Time Series Analysis – Lecture 3
Trends in Time Series Data

Dr. Kevin Burke

1 Trend

We typically wish to work with stationary time series data (described in the previous lecture). The simplest and most common departure from stationarity is due to the presence of **trend**.

Recall that for a stationary series we require $\mu_t = \mu$, i.e., the mean function is constant with time. However, in practice, we often observe trend in the series whereby the **mean function depends on time**.

We may write

$$Y_t = \mu_t + X_t$$

where X_t is a zero-mean stationary series.

If we can estimate the mean function, then we can work with the **detrended series**:

$$\hat{X}_t = Y_t - \hat{\mu}_t$$

where $\hat{\mu}_t$ is the estimated mean function and Y_t is the observed time series, i.e., $\hat{X}_t$ is the series of **residuals**.

1.1 Classical Decomposition

A classical approach is to view the components of a time series as follows:

- Overall trend: long-term change in the mean of the process over time, e.g., series is increasing/decreasing over time.
- Seasonal trend: series repeats itself at regular intervals such as weeks, months, quarters.
- Random noise: this is the unexplained, random behaviour remaining in the series once overall and seasonal trends have been removed.

The **additive** model is:

$$Y_t = o_t + s_t + X_t$$

which is equivalent to modelling the mean function as

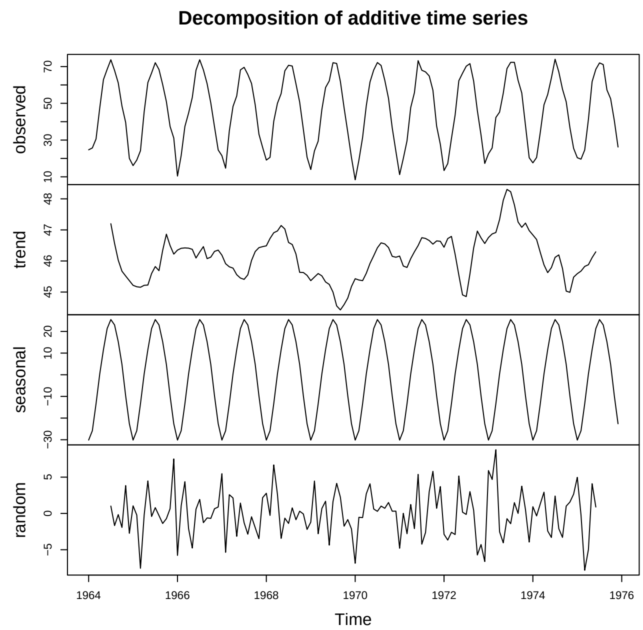
$$\mu_t = o_t + s_t.$$

The **classical decomposition** then aims to decompose the series into three parts:

1. Estimate the overall trend, o_t, in the data by fitting a parametric regression model (e.g., Section 3.1) or a non-parametric smoother (details omitted).
2. Subtract the estimate of o_t to produce the series $Y_t - \hat{o}_t$. For this series, calculate the seasonal means (e.g., the mean for each month say) and recenter by subtracting the mean of the seasonal means. This produces $\hat{s}_t$.
3. Subtract $\hat{o}_t$ and $\hat{s}_t$ from Y_t to produce $\hat{X}_t$.

The above procedure has been implemented in R using the `decompose` function.

Example 1.1. *Decomposed Temperature Data*




```
tempdecom <- decompose(tempdub)
plot(tempdecom)
diff(range(tempdub))
diff(range(tempdecom$trend,na.rm=T))
diff(range(tempdecom$seasonal,na.rm=T))
diff(range(tempdecom$random,na.rm=T))
```

The above shows us that the original data spans about 66 units whereas:

1. Overall: $\approx$ 4 units
2. Seasonal: $\approx$ 56 units
3. Noise: $\approx$ 17 units

Thus, there is little overall trend in the data - it is mainly made up of a seasonal component.

2 Estimating Constant Mean

Before we look at approaches for estimating overall and seasonal trend (other than the classical decomposition), we will first consider the most simple model where the mean function is constant:

$$Y_t = \mu + X_t$$

i.e., there is no trend so that Y_t is stationary.

