# Statistical Methods MAT3012

# Coursework Draft 1

## Problem Statement:

We are interested in looking at analysing the sales of company X's flagship product over the 42-year period between January 1979 to December 2020. We have data about the total sales value of the product per month, which gives us 504 months worth of data points, or time series.

It is suggested that we model the time series data with a stochastic process of the form:

$$X_t = m_t + S_t + Z_t.$$

Where

- $X_t$ is our sales time series data, each point denoted by $x_1, x_2, \cdots, x_{504}$.
- $m_t$ is entirely non-random, denotes the **trend.**
- $S_t$ is also entirely non-random, denotes **seasonality**.
- $Z_t$ is a white noise process, which we assume to be a **random variable from the Normal distribution**.

The layout of the analysis will be:

1. Estimating trend using a global quadratic polynomial fit.
2. Estimating trend using a linear polynomial fit.
3. Evaluating the methods for estimating trend.
4. Estimating the seasonality component using the better trend estimate.
5. Further Analysis
   - Adding additional parameters to out model.
   - EWA method (and Holt-Winters).

## 1. Estimating trend using a global quadratic polynomial fit.

**Trend** is the component of a stochastic process that represents long-term movement in that process. Usually denoted as $m_t$, it is a set pf deterministic values for all values of $t$.

In this case we will treat $Z_t$ as our random variable with expectation 0. And hence,

$$E[X_t] = E[m_t + Z_t] = E[m_t] + E[Z_t] = m_t + 0 = m_t$$

which means that the trend values represent the mean of the ***stochastic process***.

## 1a) The Global Polynomial Fit

If our underlying trend is approximated well with a polynomial, we can use the global polynomial method to find an estimate for $m_t$ denoted by $\widehat{m_t}$ .

The estimate would be of the form

$$\widehat{m_t} = \beta_0 + \beta_1 t + \beta_2 t^2 + .. \beta_k t^k \ for\ 1 \leq t \leq n,$$

where n is the number of data points, in our case n = 504 sales points; $k$ is the degree of the polynomial; and $\beta_0, \beta_1, \beta_2, \cdots \beta_k$ are the coefficients of the model, which we would need to find.

If we consider a global quadratic polynomial fit, which has degree $k = 2$. The trend model will take the form

$$m_t = \beta_0 + \beta_1 t + \beta_2 t^2$$

and the stochastic process will be of the form

$$X_t = \beta_0 + \beta_1 t + \beta_2 t^2 + Z_t.$$

To get the estimates for our coefficients we use the least squares fitting. Hence we minimize the sum of squares:

$$S(\beta_0, \beta_1) = \sum_{t=1}^{504} (x_t - \beta_0 - \beta_1 t - \beta_2 t^2)^2$$

We differentiate $S(\beta_0, \beta_1, \beta_2)$ with respect to $\beta_0, \beta_1\ and\ \beta_2$:

$$\begin{aligned}
\frac{dS}{d\beta_0} &= \sum_{t=1}^{504} \frac{d}{d\beta_0} (x_t - \beta_0 - \beta_1 t - \beta_2 t^2)^2 \\
&= \sum_{t=1}^{504} -2(x_t - \beta_0 - \beta_1 t - \beta_2 t^2) = -2 \sum_{t=1}^{504} (x_t - \beta_0 - \beta_1 - t - \beta_2 t^2) \\
\frac{dS}{d\beta_1} &= \sum_{t=1}^{504} \frac{d}{d\beta_1} (x_t - \beta_0 - \beta_1 t - \beta_2 t^2)^2 \\
&= \sum_{t=1}^{504} -2t(x_t - \beta_0 - \beta_1 t - \beta_2 t^2) = -2 \sum_{t=1}^{504} t\,(x_t - \beta_0 - \beta_1 t - \beta_2 t^2) \\
\frac{dS}{d\beta_2} &= \sum_{t=1}^{504} \frac{d}{d\beta_2} (x_t - \beta_0 - \beta_1 t - \beta_2 t^2)^2 \\
&= \sum_{t=1}^{504} -2t^2(x_t - \beta_0 - \beta_1 t - \beta_2 t^2) = -2 \sum_{t=1}^{504} t^2\,(x_t - \beta_0 - \beta_1 t - \beta_2 t^2)
\end{aligned}$$

To find the optimal values of the coefficients which minimize the sum of squares and set the differentiated equations to 0 and add hats to them to show that they are estimates. Hence $\hat{\beta}_0,\ \hat{\beta}_1\ and\ \hat{\beta}_2$ must satisfy the normal equations:

$$-2\sum_{t=1}^{504}\left(x_t-\hat{\beta}_0-\hat{\beta}_1 t-\hat{\beta}_2 t^2\right)=0$$
$$-2\sum_{t=1}^{504} t\left(x_t-\hat{\beta}_0-\hat{\beta}_1 t-\hat{\beta}_2 t^2\right)=0$$
$$-2\sum_{t=1}^{504} t^2\left(x_t-\hat{\beta}_0-\hat{\beta}_1 t-\hat{\beta}_2 t^2\right)=0$$

We simplify these to

$$\sum_{t=1}^{504} x_t=\hat{\beta}_0\sum_{t=1}^{504} 1+\hat{\beta}_1\sum_{t=1}^{504} t+\hat{\beta}_2\sum_{t=1}^{504} t^2$$
$$\sum_{t=1}^{504} t\,x_t=\hat{\beta}_0\sum_{t=1}^{504} t+\hat{\beta}_1\sum_{t=1}^{504} t^2+\hat{\beta}_2\sum_{t=1}^{504} t^3$$
$$\sum_{t=1}^{504} t^2\,x_t=\hat{\beta}_0\sum_{t=1}^{504} t^2+\hat{\beta}_1\sum_{t=1}^{504} t^3+\hat{\beta}_2\sum_{t=1}^{504} t^4$$

These simultaneous equations can be represented as matricies which we can solve. Suppose

$$A=\begin{bmatrix}\sum_{t=1}^{504} 1 & \sum_{t=1}^{504} t & \sum_{t=1}^{504} t^2\\ \sum_{t=1}^{504} t & \sum_{t=1}^{504} t^2 & \sum_{t=1}^{504} t^3\\ \sum_{t=1}^{504} t^2 & \sum_{t=1}^{504} t^3 & \sum_{t=1}^{504} t^4\end{bmatrix}$$

We have the vector of the unknows parameters

$$x=\begin{bmatrix}\hat{\beta}_0\\ \hat{\beta}_1\\ \hat{\beta}_2\end{bmatrix}$$

and the solution vector

$$b = \begin{bmatrix} \sum_{t=1}^{504} x_t \\ \sum_{t=1}^{504} t\, x_t \\ \sum_{t=1}^{504} t^2\, x_t \end{bmatrix}.$$

