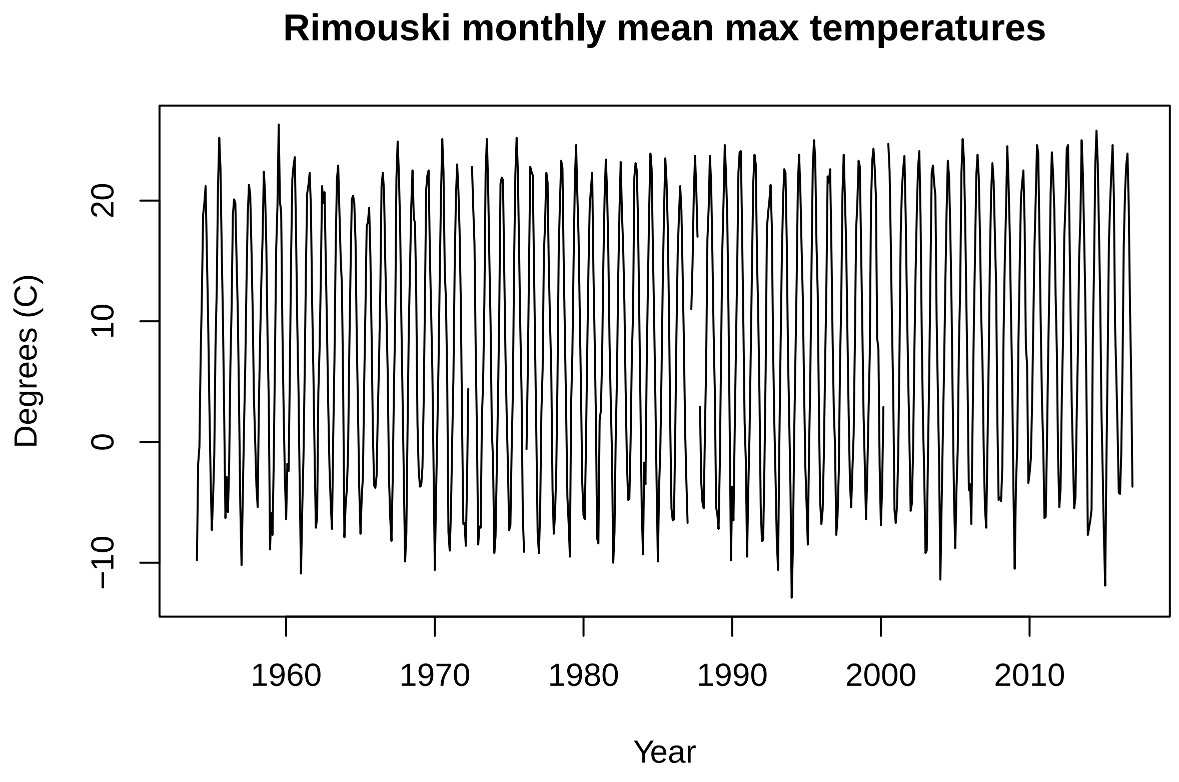
Assignment 1

Derek Situ (62222807)

Question 1

(1a)

```r
# read data and plot series
climate <- read.csv("rimouski_climate_1954_2016.csv", header = TRUE)
mean_max_temp <- ts(climate$Mean.Max.Temp..Â.C., start = c(1954, 1),
                    frequency = 12)
plot(mean_max_temp, main = "Rimouski monthly mean max temperatures",
     xlab = "Year", ylab = "Degrees (C)")
```



The series does not appear to have a clear trend from this plot. There is seasonal variation with a period of 1 year or 12 observations. Because of this seasonality, the expected value of the observations is not constant, so the series is not stationary. An additive model is more appropriate than a multiplicative model to decompose this series since the seasonal effect does not seem to scale with the temperature itself, so it makes more sense to add the seasonal effect rather than multiply it with the trend component.

(1b)

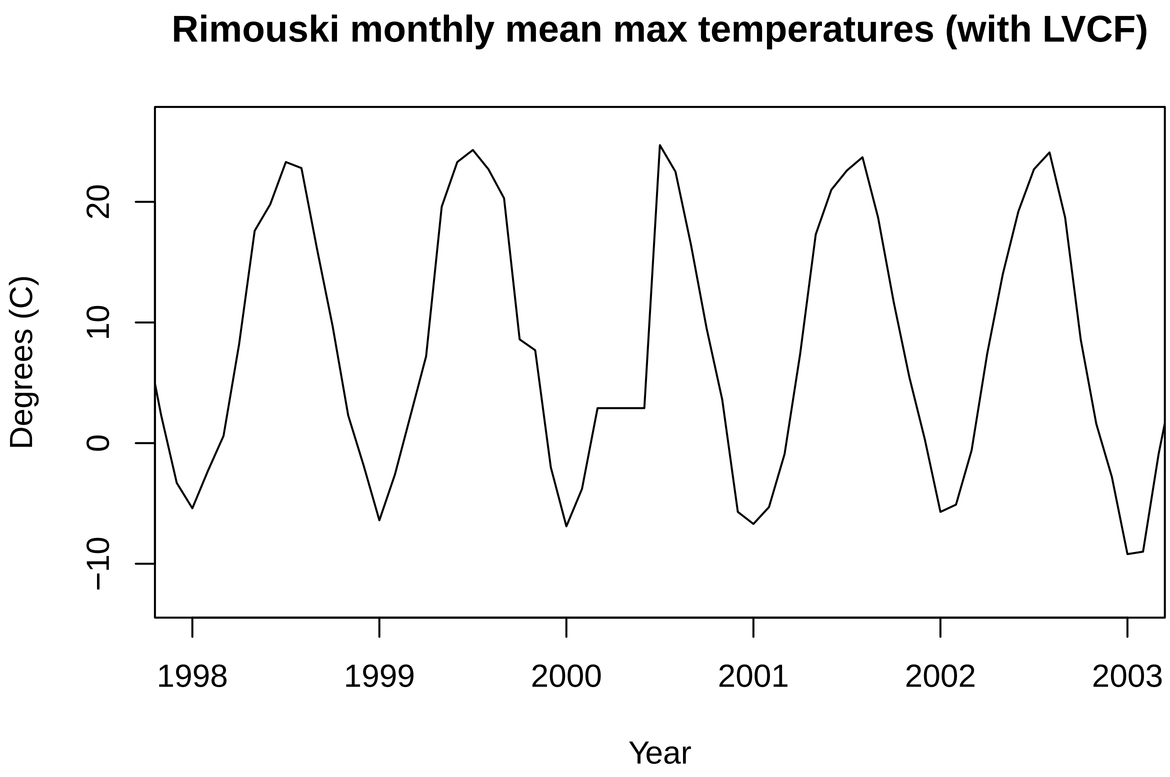
```r
# identify missing values
for (i in 1:length(climate$Mean.Max.Temp..Â.C.)) {
  if (is.na(climate$Mean.Max.Temp..Â.C.[i])) {
    cat("Missing value at index", i, ":", climate$Date.Time[i], "\n")
  }
}
```

```
## Missing value at index 221 : 1972-05
## Missing value at index 222 : 1972-06
## Missing value at index 266 : 1976-02
## Missing value at index 398 : 1987-02
## Missing value at index 399 : 1987-03
## Missing value at index 406 : 1987-10
## Missing value at index 556 : 2000-04
## Missing value at index 557 : 2000-05
## Missing value at index 558 : 2000-06
```