To estimate μ we simply use the sample mean:

$$\overline{Y} = \tfrac{1}{n}\sum_{t=1}^{n} Y_t$$

which is an unbiased estimator since

$$\begin{aligned} E(\overline{Y}) = E\left(\tfrac{1}{n}\sum_{t=1}^{n} Y_t\right) &= \tfrac{1}{n}\sum_{t=1}^{n} E(Y_t) \\ &= \tfrac{1}{n}\sum_{t=1}^{n} \mu \qquad (\text{since } E[X_t]=0) \\ &= \tfrac{1}{n} n\mu = \mu. \end{aligned}$$

We can also work out the variance of this estimator

$$\begin{aligned} \mathrm{Var}(\overline{Y}) &= \tfrac{1}{n^2}\left[\sum_{t=1}^{n}\mathrm{Var}(Y_t) + 2\sum_{t=2}^{n}\sum_{s=1}^{t-1}\mathrm{Cov}(Y_t, Y_s)\right] \\ &= \tfrac{1}{n^2}\left[\sum_{t=1}^{n}\gamma_{t,t} + 2\sum_{t=2}^{n}\sum_{s=1}^{t-1}\gamma_{t,s}\right] \\ &= \tfrac{1}{n^2}\left[\sum_{t=1}^{n}\gamma_0 + 2\sum_{t=2}^{n}\sum_{s=1}^{t-1}\gamma_{t-s}\right] \\ &\qquad\qquad (\text{since } Y_t \text{ is stationary}) \\ &= \tfrac{1}{n^2}\left[n\gamma_0 + 2\sum_{t=2}^{n}\sum_{s=1}^{t-1}\gamma_{t-s}\right] \end{aligned}$$

It turns out that the above double-sum can be simplified. All we need to do is to write out some terms to see the pattern that emerges:

$$\begin{array}{lcr} & & \displaystyle\sum_{s=1}^{t-1}\gamma_{t-s} \\ t=2: & \Rightarrow & \gamma_1 \\ t=3: & \Rightarrow & \gamma_2+\gamma_1 \\ t=4: & \Rightarrow & \gamma_3+\gamma_2+\gamma_1 \\ & \vdots & \\ t=n-1: & \Rightarrow & \gamma_{n-2}+\cdots+\gamma_2+\gamma_1 \\ t=n: & \Rightarrow & \gamma_{n-1}+\gamma_{n-2}+\cdots+\gamma_2+\gamma_1 \end{array}$$

Therefore the double-sum is

$$\begin{aligned} &\sum_{t=2}^{n}\sum_{s=1}^{t-1}\gamma_{t-s} \\ &= (n-1)\gamma_1 + (n-2)\gamma_2 + \cdots + (1)\gamma_{n-1} \\ &= \sum_{k=1}^{n-1}(n-k)\gamma_k \end{aligned}$$

We can now get an expression for the variance of $\overline{Y}$:

$$\begin{aligned} \mathrm{Var}(\overline{Y}) &= \tfrac{1}{n^2}\left[n\gamma_0 + 2\sum_{k=1}^{n-1}(n-k)\gamma_k\right] \\ &= \tfrac{1}{n^2}\left[n\gamma_0 + 2\tfrac{n\gamma_0}{n\gamma_0}\sum_{k=1}^{n-1}(n-k)\gamma_k\right] \\ &= \tfrac{1}{n^2}\left[n\gamma_0 + 2n\gamma_0\sum_{k=1}^{n-1}(1-\tfrac{k}{n})\tfrac{\gamma_k}{\gamma_0}\right] \\ &= \tfrac{n\gamma_0}{n^2}\left[1 + 2\sum_{k=1}^{n-1}(1-\tfrac{k}{n})\rho_k\right] \\ &= \tfrac{\gamma_0}{n}\left[1 + 2\sum_{k=1}^{n-1}(1-\tfrac{k}{n})\rho_k\right]. \end{aligned}$$

2.1 Effect of Correlation

In the standard situation (in other statistics courses) whereby the observations are independent and identically distributed (i.e., white noise in the language of this course), the variance of the sample mean

$$\mathrm{Var}(\overline{Y}) = \tfrac{\gamma_0}{n}$$

since all of the ρ_k values are zero in this case. This should be familiar from previous courses (where the notation there would have been $\frac{\sigma^2}{n}$).