Then we have the inhomogeneous matrix equation

$$Ax = b.$$

In this case we can use our knowlegde from linear algebra to find the particular solution. We take the inverse of $A$ to find the coefficients, hence the coefficients are found by

$$b = A^{-1}x.$$

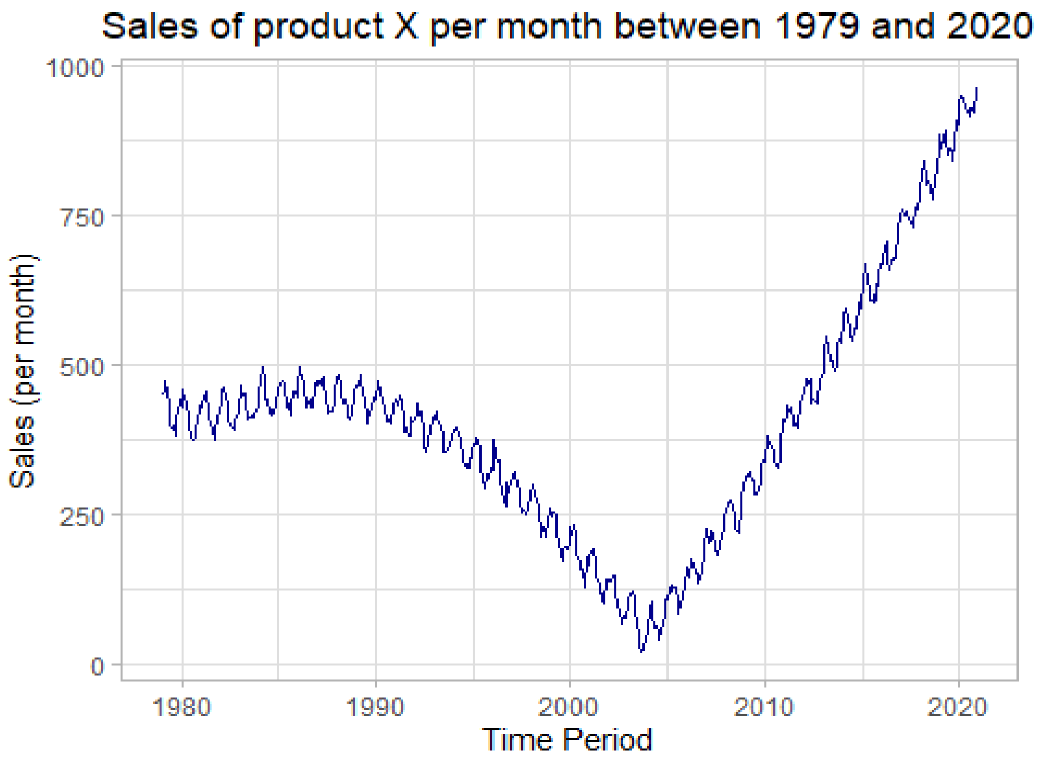
## Initial analysis and Visualization

Reading in out data from a text file.

```r
data <- read.table("Assessed Data 20-21.txt", col.names = c("sales"))

X <- data$sales  ## a time series vector from the sales column
n <- length(X) ## the length of the vector X
```

Plotting the data using the following code to get a better understanding of what it looks like.

```r
## Importing packages
library(ggplot2)
library(dplyr)

library(ggeasy) ## package to help me centre the title of the chart

# Creating a dataframe with Sales and Date values
ts_data <- data.frame(
  months = seq(as.Date("1979-01-01"), as.Date("2020-12-31"), by="months"), # adding
dates
  Sales = X) # the Sales data

# Plotting our data

p <- ggplot(ts_data, aes(x=months, y=Sales)) + ## plot sales (=y) against time (=x)
  geom_line( color="darkblue") +  ## in the colour blue
  xlab("Time Period") + ## labeling the x axis
  ylab("Sales (per month)") +
  ggtitle("Sales of product X per month between 1979 and 2020") +  ## add title
  theme(plot.title = element_text(size=14, hjust = 'right', face="bold.italic")) +
## main title formatting
  theme_bw() +  ## white background
  theme_light() + ## light background lines
  scale_x_date(date_labels = "%Y") +  ## add only the years of the time period
  ggeasy::easy_center_title()  ## centered title

## plot the graph
p
```



## 1b) Fitting a Global Linear Model

The graph doesn't show one specific trend but might stem from two or more trends. A initial guess is that the global quadratic fit is not going to be too simplistic for our data due to that reason. But we will test that hypothesis by fitting the model and compare it to other models.

We can also see some seasonality straight away as the points oscillate evenly.

We will use R's built-in code for fitting linear models, the command: `lm()`. This would do the above calculation to estimate the coefficients and we will examine the results.

```
t <- 1:n ## we create a range of values from 1 to the length as X as T
global_fit <- lm(X~t + I(t^2)) ## here we fit the linear model
global_fit ## output the global linear fit short summary

## output below
```

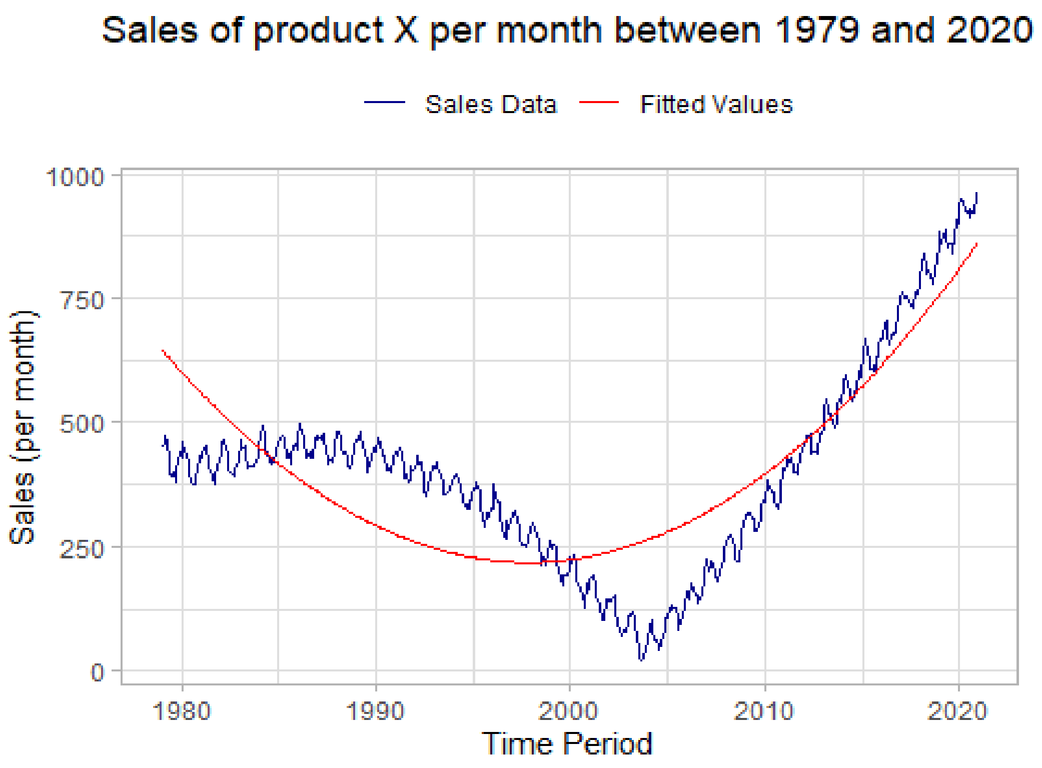
```
##
## Call:
## lm(formula = X ~ t + I(t^2))
##
## Coefficients:
## (Intercept)            t       I(t^2)
##  648.997860    -3.793496     0.008355
```

