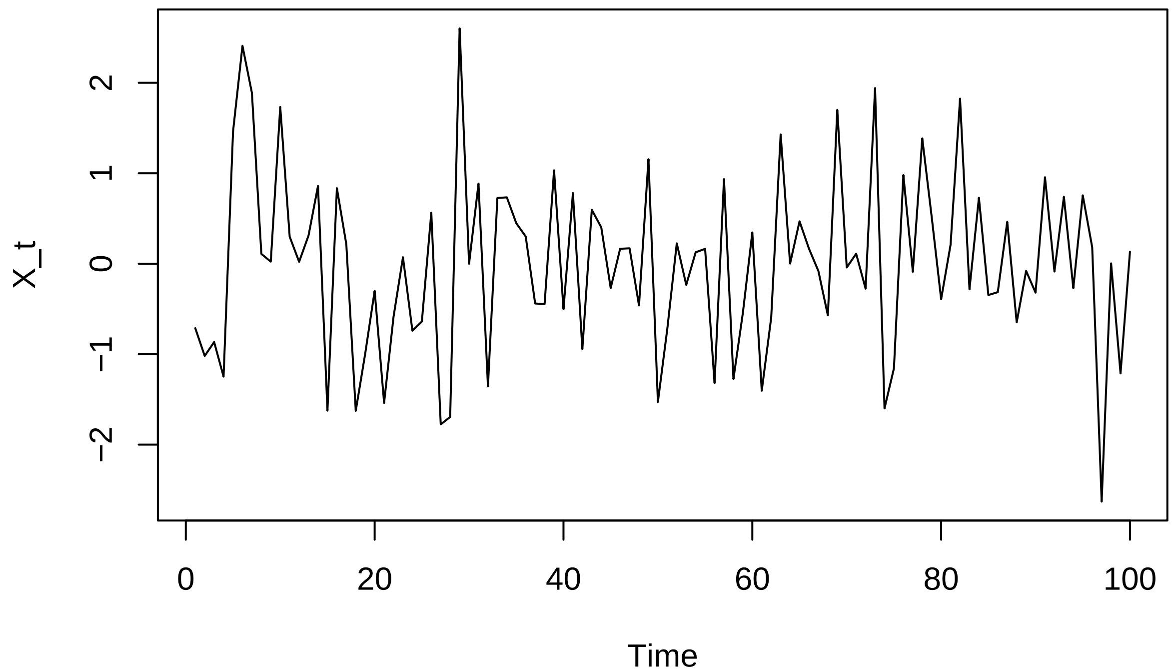
Estimated Coefficients (with 95% Confidence Intervals)
$\widehat{a} = 0.0040 \quad (0.0037, 0.0043)$
$\widehat{b} = 0.4967 \quad (0.4940, 0.4994)$
$\widehat{c} = 50.30 \quad (44.37, 56.23)$

We can now rearrange Y_t to run analysis on X_t:

$$Y_t = X_t + \widehat{a}t^2 + \widehat{b}t + \widehat{c}X_t = Y_t - \widehat{a}t^2 - \widehat{b}t - \widehat{c},$$