The LVCF imputation method is not appropriate here because it does not capture the different seasonal effect from one month to the next. And since there are consecutive missing values, using LVCF will create big gaps where the appropriate seasonal effect is not captured. The following code imputes with LVCF and creates a plot to demonstrate the weird result it generates.

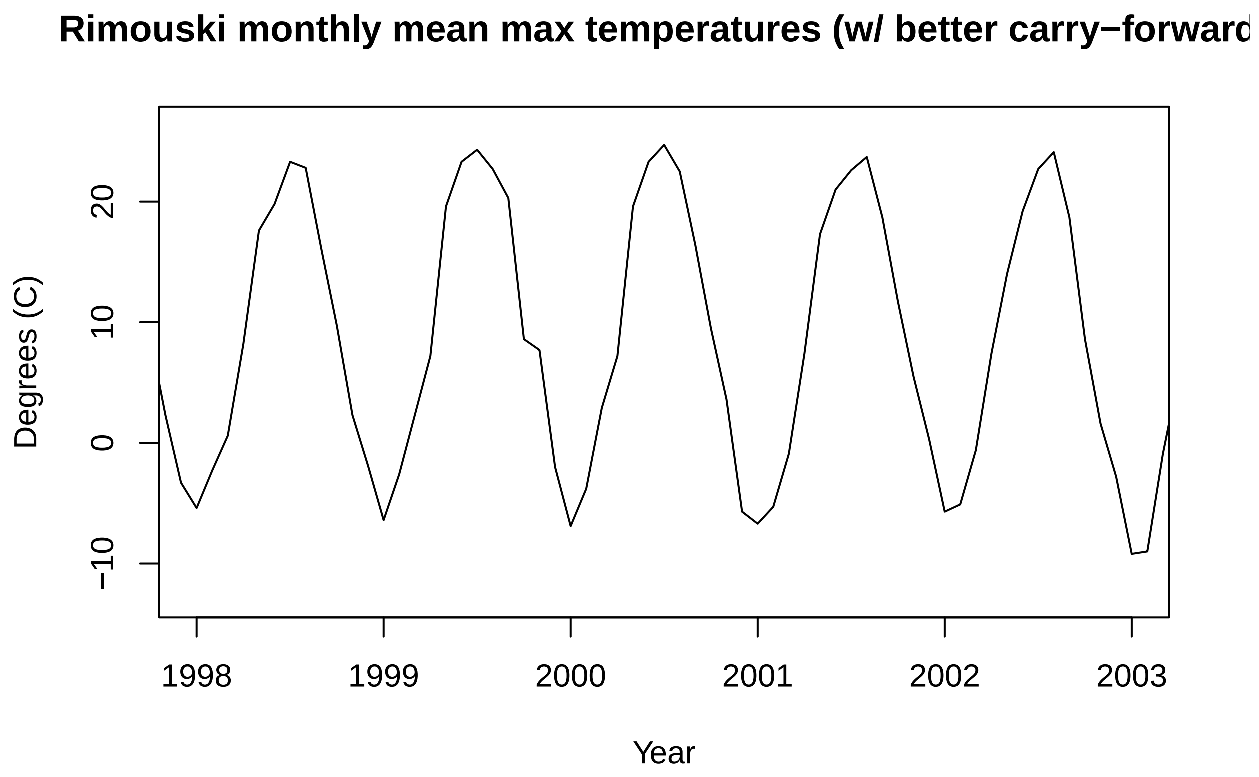
```r
# plot with LVCF imputation
mean_max_temp_lvcf <- na.locf(climate$Mean.Max.Temp..Â.C.) |>
  ts(start = c(1954, 1), frequency = 12)

plot(mean_max_temp_lvcf,
     main = "Rimouski monthly mean max temperatures (with LVCF)",
     xlab = "Year", ylab = "Degrees (C)", xlim = c(1998, 2003))
```



Notice that in 2000, where there are 3 consecutive imputed values, the plot looks very unusual compared to the observed data. Something that might be better could be to carry forward the value from the same month last year. Let's see what the plot looks like with this adapted carry-forward method.