Hence the global quadratic fit for the trend looks like

$$\widehat{m_t} = 648.9979 - 3.7935t + 0.0083t^2.$$

We now visualize the raw data and fitted values of our global quadratic model to see how it performs.

```r
## save predictions of the model, actual Sales and time stamp in the new data frame
## we want to plot against the time

df <- data.frame(global_fit = predict(global_fit, ts_data), sales=X, time=seq(as.Da
te("1979-01-01"), as.Date("2020-12-31"), by="months"))

## show first 4 rows of our dataframe
## head(df,3)

## Create plot p using ggplot2 and ggeasy libraries we imported above

p <-ggplot(data = df, aes(x = time, y = sales)) +
  geom_line(aes( y= sales, color="Sales Data")) +  ## plot Sales against time
  geom_line(data = df, aes(x=time, y=global_fit, color='Fitted Values')) + ## plot
fitted values against time
  xlab("Time Period") + ## labeling the x axis
  ylab("Sales (per month)") +
  ggtitle("Sales of product X per month between 1979 and 2020") +  ## add title
  theme(plot.title = element_text(size=14, hjust = 'right', face="bold.italic")) +
## main title
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()  + ## centered title
  scale_colour_manual("",
                      breaks = c("Sales Data", "Fitted Values"), ## legend names
                      values = c("darkblue", "red")) ## define legend colors


## plot the graph
p
```



The global quadratic fit is the red line on the chart. We can see that the fit shows a slight downward trend around with a minimum point at the year 2000, whereas the raw data has a trough around the year 2004. The chart correctly shows that there is an upward trend beyond the year 2010 but the trend seems to be growing faster than the quadratic polynomial. Perhaps the trend beyond the trough at year 2004 becomes exponential.

## 3a) Model evaluation

Let's consider the residual values in order to examine the goodness of fit of our mode. The residuals are the deviations between the average value of our predicted value, and a particular observation (of our Sales data). In other words: Residual = Observed value – Predicted value. If you let $\boldsymbol{e_t}$ be the residual value, then

$$e_t \sim N(0, \sigma^2).$$

The residuals should follow the following properties:

- they sum to 0

$$\sum_{i=1}^{n} e_t = 0$$

- the mean of the ith residual is 0:

$$E(e_t) = 0$$

- the variance of each residual is given by

$$Var[e_t] = \sigma^2 \left(1 - \frac{1}{n} - \frac{(x_t - \hat{x})^2}{S_{xx}}\right)$$

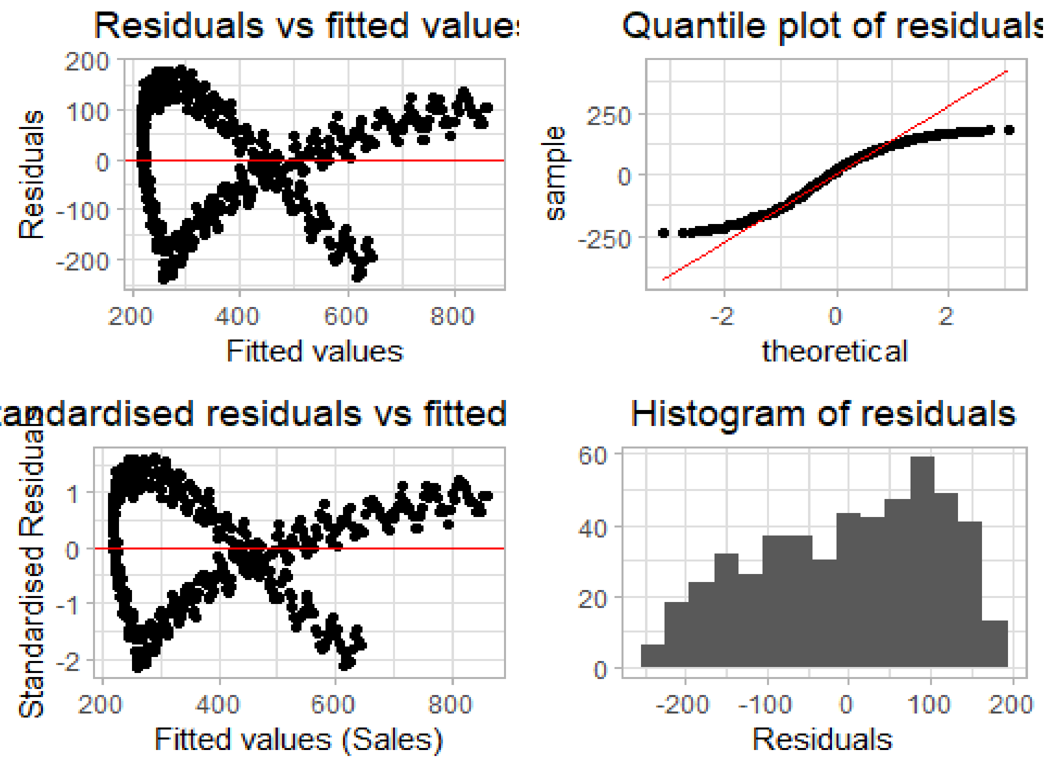
where *n=504* and

$$S_{xx} = \sum_{i=1}^{n} (x_t - \hat{x})^2$$

is the regression sum of squares.

Let us make the data and plot four charts to help analyse our fitted values.

```
## Add package to help plot a grid plot

library(cowplot)

## Add residuals to data
df$residual <- global_fit$residuals

# Make a scatter plot of residuals against fitted values
p1 <- qplot(global_fit$fitted, global_fit$residuals, geom = "point") +
  geom_abline(intercept = 0,slope = 0,colour = "red") +
  ##
  xlab("Fitted values") + ## labeling the x axis
  ylab("Residuals") +
  ggtitle("Residuals vs fitted values") +  ## add title
  theme(plot.title = element_text(size=4, hjust = 'right', face="bold.italic")) + #
main title
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  ggeasy::easy_center_title()

# Make a quantile-quantile plot
p2 <-ggplot(data = df,aes(sample = residual)) +
  geom_qq() +
  geom_qq_line(colour = "red") +
  labs(title = "Quantile plot of residuals") +
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()

p3 <- qplot(global_fit$fitted, rstandard(global_fit), geom = "point") +
  geom_abline(intercept = 0,slope = 0,colour = "red") +
  labs(title = "Standardised residuals vs fitted values", x = "Fitted values (Sales
)", y = "Standardised Residuals") +
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  ggeasy::easy_center_title()

## Make a histogram of the residuals
p4 <- qplot(global_fit$residuals,geom = "histogram",bins = 15) +
  labs(title = "Histogram of residuals",x = "Residuals") +
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()

## Plot the graphs
plots <- plot_grid(p1, p2,p3,p4, ncol=2)
plots
```



The *first plot* (Fitted Values vs Residuals) is a simple scatterplot showing the residuals or errors on the y-axis and the predicted mean values for each observation on the x-axis. The plot shows a clear pattern, hence the assumptions of constant variance and linearity are not satisfied.

The *second plot* is a qq-plot or Quantile-Quantile plot. It shows the theoretical quantiles on the x-axis, i.e. where a residuals would be in terms of a standard normal distribution if the variance was known and the residuals come from a normal distribution. On the y-axis are the sample quantiles of the sample data points - our residuals. You can see some extreme deviations in the tails of the graph which suggests that the residuals do not follow a normal distribution.

The *third plot* is similar to the first plot, but the y-axis are the standardized residuals plotted against fitted values. It shows that there aren't any large outliers in the data but also confirms that the assumptions for our residuals are note met.

The *fourth plot* is a simple histogram of the residuals. The plot of the residuals doesn't follow a normal distribution, bell shaped curve centered around 0 with some variability on each side.

To conclude, the simple quadratic polynomial fit is not a good fit for our Sales data and we should look for an alternative model.