Now consider a process whereby $\rho_1 = -0.4$ and all other correlations are zero:

$$\begin{aligned} \mathrm{Var}(\overline{Y}) &= \tfrac{\gamma_0}{n}\left[1 + 2(1-\tfrac{1}{n})(-0.4)\right] \\ &= \tfrac{\gamma_0}{n}\left[1 - 0.8(\tfrac{n-1}{n})\right] \\ &\approx \tfrac{\gamma_0}{n}\left[1-0.8\right] \qquad (\text{when } n \text{ is large, } \tfrac{n-1}{n}\approx 1) \\ &= 0.2\tfrac{\gamma_0}{n}. \end{aligned}$$

Thus the estimate of $\hat{\mu}$ is *more precise* (less variance) than the case where the series is white noise. This is due to the negative correlation which causes the series to oscillate around the true mean, μ, so that we need less data to get a precise estimate.

On the other hand if $\rho_1 = 0.4$ then

$$\mathrm{Var}(\overline{Y}) \approx 1.8\tfrac{\gamma_0}{n}.$$

In this case the estimate is less precise due to the fact that values in the series hang together and can get further away from μ.

Note: We cannot have $\rho_1 < -0.5$, for example, $\rho_1 = -0.6$ leads to $\text{Var}(\overline{Y}) \approx -0.2\frac{\gamma_0}{n}$ and, of course, variance cannot be negative. There are clearly constraints on allowable values for the sequence of autocorrelations $\{\rho_k\}$ such that the expression $\frac{\gamma_0}{n}\left[1 + 2\sum_{k=1}^{n-1}(1-\frac{k}{n})\rho_k\right]$ is positive. This is related to the discussion of *non-negative definiteness* mentioned in the previous lecture.

3 Estimating Trend using Regression

Regression analysis can be used to estimate the parameters of common non-constant mean trend models using *ordinary least squares*.

It is worth noting that the **standard errors (and associated p-values) produced by R are incorrect** as these are based on the more typical assumption of independent and identically distributed residuals (i.e., white noise). As we are dealing with time series data with autocorrelation, the residuals will not be independent. We can account for the correlation structure in the data, and produce correct standard errors, using *generalised least squares* - we will not pursue this.

3.1 Overall Trend: o_t component

Here we consider models for the overall trend component, o_t of the mean function.

Linear Trend

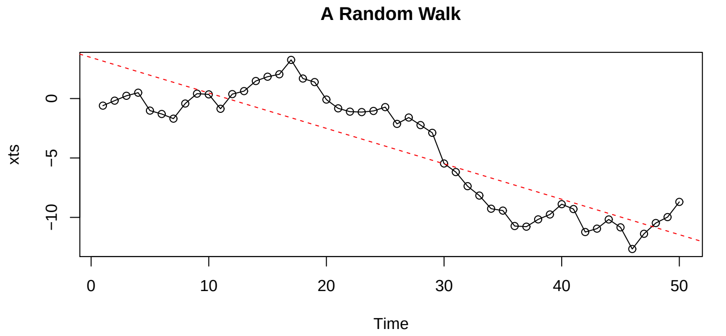
In this case we assume that

$$o_t = \beta_0 + \beta_1 t.$$

We must be careful about assuming trends models which **imply that trends continue beyond the time window that we have observed.** The following example makes this clear.

Example 3.1. *Random Walk*
We can simulate, and plot, a random walk as follows:

```
set.seed(3112832)
n <- 50
x <- rnorm(n)
xts <- ts(cumsum(x))
dev.new(width=8, height=4)
plot(xts, main="A Random Walk", type="o")
```



This appears to exhibit downward trend. We have superimposed a linear trend model on the above time series plot which appears to describe the time series. However, we have simulated this series and know it is simply a random walk with no real trend. The perceived trend is only due to positive correlation between neighbouring values (derived in the previous lecture).

```
ti <- as.vector(time(xts))
xreg <- lm(xts~ti)
lines(x=ti, y=predict(xreg), lty=2, col=2)
summary(xreg)
```