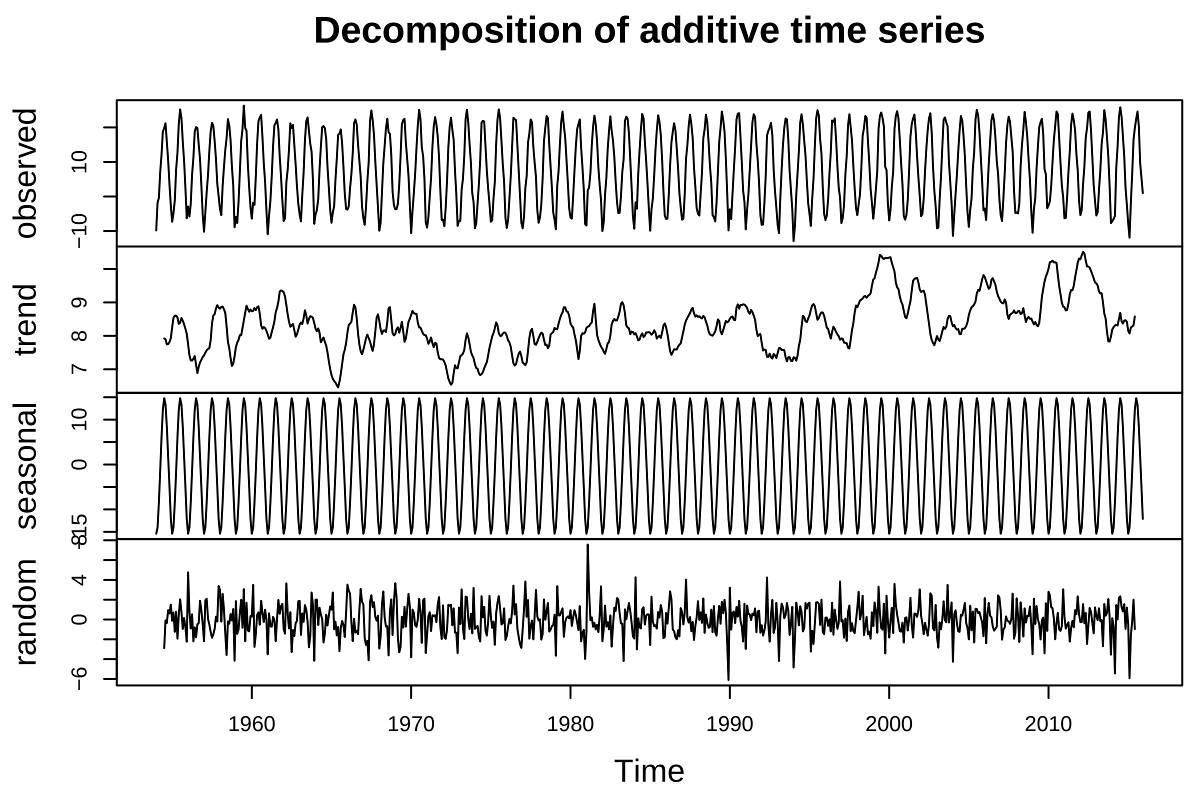
```r
# plot with a better carry-forward method
mean_max_temp_bcf <- climate$Mean.Max.Temp..Â.C.
for (i in 1:length(mean_max_temp_bcf)) {
  if (is.na(mean_max_temp_bcf[i])) {
    mean_max_temp_bcf[i] = mean_max_temp_bcf[i - 12]
  }
}
mean_max_temp_bcf_ts <- ts(mean_max_temp_bcf, start = c(1954, 1),
                            frequency = 12)
plot(mean_max_temp_bcf_ts,
     main = "Rimouski monthly mean max temperatures (w/ better carry-forward)",
     xlab = "Year", ylab = "Degrees (C)", xlim = c(1998, 2003))
```



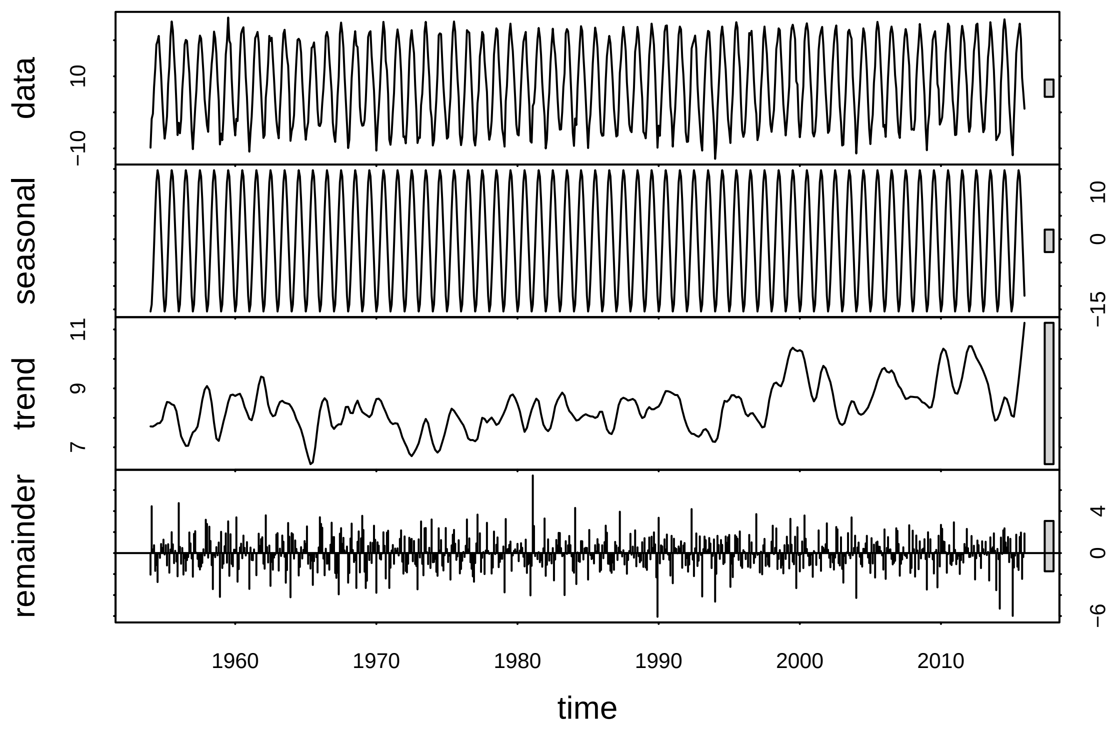
Notice that when we impute the value from the same month last year, we get a much smoother plot that looks more like what we expect based on the non-missing data.