## 2. Estimating trend using a local polynomial fit.

### 2a) The local linear polynomial

Consider estimating the trend $m_t$ by fitting a local linear polynomial to successive groups of 5 observations. In this case, for each value of t, fixed, the trend will be the following:

$$m_{t+j} = \beta_0 + \beta_1 j.$$

Here the coefficients we need to estimate are $\hat{\beta}_0\ and\ \hat{\beta}_1$. Note the *j* term shows the interval of points around a single point $x_t$, in this case $j \in \{-2,-1,0,1,2\}$. More specifically, the trend function is for the interval $t-2 \leq x \leq t+2$ is approximated by a polynomial of degree 1.

However, we are only interested in estimating the trend $m_t$ for singe point $x_t$, hence we set $j = 0$. Hence, in contrast to the global polynomial, we only need to find an estimate for $\widehat{\beta_0}$ , in this case out trend estimate is

$$\widehat{m_t} = \widehat{\beta_0} + \widehat{\beta_1} j = \widehat{\beta_0},$$

and the result will be a linear combination of the time series points.

Similarly to the global polynomial fit, to find the optimal values for the coefficients we minimize the sum of squares for fixed point t:

$$S = \sum_{j=-2}^{2} \left(x_{t+j} - \beta_0 - \beta_1 j\right)^2.$$

To find the optimal values for the coefficients, we differentiate $S$ with respect to $\beta_0$ and $\beta_1$ :

$$\begin{aligned}\frac{dS}{d\beta_0} &= \sum_{j=-2}^{2} \frac{d}{d\beta_0}\left(x_{t+j} - \beta_0 - \beta_1 j\right)^2 \\ &= \sum_{j=-2}^{2} -2\left(x_{t+j} - \beta_0 - \beta_1 j\right) = -2\sum_{j=-2}^{2}\left(x_{t+j} - \beta_0 - \beta_1 j\right) \\ \frac{dS}{d\beta_1} &= \sum_{j=-2}^{2} \frac{d}{d\beta_1}\left(x_{t+j} - \beta_0 - \beta_1 j\right)^2 \\ &= \sum_{j=-2}^{2} -2j\left(x_{t+j} - \beta_0 - \beta_1 j\right) = -2\sum_{j=-2}^{2}\left(j x_{t+j} - \beta_0 j - \beta_1 j^2\right).\end{aligned}$$

We get similar normal equations as we did in the global quadratic case. We set the equations to 0 and add hats to the coefficients to indicate that they are now estimates, we get that the two coefficients satisfy

$$\begin{aligned}-2\sum_{j=-2}^{2}\left(x_{t+j} - \hat{\beta}_0 - \hat{\beta}_1 j\right) = 0 \\ -2\sum_{j=-2}^{2}\left(j x_{t+j} - \hat{\beta}_0 j - \hat{\beta}_1 j^2\right) = 0.\end{aligned}$$

This can be further simplified to

$$\begin{aligned}\sum_{j=-2}^{2} x_{t+j} &= \hat{\beta}_0 \sum_{j=-2}^{2} 1 + \hat{\beta}_1 \sum_{j=-2}^{2} j \\ \sum_{j=-2}^{2} j\, x_{t+j} &= \hat{\beta}_0 \sum_{j=-2}^{2} j + \hat{\beta}_1 \sum_{j=-2}^{2} j^2 .\end{aligned}$$

In this case we know that

$$\textstyle\sum_{j=-2}^{2} 1 = 5, \sum_{j=-2}^{2} j = 0, \sum_{j=-2}^{2} j^2 = 10.$$

Hence, we simplify the above even further

$$\begin{aligned}\sum_{j=-2}^{2} x_{t+j} &= 5\hat{\beta}_0 \\ \sum_{j=-2}^{2} j\, x_{t+j} &= 10\hat{\beta}_1.\end{aligned}$$

In summary, the trend estimate for a single fixed point can be found using the $\widehat{\beta_0}$ coefficient, which is calculated using

$$\widehat{m_t} = \widehat{\beta_0} = \frac{1}{5}(x_{t-2} + x_{t-1} + x_t + x_{t+1} + x_{t+2}).$$

We could calculate $\widehat{\beta_1}$ but in this case we do not need to because we are only interested in a single value to estimate the trend $\widehat{m_t}$ at a fixed point.

It is important to note that since we take an evenly spaced interval (of 5 observations) around a single point, to find the mean of the trend estimate at that point, we would not be able to use this method for the first two and last 2 observations.

The iterative method would start at point $t = 3$ where we can draw the first two data points and calculate the average at that point, it is incremented once until the point $n - 2$ where we can use the last two data points, but would not be able to beyond that $n - 2$ value. Hence, in our case, the trend estimate $\widehat{m_t}$ is defined for $t = 3,4, \cdots ,502$.

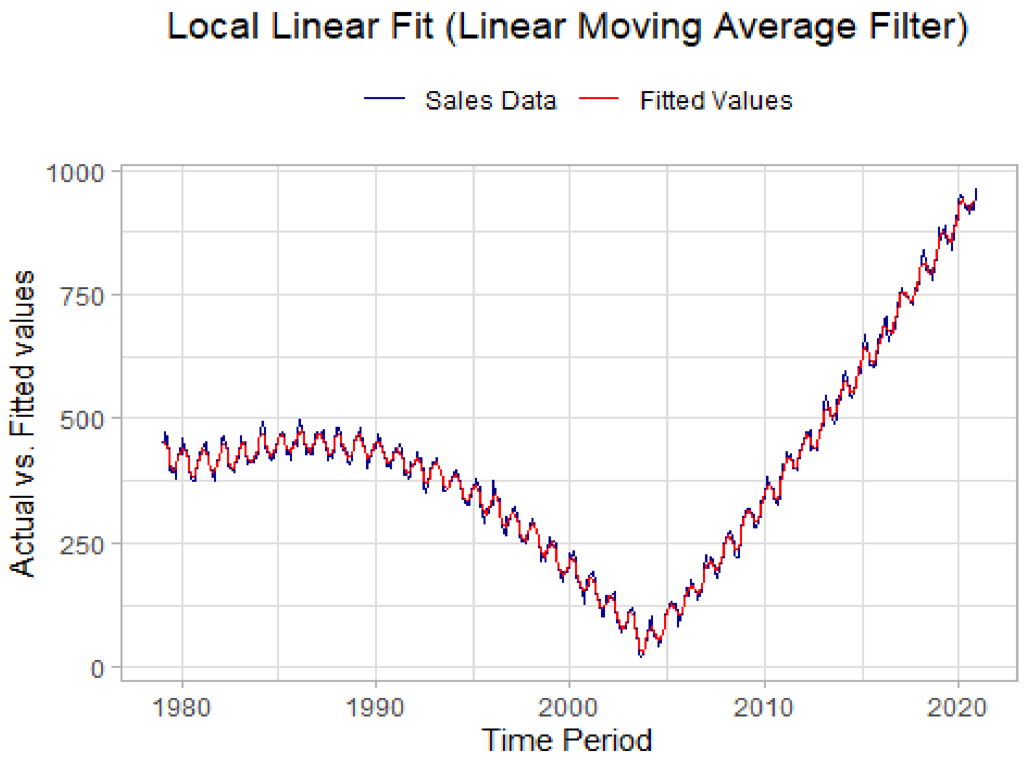
## 2b) Fitting a Local Linear Polynomial Model (to successive group of 5 observations)