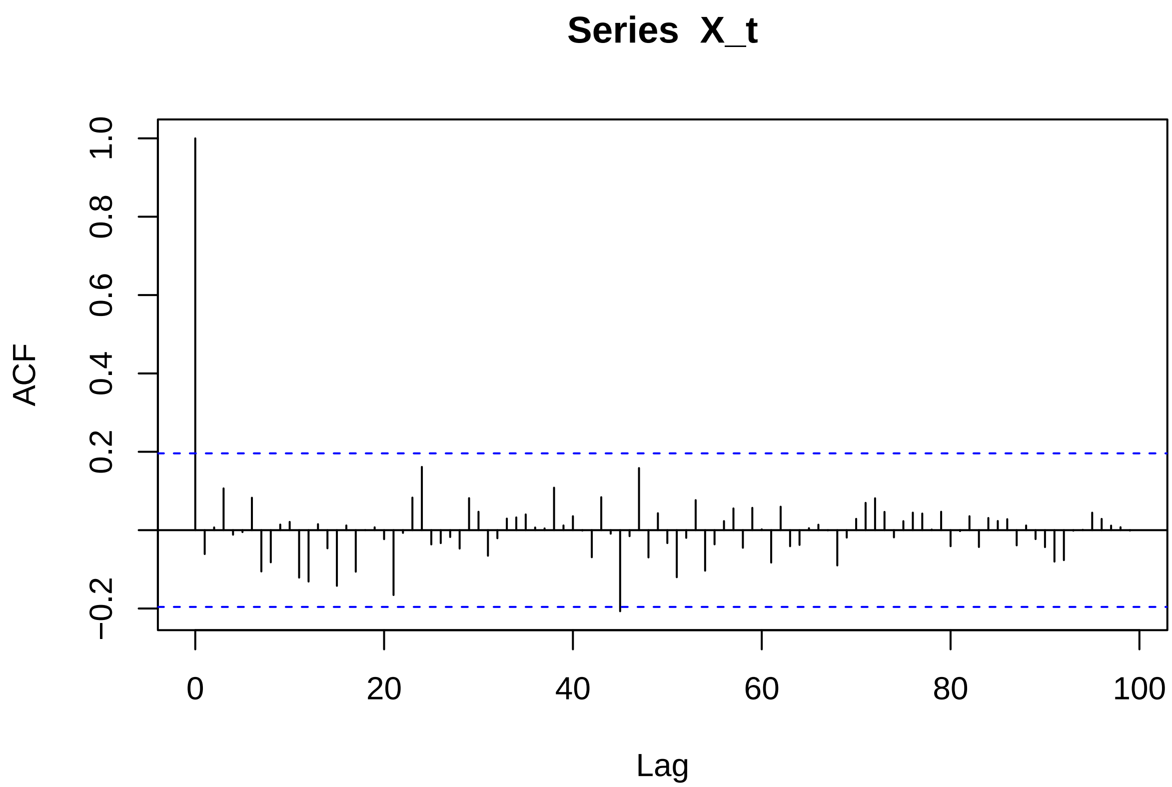
```
m_t <- function(t) 3.992e-3 * t^2 + 4.967e-1 * t + 50.3

X_t <- tser2 - m_t(1:length(tser2))
plot(X_t)
```