(1c)

```
# create training set and test set
training_set <- window(mean_max_temp_bcf_ts, start = c(1954, 1),
                       end = c(2015, 12))
test_set <- window(mean_max_temp_bcf_ts, start = c(2016, 1), end = c(2016, 12))
# plot decomposition with moving average smoothing
ma_decomp <- decompose(training_set, type = "additive")
plot(ma_decomp)
```



```
# plot Loess decomposition
loess_decomp <- stl(training_set, "periodic")
plot(loess_decomp)
```



(1d)

```
# fit a linear model to the trend component
time <- c(1:744)
trend <- ma_decomp$trend
lm.fit <- lm(trend ~ time)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = trend ~ time)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.53499 -0.46119 -0.02852  0.46585  1.78017
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7.6802144  0.0502146   152.9   <2e-16 ***
## time        0.0017591  0.0001173    15.0   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6703 on 730 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.2357, Adjusted R-squared:  0.2346
## F-statistic: 225.1 on 1 and 730 DF,  p-value: < 2.2e-16
```

```
# 95% confidence interval of the time coefficient
c(0.0017591 - 1.96 * 0.0001173, 0.0017591 + 1.96 * 0.0001173)
```

```
## [1] 0.001529192 0.001989008
```

We see from the regression output that the trend component is estimated to increase by about 0.00176 for every successive month, and that this is significant at the 99.9% level. The coefficient is very close to 0, but the 95% confidence interval does not include 0, which is a sign that there is a trend, although it is very small. We also see from the plot of the trend component that there is a very slight increase in the trend component over time. Based on this information I would use this trend component to make predictions.

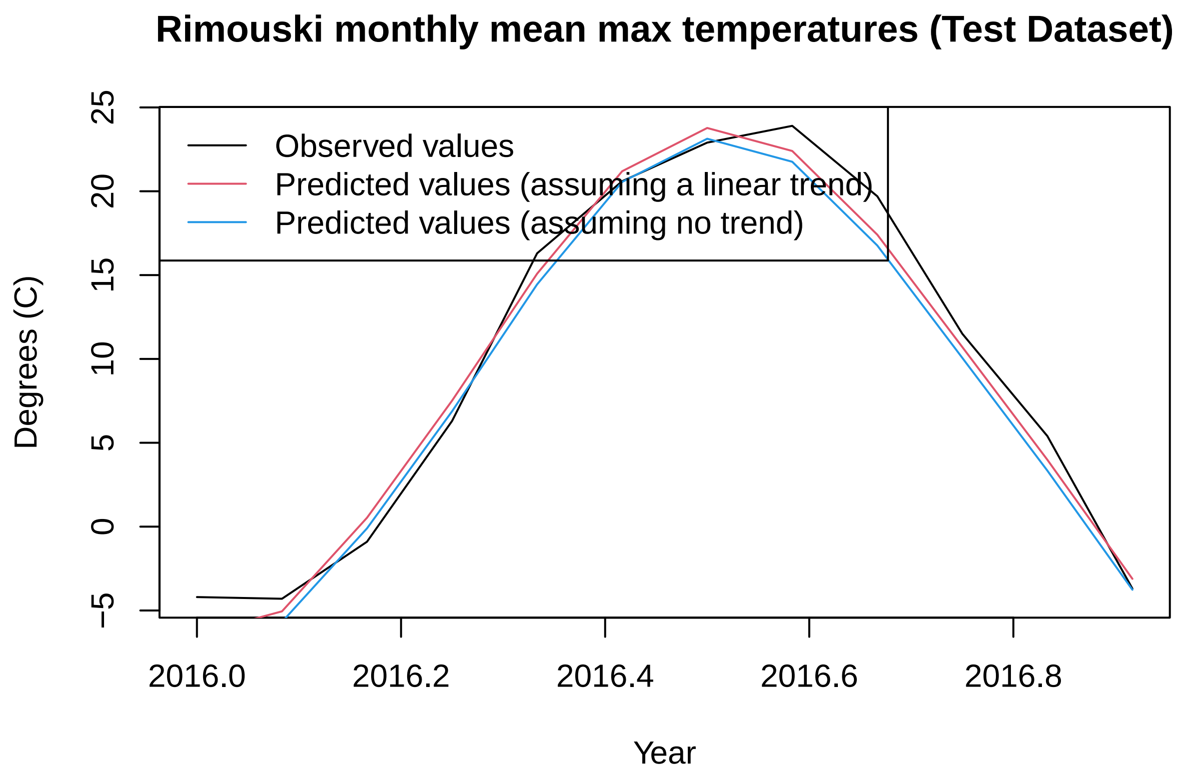
(1e)

```
time_2 <- c(745:756)

# predict test data set assuming there is a linear trend
trend_estimate <- 7.6802144 + 0.0017591 * time_2
prediction_1 <- ts(ma_decomp$seasonal[1:12] + trend_estimate, start = c(2016, 1),
                 frequency = 12)

# predict test data set assuming there is no trend
prediction_2 <- ts(ma_decomp$seasonal[1:12] + mean(mean_max_temp_bcf),
                   start = c(2016, 1), frequency = 12)

# plot observed values, prediction with trend, prediction without trend
plot(test_set,
     main = "Rimouski monthly mean max temperatures (Test Dataset)",
     xlab = "Year",
     ylab = "Degrees (C)")
lines(prediction_1, col = 2)
lines(prediction_2, col = 4)
legend(
  "topleft",
  legend = c(
    "Observed values",
    "Predicted values (assuming a linear trend)",
    "Predicted values (assuming no trend)"),
  lty = 1,
  col = c(1, 2, 4))
```



```r
# MSPE of prediction assuming linear trend
sum((test_set - prediction_1)^2) / length(test_set)
```

```
## [1] 1.841399
```