We expect that this method (symmetric simple moving average) would perform better than the Global Polynomial fit. However, the method also assumes some degree of smoothness in the trend of our observations, at least we expect smoothness around the 5 observations that help estimate a single point trend.

```
X <- df$sales  ## a time series vector from the sales column
n <- length(X) # the length of the vector X

## vector to store the trend estimates
vect <- (1:n)

## Setting the first two and last two to 'NA' because they would not be calculated
vect[1] <- NA; vect[2] <- NA; vect[n-1] <- NA; vect[n] <- NA

## Set starting point of iteration
t <- 3

## iteration to update our empty vector with the trend estimates

## the while loop does calculation of trend estimate at each point in the time seri
es data
## it starts at t=3 and loops/increments until t reaches n-1=503th point
while(t < n-1)
{
  ## Step 1
  vect[t] <- (X[t-2]+X[t-1]+X[t]+X[t+1]+X[t+2] ) / 5  ## calculating the moving ave
rage at each fixed point and updating our empty vector

  ## Step 2
  t <- t+1 ## go to the next point t+1 and repeat step 1
}
```

We expect to see a better fit:

```
## Add new vector to our dataframe so we can plot our data
df$linear_polynomial <- vect

## Create plot p using ggplot2 and ggeasy libraries we imported above

p <-ggplot(data = df, aes(x = time, y = sales)) +
  geom_line(aes( y= sales, color="Sales Data")) +  ## plot Sales against time
  geom_line(data = df, aes(x=time, y=linear_polynomial, color='Fitted Values')) + #
# plot fitted values against time
  xlab("Time Period") + ## labeling the x axis
  ylab("Actual vs. Fitted values") +
  ggtitle("Local Linear Fit (Linear Moving Average Filter)") +  ## add title
  theme(plot.title = element_text(size=10, hjust = 'right', face="bold.italic")) +
## main title
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()  + ## centered title
  scale_colour_manual("",
                      breaks = c("Sales Data", "Fitted Values"), ## legend names
                      values = c("darkblue", "red")) ## define legend colors
## plot the graph
p

## Warning: Removed 4 row(s) containing missing values (geom_path).
```

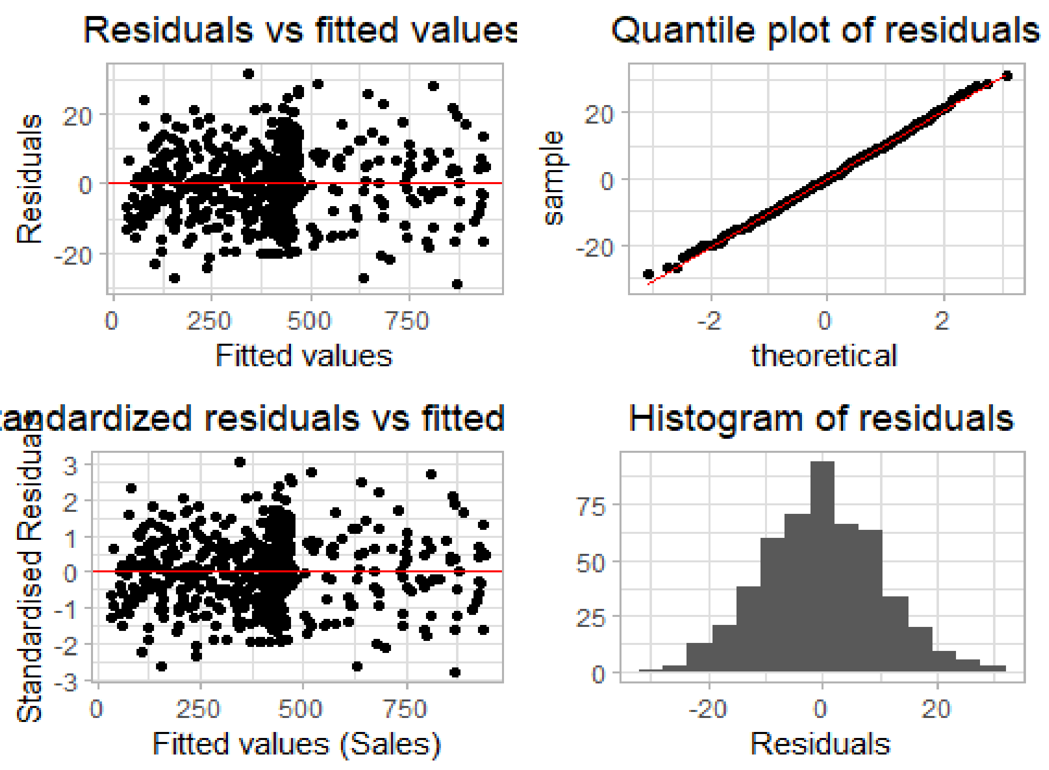


We can see that the red line which show the trend of the fitted values is a much better fit than the global method we examined earlier. However, there might be some degree of overfitting using this method because it follows the real sales data extremely well as it might have absorbed some random fluctuation into the model.

## 3b) Model Evaluation

The model seems to be a better fit for our data but we will still want to undertsnad how it works better.

```
## Calculating the residuals
df$local_residual <- df$sales - df$linear_polynomial

## Calculating the standardized residuals

## assign local residual to a vector
local_res <- df$local_residual[3:(n-2)]  ## exclude the first two and last two vect
or values
## since they are NA

df$local_std_res <- NA  ## create an empty column in the data frame

## standardized residuals are found by dividing each residual in vector 'local_res'
by
## the standard deviation of the residuals calculated by 'sd(local_res)'
## we assign the new (calculated) vector to a the new column in our data frame call
ed 'local_std_res'
## but we only assign it to the the rows from 3 to n-2=502 because the first and la
st two rows are NA

df$local_std_res[3:(n-2)] <- local_res / sd(local_res)


## Make a scatter plot of residuals against fitted values
p1 <- qplot(df$linear_polynomial, df$local_residual, geom = "point") +
  geom_abline(intercept = 0,slope = 0,colour = "red") +
  xlab("Fitted values") + ## labeling the x axis
  ylab("Residuals") +
  ggtitle("Residuals vs fitted values") +  ## add title
  theme(plot.title = element_text(size=4, hjust = 'right', face="bold.italic")) + #
# main title
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  ggeasy::easy_center_title()

## Make a quantile-quantile plot
p2 <-ggplot(data = df,aes(sample = local_residual)) +
  geom_qq() +
  geom_qq_line(colour = "red") +
  labs(title = "Quantile plot of residuals") +
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()

p3 <- qplot(df$linear_polynomial, df$local_std_res, geom = "point") +
  geom_abline(intercept = 0,slope = 0,colour = "red") +
  labs(title = "Standardized residuals vs fitted values", x = "Fitted values (Sales
)", y = "Standardised Residuals") +
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  ggeasy::easy_center_title()

# Make a histogram of the residuals
p4 <- qplot(df$local_residual,geom = "histogram",bins = 15) +
  labs(title = "Histogram of residuals",x = "Residuals") +
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()

# Plot the plots
plots <- plot_grid(p1, p2,p3,p4, ncol=2)

## Warning: Removed 4 rows containing missing values (geom_point).

## Warning: Removed 4 rows containing non-finite values (stat_qq).

## Warning: Removed 4 rows containing non-finite values (stat_qq_line).

## Warning: Removed 4 rows containing missing values (geom_point).

## Warning: Removed 4 rows containing non-finite values (stat_bin).

plots
```



We can see that the residual plots from the local linear fit method are better than the ones in 3a). The assumptions made about the residuals that they are independent and normally distributed in section 3a) look like they have been met.