R output:

```
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.45178    0.68357    5.05 6.81e-06 ***
ti          -0.29813    0.02333  -12.78  < 2e-16 ***
---
Residual standard error: 2.381 on 48 degrees of freedom
Multiple R-squared: 0.7728,     Adjusted R-squared: 0.7681
```

Quadratic Trend

A quadratic trend is given by:

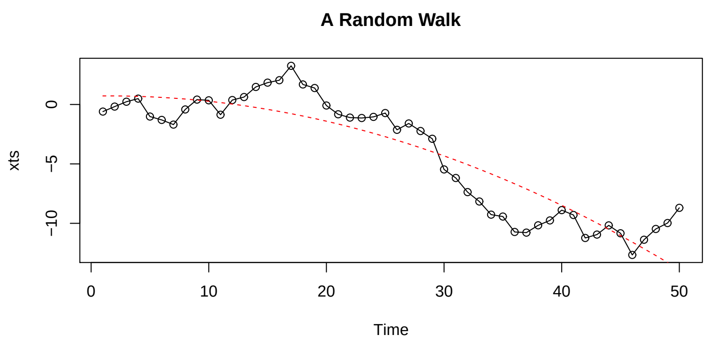
$$o_t = \beta_0 + \beta_1 t + \beta_2 t^2.$$

Example 3.2. *Random Walk*
We can fit a quadratic trend model as follows:

```
xreg <- lm(xts~ ti + I(ti^2))
summary(xreg)
```

R output:

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.708134   0.923344   0.767 0.446963
ti           0.018445   0.083522   0.221 0.826177
I(ti^2)     -0.006207   0.001588  -3.910 0.000296 ***
---
Residual standard error: 2.09 on 47 degrees of freedom
Multiple R-squared: 0.8286,     Adjusted R-squared: 0.8213
```



This fit seems to be even better - but, of course, we know there is not real trend in the data. However, in practice we will not know whether or not the trend is real (and continues forever) or a consequence of autocorrelation in the series.

Polynomial Trend

More general polynomial trend is given by

$$o_t = \beta_0 + \beta_1 t + \beta_2 t^2 + \cdots + \beta_q t^q,$$

and can be estimated using R, for example, a 4th order polynomial (fitted to the random walk above) is

```
xreg <- lm(xts~ ti + I(ti^2) + I(ti^3) + I(ti^4))
```

3.2 Seasonal Trend: s_t component

Here we consider models for the overall trend component, s_t of the mean function.

Seasonal Effects

A seasonal means model gives each month a different mean as follows:

$$s_t = \begin{cases} \beta_1 & \text{for } t = 1,\ 13,\ 25, \ldots \\ \beta_2 & \text{for } t = 2,\ 14,\ 26, \ldots \\ \beta_3 & \text{for } t = 3,\ 15,\ 27, \ldots \\ \vdots & \\ \beta_{12} & \text{for } t = 12,\ 24,\ 36, \ldots \end{cases}$$

Note: the assumption has been made here that the we have monthly data.

Here each regression coefficient represents the seasonal effect for that month (being the average value that month).

An equivalent model is:

$$s_t = \beta_0 + \begin{cases} 0 & \text{for } t = 1,\ 13,\ 25, \ldots \\ \beta_1 & \text{for } t = 2,\ 14,\ 26, \ldots \\ \beta_2 & \text{for } t = 3,\ 15,\ 27, \ldots \\ \vdots & \\ \beta_{11} & \text{for } t = 12,\ 24,\ 36, \ldots \end{cases}$$

In this formulation of the model, β_0 is the January effect and the other coefficients measure the effect of each month relative to January.

The second version of the model is more standard as it is **good practice to fit regression models with intercepts**, β_0.

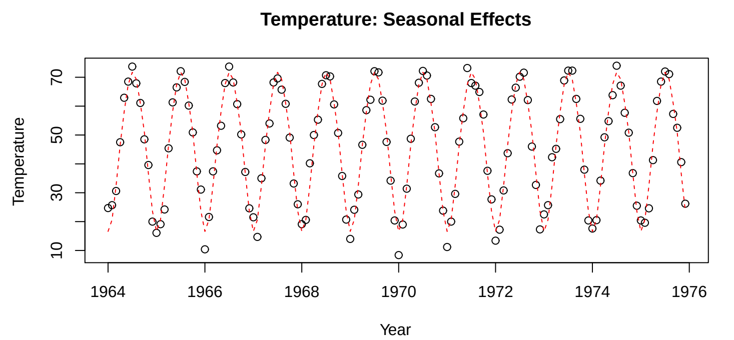
Note: the R^2 value outputted by software is invalid for models with no intercept.

Example 3.3. *Temperature Data*
Code for the second version of the seasonal model:

```
months <- season(tempdub)
modeltemp <- lm(tempdub~ months)
summary(modeltemp)
plot(tempdub, xlab="Year", ylab="Temperature", type="p")
lines(as.vector(time(tempdub)), predict(modeltemp),
       lty=2, col=2)
```

R output:

```
                Estimate Std. Error t value Pr(>|t|)
(Intercept)       16.608      0.987  16.828  < 2e-16 ***
monthsFebruary     4.042      1.396   2.896  0.00443 **
monthsMarch       15.867      1.396  11.368  < 2e-16 ***
monthsApril       29.917      1.396  21.434  < 2e-16 ***
monthsMay         41.483      1.396  29.721  < 2e-16 ***
monthsJune        50.892      1.396  36.461  < 2e-16 ***
monthsJuly        55.108      1.396  39.482  < 2e-16 ***
monthsAugust      52.725      1.396  37.775  < 2e-16 ***
monthsSeptember   44.417      1.396  31.822  < 2e-16 ***
monthsOctober     34.367      1.396  24.622  < 2e-16 ***
monthsNovember    20.042      1.396  14.359  < 2e-16 ***
monthsDecember     7.033      1.396   5.039 1.51e-06 ***
---
Residual standard error: 3.419 on 132 degrees of freedom
Multiple R-squared: 0.9712,     Adjusted R-squared: 0.9688
```



The above plot shows the fitted model (dashed line) with the actual data (points). We can see that the seasonal pattern appears to be captured quite well.

Harmonics

Harmonic terms (sin and cos) can also be used to describe regular seasonal patterns using fewer parameters than the seasonal means model above. If this model is appropriate for the data, then predictions can be more efficient (less variance) than the seasonal means model.

Here the seasonal component is given by

$$s_t = \beta_0 + \beta_1 \cos(2\pi f t) + \beta_2 \sin(2\pi f t)$$

where f is the frequency which we set in advance. This model describes patterns which repeat every $\frac{1}{f}$ time units. Note that this only has three parameters compared to the seasonal means model which has twelve.

For monthly data where time is stored as

- $t = 1, 2, 3, 4, \ldots$ we would set $f = \frac{1}{12}$ so that the pattern repeats every $\frac{1}{f} = 12$ time units (i.e., 12 months).
- $t = 2016.000,\ 2016.083,\ 2016.167,\ 2016.250, \ldots$ we would set $f = 1$ so that the pattern repeats every $\frac{1}{f} = 1$ time unit (i.e., 1 year).

More flexible harmonic models can be generated by having more terms:

$$s_t = \beta_0 + \sum_{i=1}^{p} [\beta_{2i-1} \cos(2\pi f_i t) + \beta_{2i} \sin(2\pi f_i t)]$$

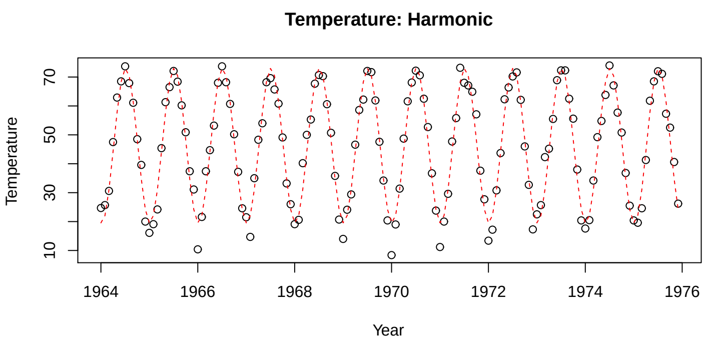
where $(f_1, \ldots, f_p)$ is a vector of pre-specified frequencies.

Example 3.4. *Temperature Data*
Below is the code for the second version of the seasonal model:

```
hterms <- harmonic(tempdub, m=1)
modeltemp2 <- lm(tempdub~ hterms)
summary(modeltemp2)
plot(tempdub, xlab="Year", ylab="Temperature", type="p")
lines(as.vector(time(tempdub)), predict(modeltemp2),
      lty=2, col=2)
```

R output:

```
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          46.2660     0.3088 149.816  < 2e-16 ***
htermscos(2*pi*t) -26.7079     0.4367 -61.154  < 2e-16 ***
htermssin(2*pi*t)  -2.1697     0.4367  -4.968 1.93e-06 ***
---
Residual standard error: 3.706 on 141 degrees of freedom
Multiple R-squared: 0.9639,     Adjusted R-squared: 0.9634
```



The fit from the harmonic model is clearly very similar to that of the seasonal means model.

3.3 Mean Function: $\mu_t = o_t + s_t$

Above we have addressed the o_t and s_t components of the mean function, $\mu_t = o_t + s_t$. Clearly then, in general we can have mean functions such as:

$$\mu_t = \beta_0 + \beta_1 t + \beta_2 \cos(2\pi f t) + \beta_3 \sin(2\pi f t)$$

i.e., linear overall trend and harmonic seasonal trend. This can be fitted in R using:

`lm( x ~ time(x) + harmonic(x,m=1) )`

where `x` here is a time series object.

Similarly, we could have

$$\mu_t = \beta_0 + \beta_1 t + \beta_2 t^2 + \begin{cases} 0 & \text{for } t = 1,\ 13,\ 25, \ldots \\ \beta_3 & \text{for } t = 2,\ 14,\ 26, \ldots \\ \beta_4 & \text{for } t = 3,\ 15,\ 27, \ldots \\ \vdots & \\ \beta_{13} & \text{for } t = 12,\ 24,\ 36, \ldots \end{cases}$$

i.e., quadratic overall trend and individual seasonal effects. This can be fitted in R using:

`lm( x ~ time(x) + I(time(x)^2) + season(x) )`

3.4 Note on Differencing

In Sections 3.1 - 3.3, we looked at methods for *modelling* trend in a time series. However, as noted in Section 3.1, we should be careful about assuming a trend model which implies such trend continues indefinitely.

An alternative approach is to *eliminate* trend rather than trying to develop a model for it - this can be achieved through differencing, e.g.,

- differencing once eliminates linear trend,
- differencing twice eliminates quadratic trend,
- seasonal differencing eliminates seasonal trend.

Typically high order differencing (i.e., beyond differencing once or twice) is not required in practice.

Important: If we simply want to convert a process into a stationary process for further analysis (without assuming a trend model), then differencing is more appropriate - and is a core part of this course.

4 Detrended Series (Residuals)

Once we have estimated the trend in the data via $\hat{\mu}_t$, we can then **detrend the data by subtracting $\hat{\mu}_t$**:

$$Y_t = \mu_t + X_t$$

$$\Rightarrow \hat{X}_t = Y_t - \hat{\mu}_t$$

where the trend model is estimated using the methods of Section 3 for example.

The $\hat{X}_t$ values are the **residuals** which we can then investigate. A number of standard things can be done with the residuals such as investigating:

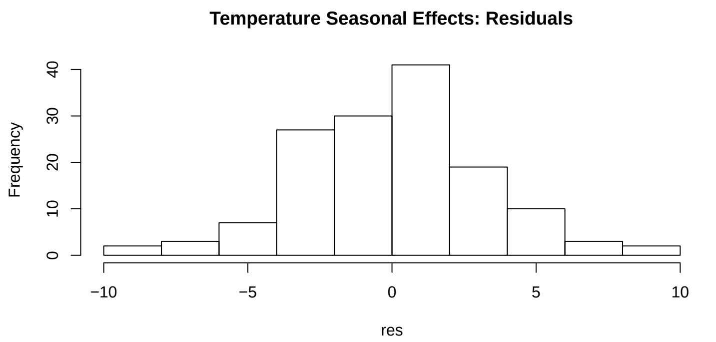
- Normality using a histogram, QQ-plot or Shapiro-Wilk test.
- Presence of non-constant variance by plotting them against predicted values (i.e., $\hat{\mu}_t$).

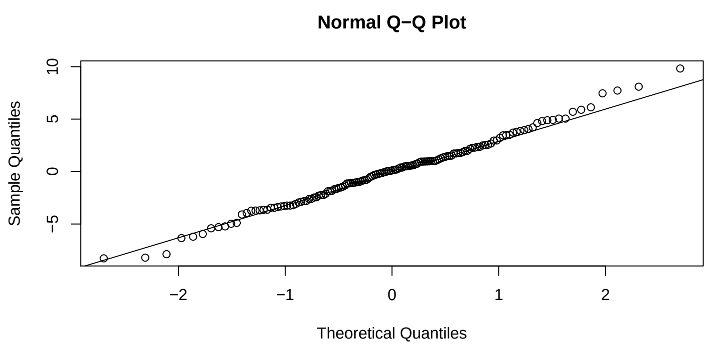
However, more specific to time series analysis, we will be interested in the:

- Correlation structure of $\hat{X}_t$ via its sample auto-correlation function.
- We can also carry out a runs test of independence.

Example 4.1. *Temperature Data*
Following from Example 3.3, we will now investigate the residuals, i.e., the detrended series as seasonal trend has been removed.