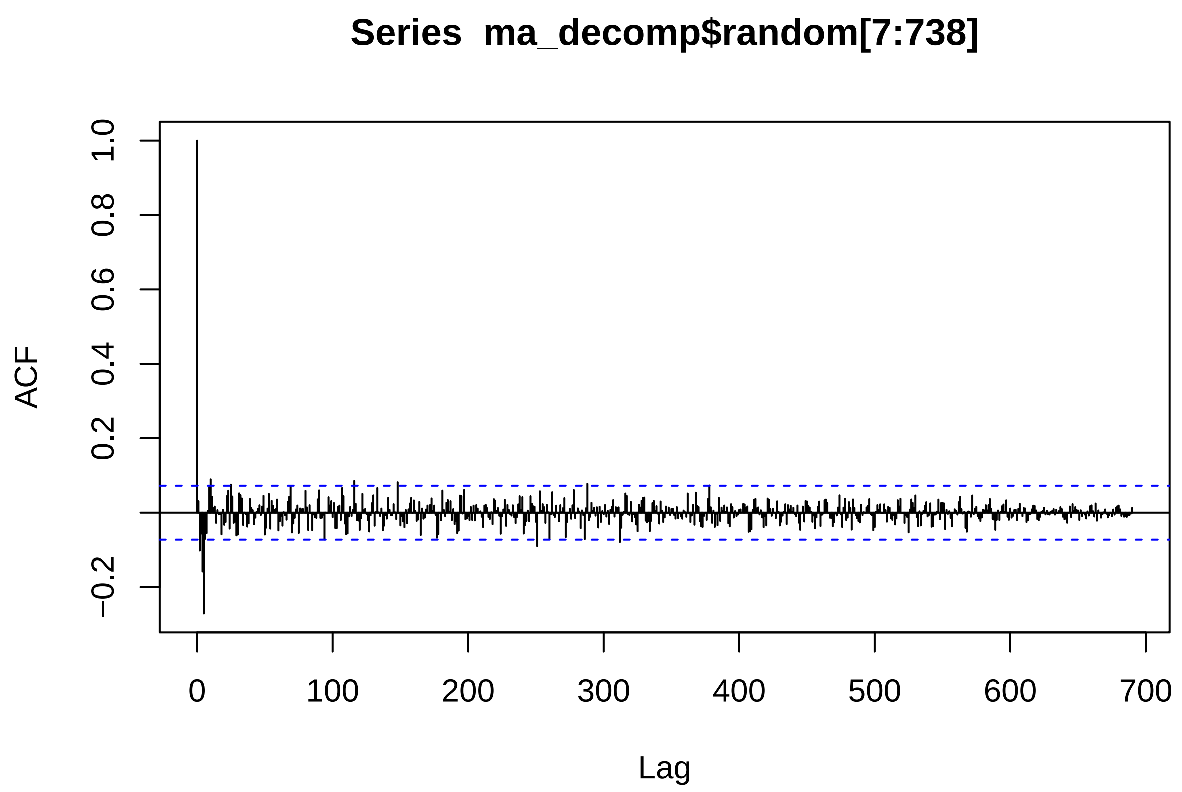
```r
# MSPE of prediction assuming no trend
sum((test_set - prediction_2)^2) / length(test_set)
```

```
## [1] 2.837501
```

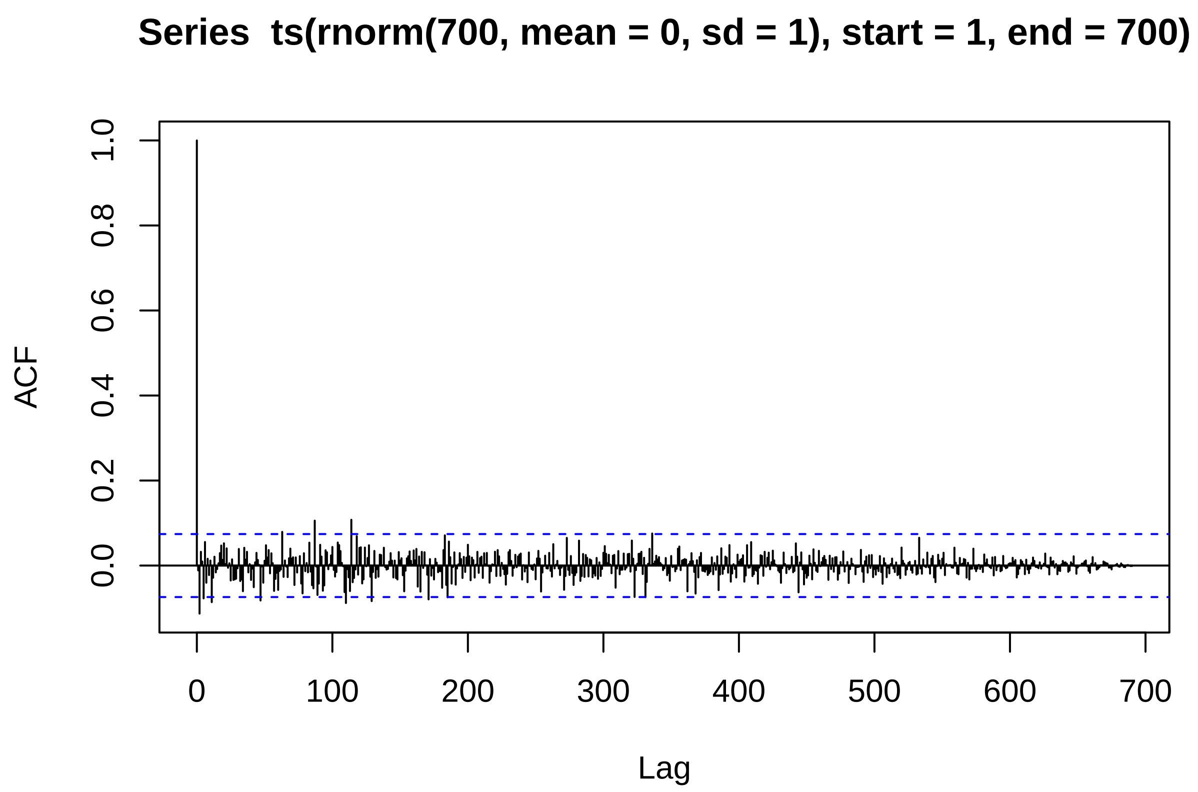
The model assuming that there is a linear trend has a smaller MSPE. This is consistent with the plot, as the values predicted with a linear trend assumption are closer to the observed values than the values predicted without a trend.