From plot 1 (Residuals vs fitted values) and plot 3 (Standardized residuals vs fitted values), we can see that the residuals are scattered randomly against the fitted values in both cases, (somewhat) meeting the assumption that

$$e_t \sim N(0, \sigma^2).$$

Plot 1 shows a significant reduction in the residual values from 200 and -200 using the global quadratic method to 20 and -20 using the local linear one.

Furthermore plot 3 shows some residuals which can be potential outliers because they lie beyond the values of 2 and -2 but the sales at thos points could also be affected by seasonality which could explain the wide spread.

We should note that the residuals below the Predicted Sales of 500, are a lot more than the residuals beyond that sales point because of the shape of the raw sales data. It is slightly quadratic and the sales value only increases beyond 500 after the year 2010. Hence plots 1 and 3 shows a good spread of residuals.

The qq-plot in plot 2, and the histogram in plot 4 show that the residuals in the local linear fit are coming from a normal distribution because there are no tails in the qq-plot and the histogram is centered at 0 and evenly spread assimilating a bell-shaped curve.

To sum up, the local linear fit is a better fit for predicting the trend of the company's sales data. For this reason, we will use the local linear fit to conduct further analysis to estimate the seasonality of the data in the next section.

## 4. Seasonality in the data

The **seasonal period** is the smallest positive integer such that a seasonal pattern recurs every s steps apart. The difference between seasonality and trend is that seasonality has a recurring pattern. For example, for the sales data we might have the seasonal period, s=12 since we have monthly sales data.

Here we will use the time series $y_t$ where

$$y_t = x_t - \widehat{m_t}$$

to calculate the seasonality estimates. The model becomes

$$Y_t = S_t + Z_t$$

where the $S_t$ describes the seasonal component and $Z_t$ is a white noise process.

We will look at seasonality estimates with period 12.

### 4a) Obtaining the seasonality estimates

We assume that we have constant values within season and hence

$$S_1 = S_{s+1} = \cdots = S_{(k-1)s+1}, S_2 = S_{s+2} = \cdots = S_{(k-1)s+2}, \cdots, S_{12} = S_{s+12} = \cdots = S_{(k-1)s+12}.$$

This means that $n = ks = 504$ so that we have k=42 complete cycles of seasons. Hence we have that

$$X_1, X_{s+1}, X_{2s+1}, \cdots, X_{(k-1)s+1}$$

all have the same cycle, and

$$X_2, X_{s+2}, X_{2s+2}, \cdots, X_{(k-1)s+2}$$

all have the same cycle but different to the previous one, and so on.

This model is simple because we have that s=12 and n=504 and $s < n$. We write the stochastic process as

$$X_{js+1} = S_i + Y_{js+1} \; for \; i \in 1,2,\cdots,12, \; j \in 0,1,\cdots,41.$$

To find the optimal seasonality estimates, we need to minimize the sum of squares

$$
\begin{aligned}
S \quad &= \sum_{i=1}^{ks} (x_i - S_i)^2 \\
&= \sum_{j=0}^{k-1} \left(x_{js+1} - S_1\right)^2 + \sum_{j=0}^{k-1} \left(x_{js+2} - S_2\right)^2 \\
&+ \sum_{j=0}^{k-1} \left(x_{js+3} - S_3\right)^2 + \sum_{j=0}^{k-1} \left(x_{js+4} - S_4\right)^2 \\
&+ \sum_{j=0}^{k-1} \left(x_{js+5} - S_5\right)^2 + \sum_{j=0}^{k-1} \left(x_{js+6} - S_6\right)^2 + \\
&+ \sum_{j=0}^{k-1} \left(x_{js+7} - S_7\right)^2 + \sum_{j=0}^{k-1} \left(x_{js+8} - S_8\right)^2 + \\
&+ \sum_{j=0}^{k-1} \left(x_{js+9} - S_9\right)^2 + \sum_{j=0}^{k-1} \left(x_{js+10} - S_{10}\right)^2 + \\
&+ \sum_{j=0}^{k-1} \left(x_{js+11} - S_{11}\right)^2 + \sum_{j=0}^{k-1} \left(x_{js+12} - S_{12}\right)^2 .
\end{aligned}
$$

Hence, we differentiate $S$ with respect to $S_1, S_2, \cdots, S_{12}$ and set the differentials to 0 to get the normal equations. It is easy to see that $S_1$ is only contained in the first sum of squares, $S_2$ is in the second sum of squares and so on. Hence the differentials are

$$\frac{dS}{dS_1} = \sum_{j=0}^{k-1} 2\left(x_{js+1} - S_1\right)(-1)$$
$$\frac{dS}{dS_2} = \sum_{j=0}^{k-1} 2\left(x_{js+2} - S_2\right)(-1)$$
$$\frac{dS}{dS_3} = \sum_{j=0}^{k-1} 2\left(x_{js+3} - S_3\right)(-1)$$
$$\frac{dS}{dS_4} = \sum_{j=0}^{k-1} 2\left(x_{js+4} - S_4\right)(-1)$$
$$\frac{dS}{dS_5} = \sum_{j=0}^{k-1} 2\left(x_{js+5} - S_5\right)(-1)$$
$$\frac{dS}{dS_6} = \sum_{j=0}^{k-1} 2\left(x_{js+6} - S_6\right)(-1)$$
$$\frac{dS}{dS_7} = \sum_{j=0}^{k-1} 2\left(x_{js+7} - S_7\right)(-1)$$
$$\frac{dS}{dS_8} = \sum_{j=0}^{k-1} 2\left(x_{js+8} - S_8\right)(-1)$$
$$\frac{dS}{dS_9} = \sum_{j=0}^{k-1} 2\left(x_{js+9} - S_9\right)(-1)$$
$$\frac{dS}{dS_{10}} = \sum_{j=0}^{k-1} 2\left(x_{js+10} - S_{10}\right)(-1)$$
$$\frac{dS}{dS_{11}} = \sum_{j=0}^{k-1} 2\left(x_{js+11} - S_{11}\right)(-1)$$
$$\frac{dS}{dS_{12}} = \sum_{j=0}^{k-1} 2\left(x_{js+12} - S_{12}\right)(-1).$$

The normal equations are

$$-2\sum_{j=0}^{k-1}\left(x_{js+1}-\widehat{S_1}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+2}-\widehat{S_2}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+3}-\widehat{S_3}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+4}-\widehat{S_4}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+5}-\widehat{S_5}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+6}-\widehat{S_6}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+7}-\widehat{S_7}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+8}-\widehat{S_8}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+9}-\widehat{S_9}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+10}-\widehat{S_{10}}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+11}-\widehat{S_{11}}\right)=0$$
$$-2\sum_{j=0}^{k-1}\left(x_{js+12}-\widehat{S_{12}}\right)=0.$$

We know that $\sum_{j=0}^{k-1}1=42$ so the estimates for the seasonal components with period 12 are

$$\widehat{S_1} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+1}$$

$$\widehat{S_2} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+2}$$

$$\widehat{S_3} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+3}$$

$$\widehat{S_4} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+4}$$

$$\widehat{S_5} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+5}$$

$$\widehat{S_6} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+6}$$

$$\widehat{S_7} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+7}$$

$$\widehat{S_8} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+8}$$

$$\widehat{S_9} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+9}$$

$$\widehat{S_{10}} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+10}$$

$$\widehat{S_{11}} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+11}$$

$$\widehat{S_{12}} = \frac{1}{42}\sum_{j=0}^{k-1} x_{js+12},$$

which means that the best seasonal component estimates are the averages of all the time points in a season $i$.