```
res <- residuals(modeltemp)
dev.new(width=8, height=4)
hist(res, main="Temperature Seasonal Effects: Residuals")
qqnorm(res); qqline(res)
shapiro.test(res)
```

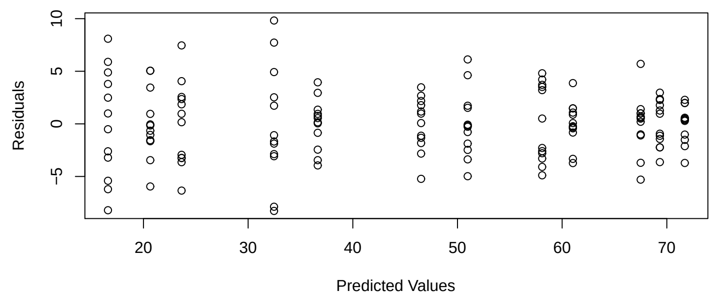





```
        Shapiro-Wilk normality test
data:  res
W = 0.99371, p-value = 0.7837
```

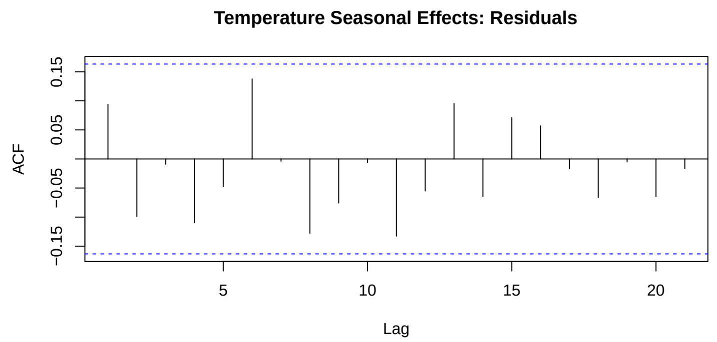
From the above histogram and QQ-plot we can see that the residuals are approximately normally distributed. This is further confirmed by the Shapiro-Wilk test as we accept the null hypothesis that the data is normal.

```
pred <- predict(modeltemp)
plot(pred,res, xlab="Predicted Values", ylab="Residuals")
```



Plotting predicted values against residuals shows that the variation is relatively constant (although slightly higher variation for smaller predicted values).

```
acf(res,main="Temperature Seasonal Effects: Residuals")
```



There is no significant autocorrelation in the detrended series (note: compare this to the acf for the original series).

Applying the runs test to the residuals yields:

```
runs(res)
```

```
 pvalue observed.runs expected.runs
  0.216         65.000        72.875
```

Thus we accept the null hypothesis that the residuals are independent (i.e., no significant autocorrelation). This is in line with the findings of the ACF.