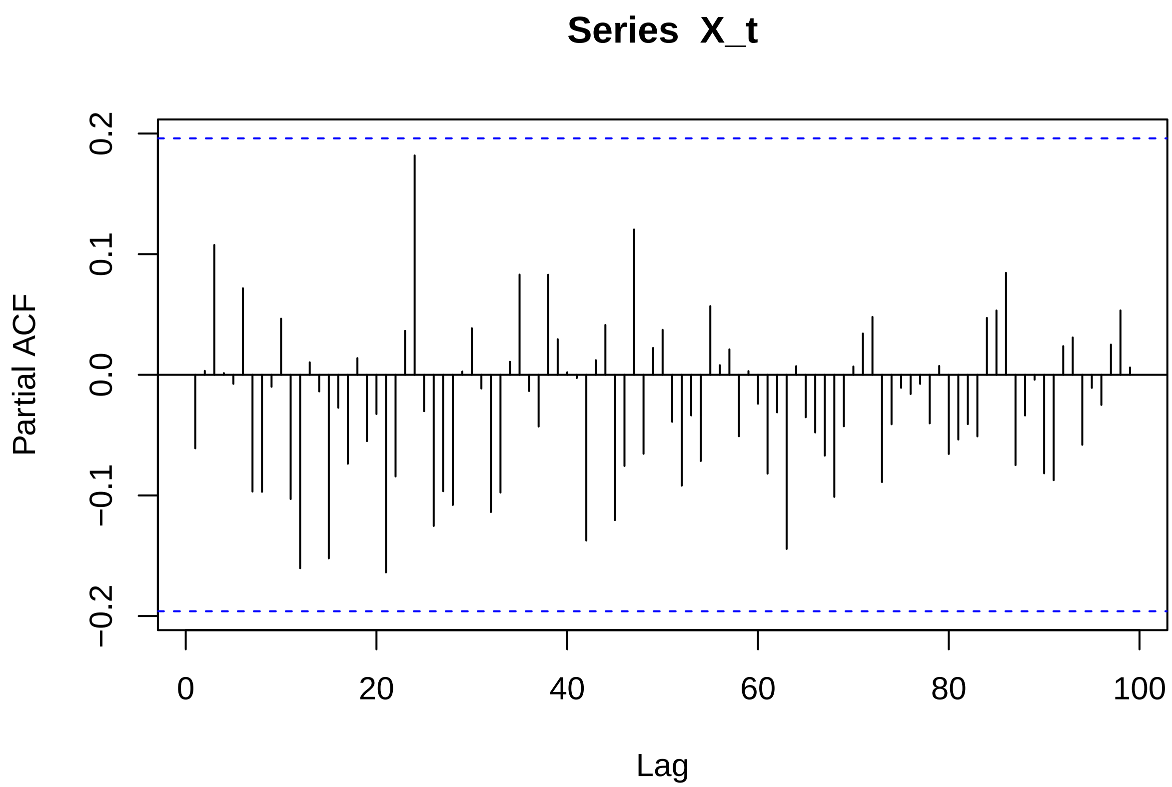


The visual impression of X_t is of a stationary series. We plot the sample ACF. A stationary series will have a fast (exponential/sinusoidal) decay or a truncation, while a slow decay is typical of a non-stationary series.

```
acf(X_t, lag.max = 100)
```




```
pacf(X_t, lag.max = 100)
```



The sample ACF plot exhibits a truncation at lag 0, indicative of a stationary series, namely a Gaussian process. A Gaussian process is inherently stationary, conducting a Kolmogorov-Smirnov test should support this claim. The null and alternative hypotheses are as follows:

H_0: the time series is stationary H_1: the time series is non-stationary

```
x4 <- X_t[1:50]
y4 <- X_t[51:100]

ks.test(x4,y4)
```

```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  x4 and y4
## D = 0.16, p-value = 0.5487
## alternative hypothesis: two-sided
```

A high p-value means we accept the null and thus X_t is a stationary series.

Applying the ADF test to a Gaussian process may not be meaningful, as white noise is already stationary and lacks the characteristics that the ADF test is designed to detect i.e a unit root.

```
adf.test(X_t)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  X_t
```

```
## Dickey-Fuller = -3.8554, Lag order = 4, p-value = 0.01902
## alternative hypothesis: stationary
```

```
adf.test(X_t,k=10)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  X_t
## Dickey-Fuller = -3.2483, Lag order = 10, p-value = 0.08373
## alternative hypothesis: stationary
```

```
adf.test(X_t,k=15)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  X_t
## Dickey-Fuller = -3.3454, Lag order = 15, p-value = 0.06757
## alternative hypothesis: stationary
```

The test, returns a small p-value but only for lag order 4 and not for higher lag orders leading to inconclusive results as expected. This could also potentially be caused by the small sample size of 100 for this time series.

Assuming X_t is a Gaussian process, the appropriate model should be an ARIMA$(0, 0, 0)$ process. We will compare with other similar process to verify.

```
ModelA <- arima(X_t, order = c(0, 0, 0))
print(ModelA)
```

```
##
## Call:
## arima(x = X_t, order = c(0, 0, 0))
##
## Coefficients:
##       intercept
##         -0.0052
## s.e.     0.0973
##
## sigma^2 estimated as 0.9466:  log likelihood = -139.15,  aic = 282.3
```

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

```
##  intercept
## -0.1998086
```

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

```
## intercept
## 0.1893594
```

```
ModelB <- arima(X_t, order = c( 0, 0, 1))
print(ModelB)
```

```
##
## Call:
## arima(x = X_t, order = c(0, 0, 1))
##
## Coefficients:
```

```
##           ma1  intercept
##       -0.0591    -0.0048
## s.e.   0.0973     0.0914
##
## sigma^2 estimated as 0.9431:  log likelihood = -138.97,  aic = 283.93
```

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

```
##        ma1  intercept
## -0.2536493 -0.1876858
```

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

```
##       ma1 intercept
## 0.1355082 0.1780683
```

All confidence intervals of the ARMA(0,1) model do not exclude 0, thus not all parameters are significant and the model is deemed not appropriate.

```
ModelC <- arima(X_t, order = c( 1, 0, 0))
print(ModelC)
```

```
##
## Call:
## arima(x = X_t, order = c(1, 0, 0))
##
## Coefficients:
##           ar1  intercept
##       -0.0607    -0.0049
## s.e.   0.0996     0.0916
##
## sigma^2 estimated as 0.943:  log likelihood = -138.96,  aic = 283.93
```

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

```
##        ar1  intercept
## -0.2598851 -0.1881121
```

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

```
##       ar1 intercept
## 0.1384875 0.1783148
```

All confidence intervals of the ARMA(1,0) model do not exclude 0, thus not all parameters are significant and the model is deemed not appropriate.

```
ModelD <- arima(X_t, order = c( 1, 0, 1))
print(ModelD)
```

```
##
## Call:
## arima(x = X_t, order = c(1, 0, 1))
##
## Coefficients:
##           ar1     ma1  intercept
##       -0.0714  0.0107    -0.0049
## s.e.   0.7778  0.7751     0.0917
##
## sigma^2 estimated as 0.943:  log likelihood = -138.96,  aic = 285.93
```