## 4b) Calculating the seasonal estimates

We can create the seasonal averages for a local linear trend by excluding the first two and last two sales values from the seasonal estimate calculation because those values are set as $NA$.

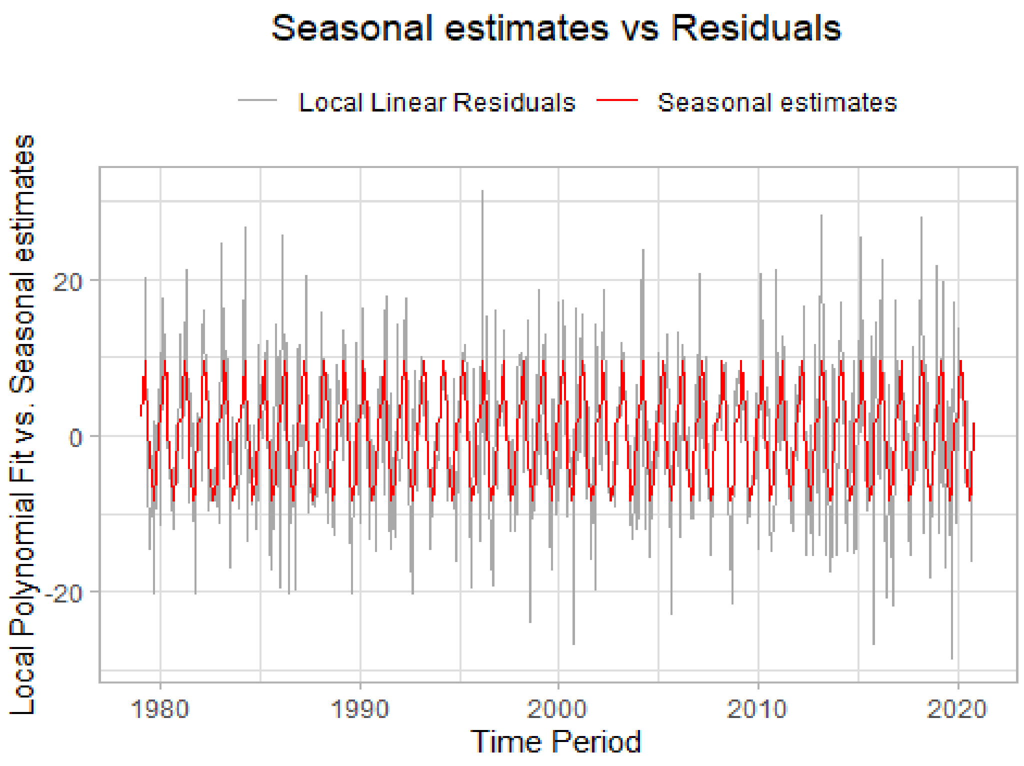
```r
## define a vector that contains the local linear fit residuals,
## excluding the first and last two data points as they are NA
X <- df$local_residual[3:502]

## create an indicator value
ind <- rep(1:12,42) ## repeat values from 1 to 12, 42 times - create a vector of 50
4 indicator values

## Exclude the first two and last two data points from our indicator vector
ind <- ind[3:502]

## take the average over all values with the same indicator variable
avr_values <- c( mean(X[ind == 1]), mean(X[ind == 2]), mean(X[ind == 3]), mean(X[in
d == 4]), mean(X[ind ==5]), mean(X[ind == 6]), mean(X[ind == 7]), mean(X[ind == 8])
, mean(X[ind == 9]), mean(X[ind == 10]), mean(X[ind == 11]), mean(X[ind == 12]))

## Create seasonal estimates by repeating the averages k=42 number of times
sfit <- rep(avr_values, 42)

df$seasonal_fit <- sfit

## head(df,5)
```

### 4c) Examine the goodness of fit

We can now examine the goodness of for our seasonal estimates by plotting the values:

```r
## Create plot p using ggplot2 and ggeasy libraries we imported above

p <-ggplot(data = df, aes(x = time, y = local_residual)) +
  geom_line(aes( y= local_residual, color="Local Linear Residuals")) +  ## plot Sal
es against time
  geom_line(data = df, aes(x=time, y=seasonal_fit, color='Seasonal estimates')) + #
# plot fitted values against time
  ##geom_point(data = df, aes(x=time, y=local_residual, color='Local Linear Residua
ls'), pch=4) +
  xlab("Time Period") + ## labeling the x axis
  ylab("Local Polynomial Fit vs. Seasonal estimates") +
  ggtitle("Seasonal estimates vs Residuals") +  ## add title
  theme(plot.title = element_text(size=10, hjust = 'right', face="bold.italic")) +
# main title
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()  + ## centered title
  scale_colour_manual("",
                      breaks = c("Local Linear Residuals", "Seasonal estimates"), #
# legend names
                      values = c("darkgrey", "red")) ## define legend colors
```

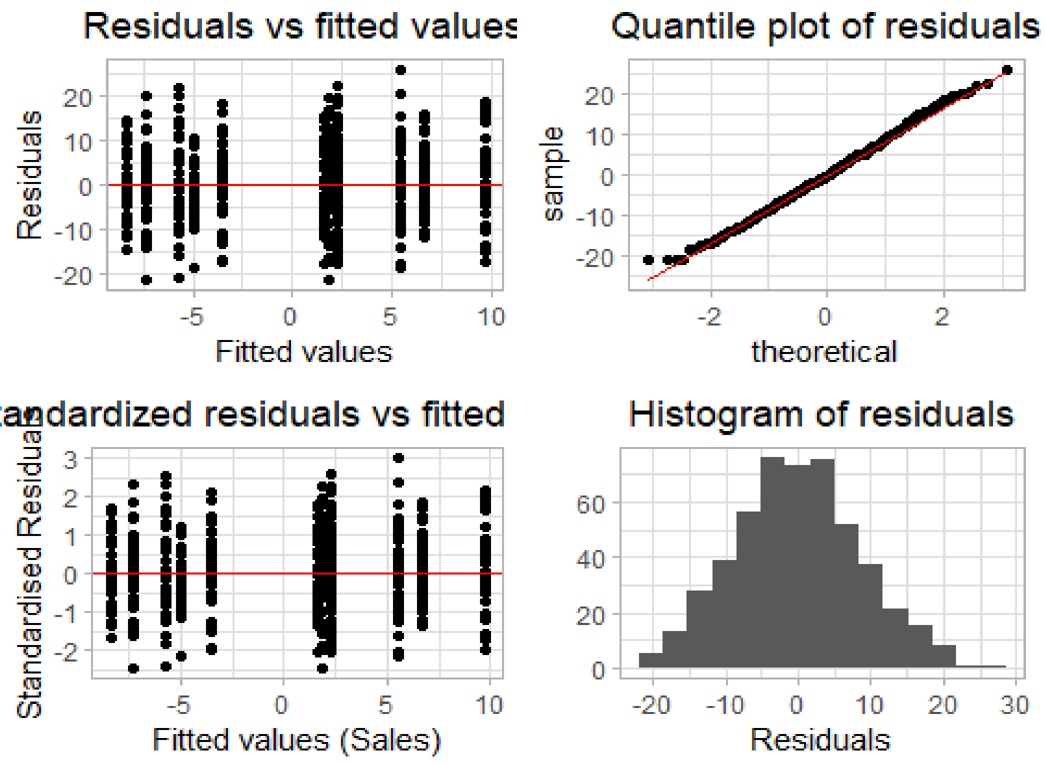
```
## plot the graph
p

## Warning: Removed 4 row(s) containing missing values (geom_path).
```