(1f)

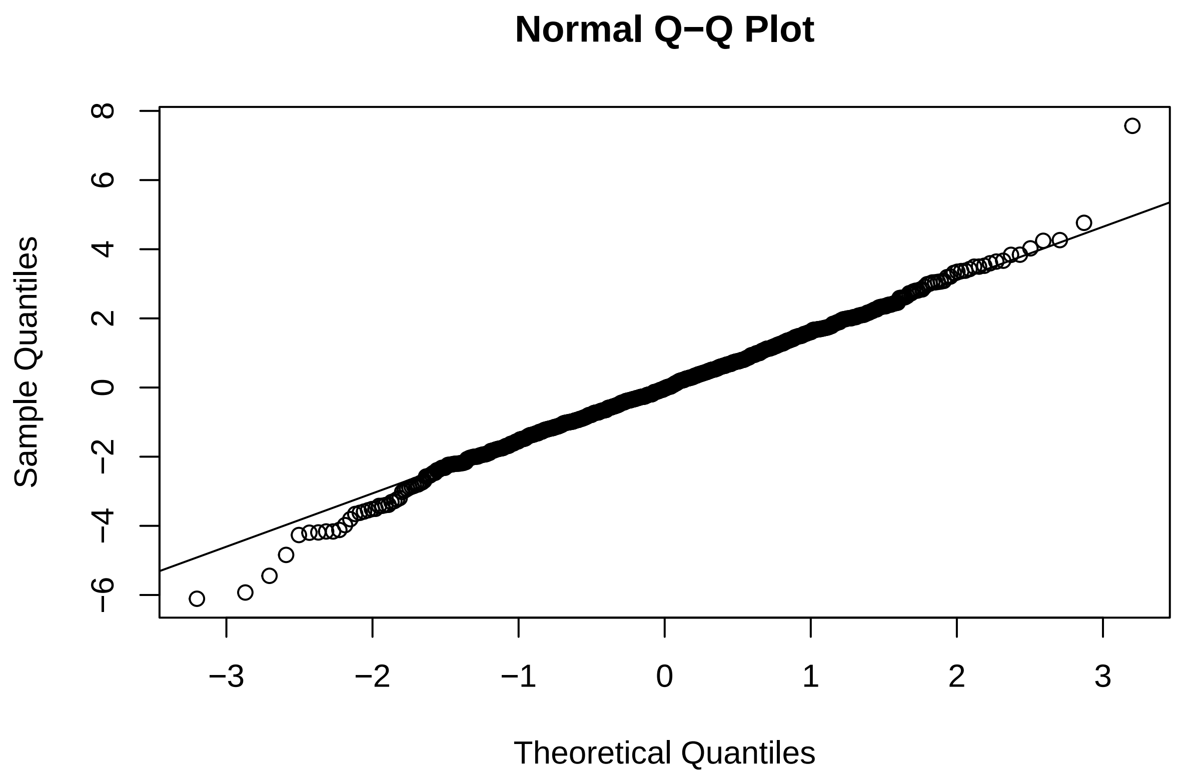
```r
# generate correlogram of the error component from the MA decomposition
acf(ma_decomp$random[7:738], lag.max = 690)
```



```r
# compare with the correlogram of a Gaussian white noise process
acf(ts(rnorm(700,mean=0,sd=1), start=1, end=700), lag.max = 690)
```



```
# generate Q-Q plot
qqnorm(ma_decomp$random[7:738])
qqline(ma_decomp$random[7:738])
```