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

```
##        ar1        ma1  intercept
## -1.6269941 -1.5394373 -0.1882585
```

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

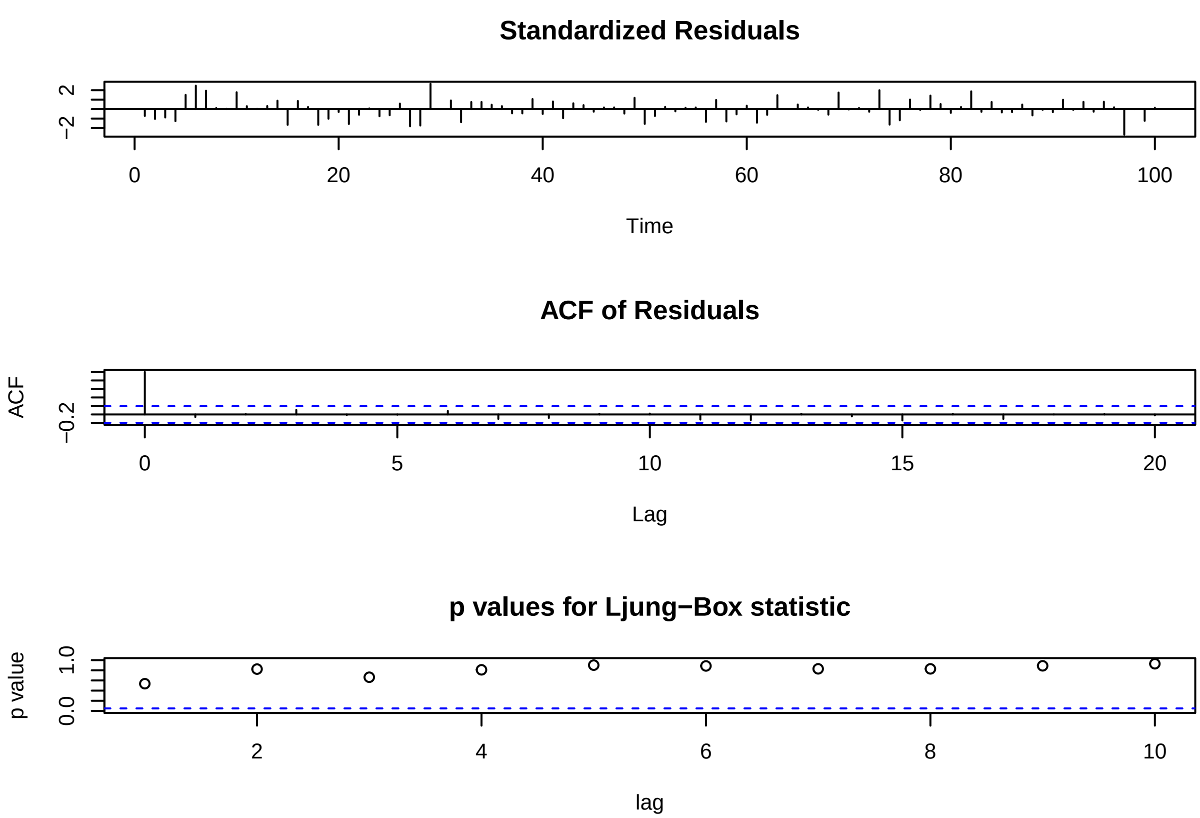
```
##       ar1       ma1 intercept
## 1.4842922 1.5607999 0.1784267
```

All confidence intervals of the ARMA(1,1) model do not exclude 0, thus not all parameters are significant and the model is deemed not appropriate.

The estimates of the data variance for the only appropriate model, which additionally had the lowest AIC at 282.3, ARIMA$(0,0,0)$ is close to its correct value 1, at 0.9466. As none of the other models are appropriate we can conclude that X_t is an ARIMA$(0,0,0)$ process i.e. white noise.

A plot of the residuals time series and its sample ACF should be compatible with that of white noise, i.e. a Gaussian distribution. A plot of the p-values of the Ljung-Box statistic over a range of lags will also show whether the residuals are compatible with being white noise.

```r
tsdiag(ModelA)
```



The ACF of Residuals and p-values of the Ljung-Box test for ARIMA$(0,0,0)$ are clearly compatible with the residuals being white noise, verifying our claim.

Our model is therefore:

$$Y_t = Z_t + 0.0040t^2 + 0.4967t + 50.30.$$