This plot shows that the seasonality estimates model the peaks and troughs of the residual data quite well. But not all values are close to the seasonal estimates, there are still some larger peaks so we can further analyze the 'leftover' residuals after we have fitted the seasonality estimates.

```
df$seas_res <- df$local_residual - df$seasonal_fit

## Calculating the standardized residuals

## assign local residual to a vector
seas_res <- df$seas_res[3:(n-2)]  ## exclude the first two and last two vector valu
es

df$Seas_std_res <- NA  ## create an empty column in the data frame

## Standardized residuals are found by dividing each residual in the vector 'seas_r
es' by
## the standard deviation of the residuals calculated by 'sd(seas_res)'.
## We assign the new (calculated) vector to a the new column in our data frame call
ed 'Seas_std_res'
## but we only assign it to the the rows from 3 to n-2=502 because the first and la
st two rows are NA
```

```r
df$Seas_std_res[3:(n-2)] <- seas_res / sd(seas_res)

# Make a scatter plot of residuals against fitted values
p1 <- qplot(df$seasonal_fit, df$seas_res, geom = "point") +
  geom_abline(intercept = 0,slope = 0,colour = "red") +
  ##
  xlab("Fitted values") + ## labeling the x axis
  ylab("Residuals") +
  ggtitle("Residuals vs fitted values") +  ## add title
  theme(plot.title = element_text(size=4, hjust = 'right', face="bold.italic")) + #
main title
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  ggeasy::easy_center_title()

# Make a quantile-quantile plot
p2 <-ggplot(data = df,aes(sample = seas_res)) +
  geom_qq() +
  geom_qq_line(colour = "red") +
  labs(title = "Quantile plot of residuals") +
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()

p3 <- qplot(df$seasonal_fit, df$Seas_std_res, geom = "point") +
  geom_abline(intercept = 0,slope = 0,colour = "red") +
  labs(title = "Standardized residuals vs fitted values", x = "Fitted values (Sales
)", y = "Standardised Residuals") +
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  ggeasy::easy_center_title()

# Make a histogram of the residuals
p4 <- qplot(df$seas_res,geom = "histogram",bins = 15) +
  labs(title = "Histogram of residuals",x = "Residuals") +
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()

# Plot the graphs
plots <- plot_grid(p1, p2,p3,p4, ncol=2)

## Warning: Removed 4 rows containing missing values (geom_point).

## Warning: Removed 4 rows containing non-finite values (stat_qq).

## Warning: Removed 4 rows containing non-finite values (stat_qq_line).
## Warning: Removed 4 rows containing missing values (geom_point).

## Warning: Removed 4 rows containing non-finite values (stat_bin).

plots
```



The plots show the leftover residuals which come from the taking the trend and seasonality estimates from the data points. So, we are looking at the random variables given by

$$Z_t = X_t - \widehat{m_t} - \widehat{S_t}.$$

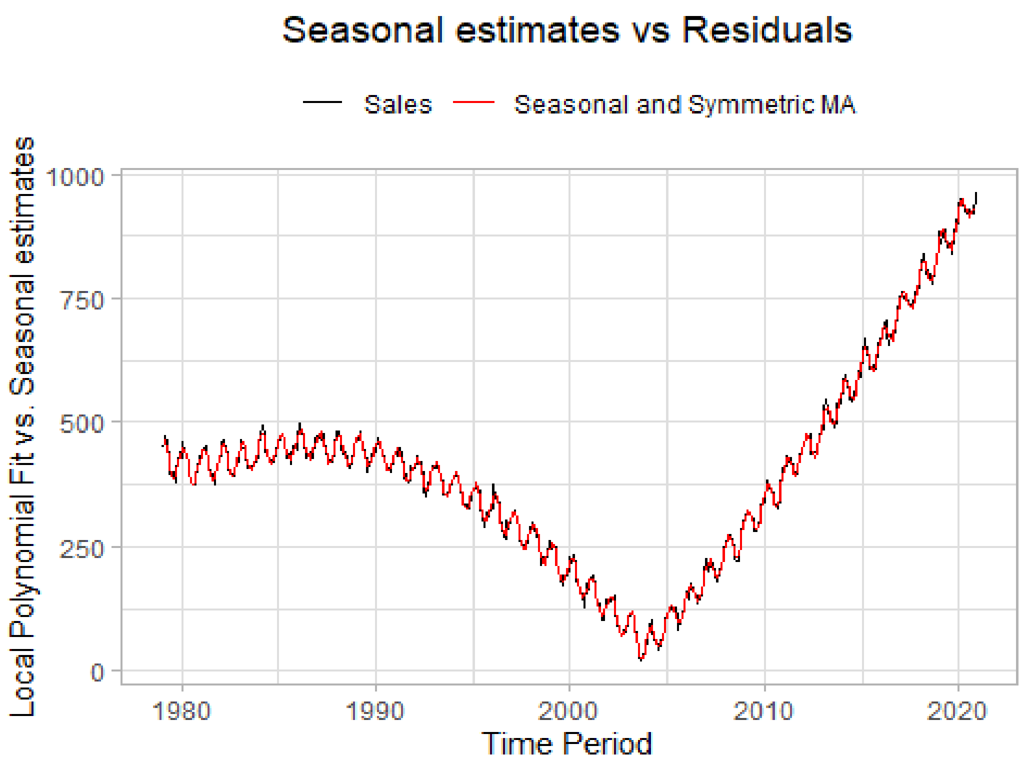
The Residuals vs Fitted Values plot as well as the Standardized residuals vs fitted values show the seasonality element taken away which forms the residuals into the column groups in the graph. The residuals in both plots follow a form of a pattern which means that they are not exactly independent. They depend on the seasonal periods.

From the qq-plot, the Standardized residuals vs fitted values plot and the histogram we can see some potential outlier points. We can see that the sample quantile follows the theoretical quantile line on the qq-plot, apart from a few points in the tails, indicating that overall, the leftover residuals also seem to stem from the normal distribution.

The histogram also shows a small tail on the right side where a point could be an outlier. In addition, the peak around 0 doesn't seem to strictly follow a bell-shaped curve with a peak at 0, hence the leftover residuals might be coming from a normal distribution and further analysis should be done.

In summary, we can plot the local linear fit and seasonal estimation together to see our overall model for the Sales data.

```
df$local_and_seasonal <- df$linear_polynomial + df$seasonal_fit

p <-ggplot(data = df, aes(x = time, y = sales)) +
  geom_line(aes( y= sales, color="Sales")) +  ## plot Sales against time
  geom_line(data = df, aes(x=time, y=local_and_seasonal, color='Seasonal and Symmet
ric MA')) + ## plot fitted values against time
  xlab("Time Period") + ## labeling the x axis
  ylab("Local Polynomial Fit vs. Seasonal estimates") +
  ggtitle("Seasonal estimates vs Residuals") +  ## add title
  theme(plot.title = element_text(size=10, hjust = 'right', face="bold.italic")) +
# main title
  theme_bw() +  ## white background
  theme_light() +  ## light background lines
  theme(legend.position="top") +
  ggeasy::easy_center_title()  + ## centered title
  scale_colour_manual("",
                      breaks = c("Sales", "Seasonal and Symmetric MA"), ## legend n
ames
                      values = c("Black", "red")) ## define legend colors

p

## Warning: Removed 4 row(s) containing missing values (geom_path).
```


Seasonal estimates vs Residuals
Sales
Seasonal and Symmetric MA
Local Polynomial Fit vs. Seasonal estimates
Time Period