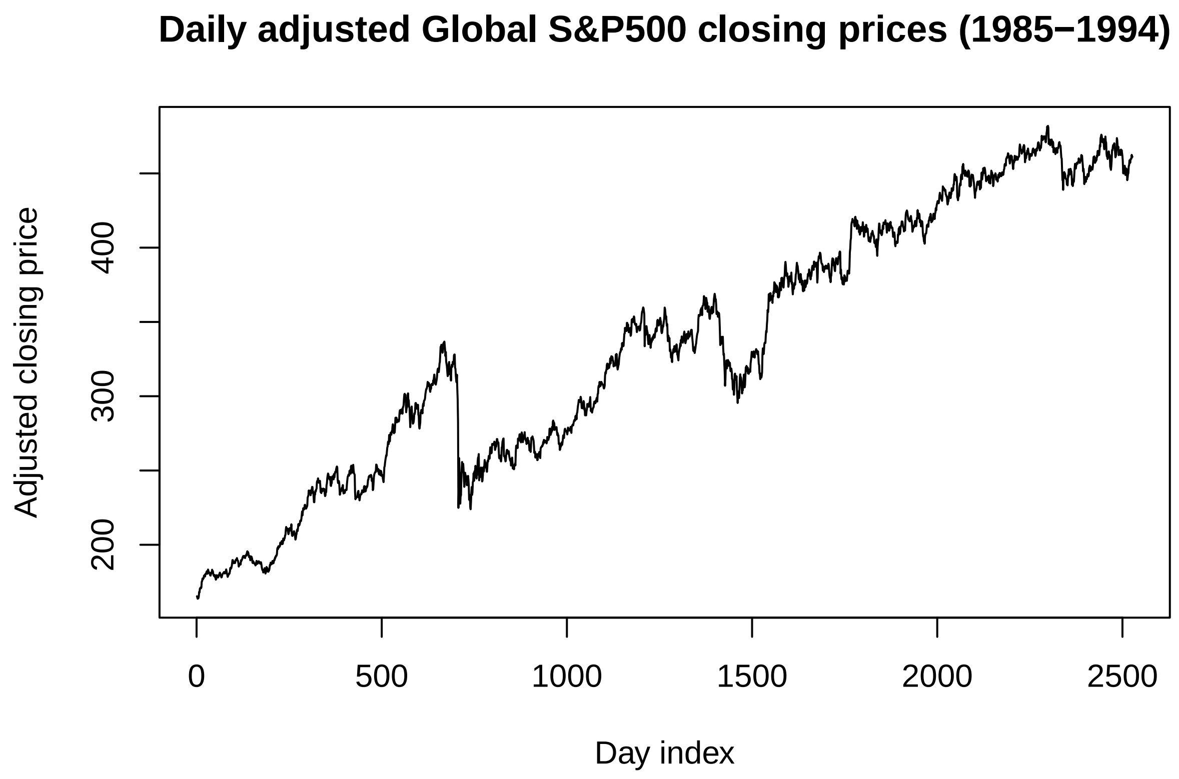


As demonstrated, the correlogram of the error component from the MA decomposition looks very similar to that of an actual white noise process. Also, the Q-Q plot shows that the sample quantiles nearly exactly match that of a normal distribution. Thus it appears that the Gaussian white noise assumption is appropriate.

Question 2

(2a)

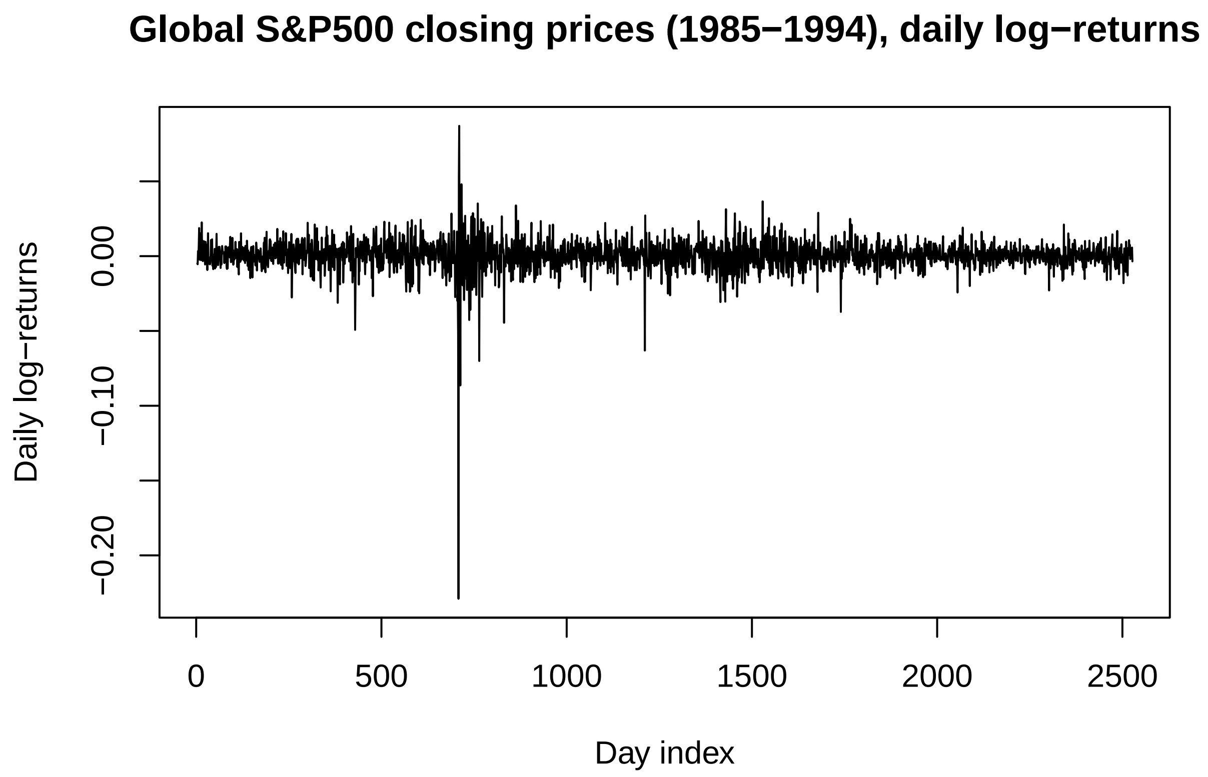
```
# plot the time series
gspc <- read.csv("GSPC.csv", header = TRUE)
daily_close <- ts(gspc$GSPC.Adjusted, start = 1, end = 2527)
plot(daily_close,
     main = "Daily adjusted Global S&P500 closing prices (1985-1994)",
     xlab = "Day index", ylab = "Adjusted closing price")
```



This time series trends upwards and is therefore not appropriate to model as stationary since the expected price increases over time. There does not seem to be a clear seasonal effect.

(2b)

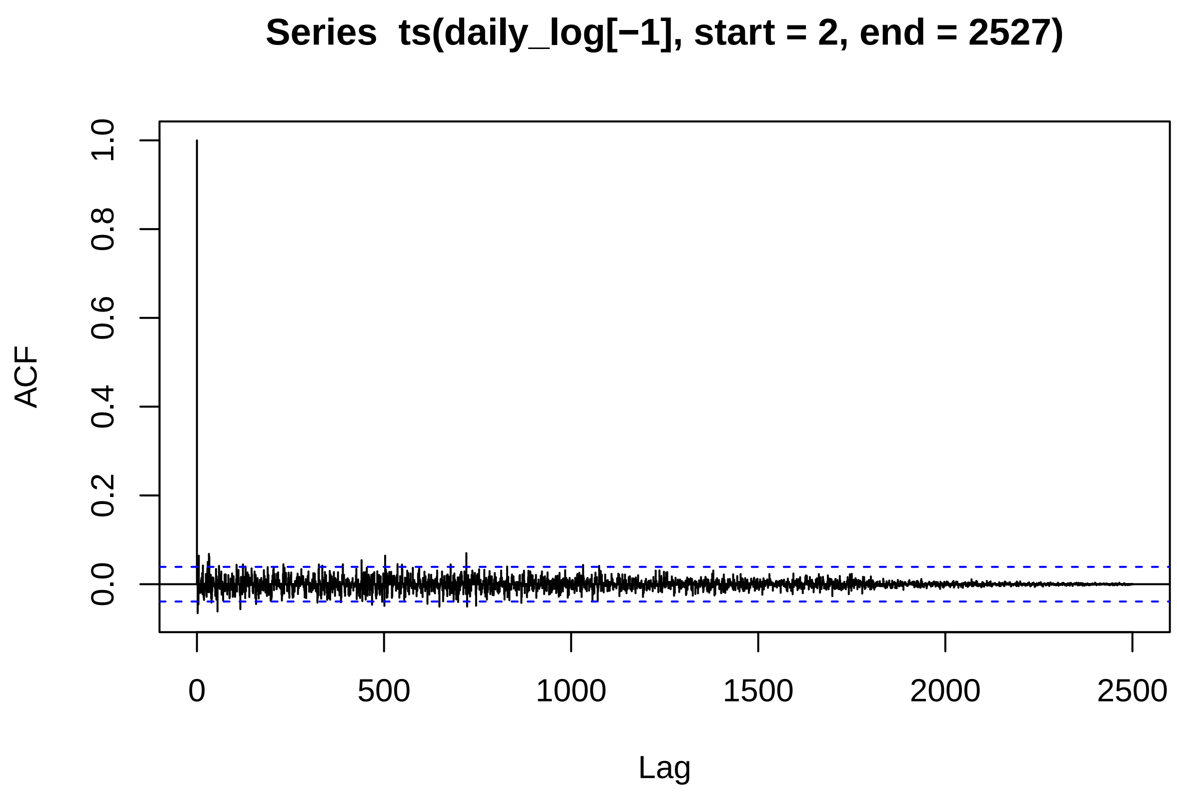
```r
# plot the daily log-returns
daily_log <- c()
for (t in 2:length(daily_close)) {
  daily_log[t] <- log(daily_close[t] / daily_close[t-1])
}
plot(ts(daily_log, start = 2, end = 2527),
     main = "Global S&P500 closing prices (1985-1994), daily log-returns",
     xlab = "Day index", ylab = "Daily log-returns")
```



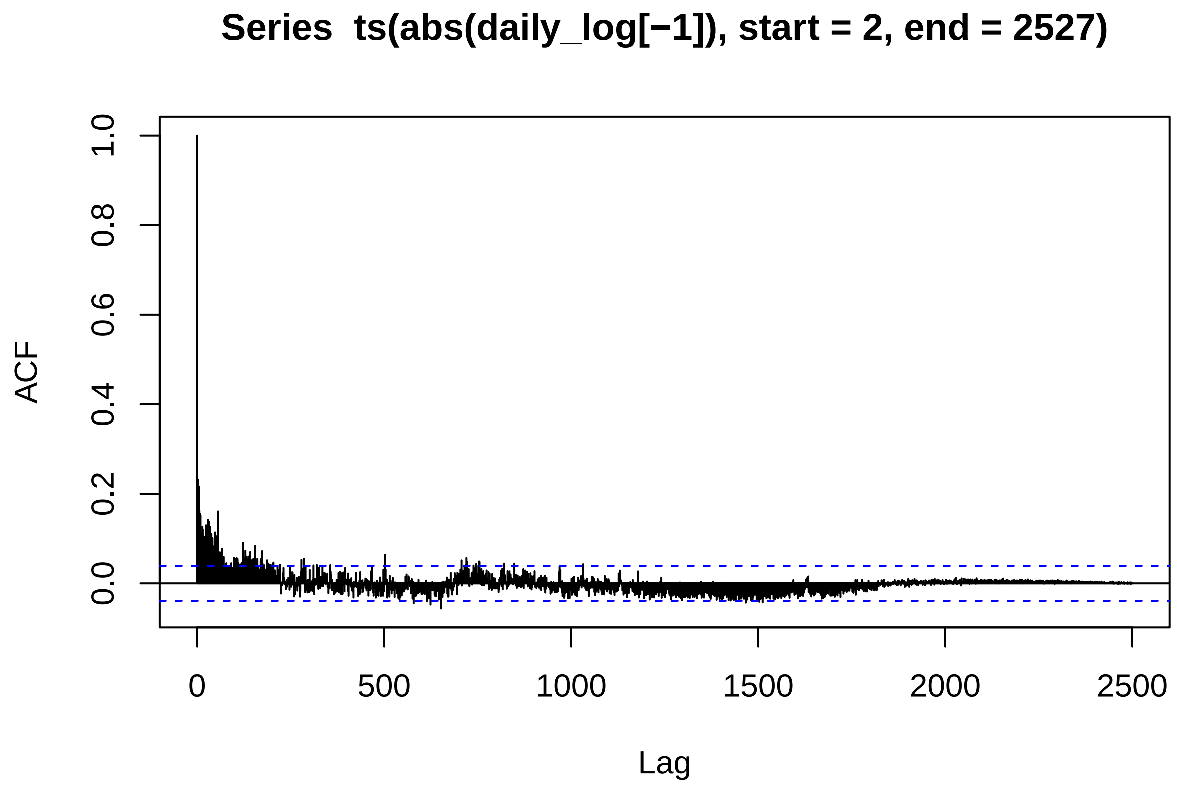
Unlike the daily closing price time series, the daily log-returns appears to have a constant mean, which is the distinction that makes it stationary. And since the daily-log returns has a constant mean, we can more confidently use it to predict the future, as past information is less likely to become irrelevant over time. Analysing the acf of this stationary series can give us more information than analysing the acf of a series with a trend, since the trend inflates the correlation of consecutive values.

(2c)

```r
# generate correlogram for daily log-returns
acf(ts(daily_log[-1], start = 2, end = 2527), lag.max = 2500)
```



```
# generate correlogram for absolute value of daily log-returns
acf(ts(abs(daily_log[-1]), start = 2, end = 2527), lag.max = 2500)
```



The acf of the daily log-return series flips between positive and negative values very often and without a clear pattern. On the other hand, the acf of the absolute value of daily log-returns has sections that are mostly positive or mostly negative. Both acf's get even closer to 0 as the lag increases.