Time Series Decomposition

Lesson 3.1

Time series components

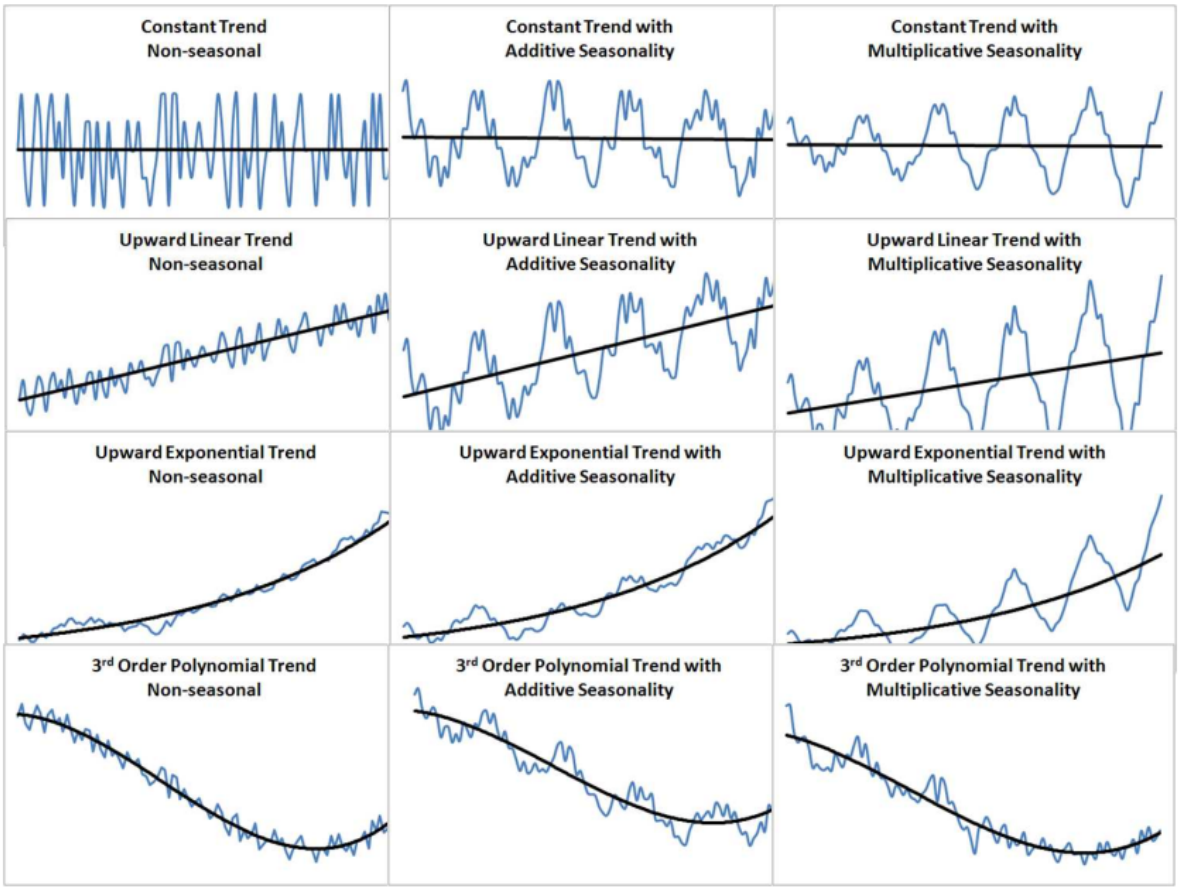
A time series consists of two general components: **systematic** and **non-systematic**. The systematic part is typically divided into three components: **trend**, **seasonality**, and **cyclicity**. While, the nonsystematic part is called **noise**.

Trend is the change in the series from one period to the next. *Seasonality* describes a short-term cyclical behavior that can be observed several times within the given series. Lastly, *noise* is the random variation that results from measurement error or other causes that are not accounted for. It is always present in a time series to some degree, although we cannot observe it directly.

The different components are commonly considered to be either *additive* or *multiplicative*. A time series with additive components can be written as: $Y_t = T_t + S_t + R_t$ and a time series with multiplicative components can be written as: $Y_t = T_t \times S_t \times R_t$.

Forecasting methods attempt to isolate the systematic part and quantify the noise level. The systematic part is used for generating point forecasts and the level of noise helps assess the uncertainty associated with the point forecasts.

Trend patterns are commonly approximated by linear, exponential and other mathematical functions. Meanwhile, seasonal patterns can be approximated as either *additive seasonality* where values in different seasons vary by a constant amount, or *multiplicative seasonality* where values in different seasons vary by a percentage



Basic steps in decomposition

1. Estimate the trend, say using moving averages
2. "De-trend" the series
    - For an additive decomposition, this is done by subtracting the trend estimates from the series.
    - For a multiplicative decomposition, this is done by dividing the series by the trend values.
3. Estimate seasonal factors using the de-trended series
    - For monthly data, this entails estimating an effect for each month of the year
    - For quarterly data, this entails estimating an effect for each quarter

- The simplest method for estimating these effects is to average the de-trended values for a specific season. For instance, to get a seasonal effect for January, we average the de-trended values for all Januarys in the series, and so on

4. Determine the random component

   - For the additive model, $R_t = Y_t - T_t - S_t$
   - For the multiplicative model, $R_t = \frac{Y_t}{T_t \times S_t}$

Classical decomposition

For the purpose of choosing adequate forecasting methods, it is useful to dissect a time series into its constituent components. This process is called **Decomposition** of a time series.

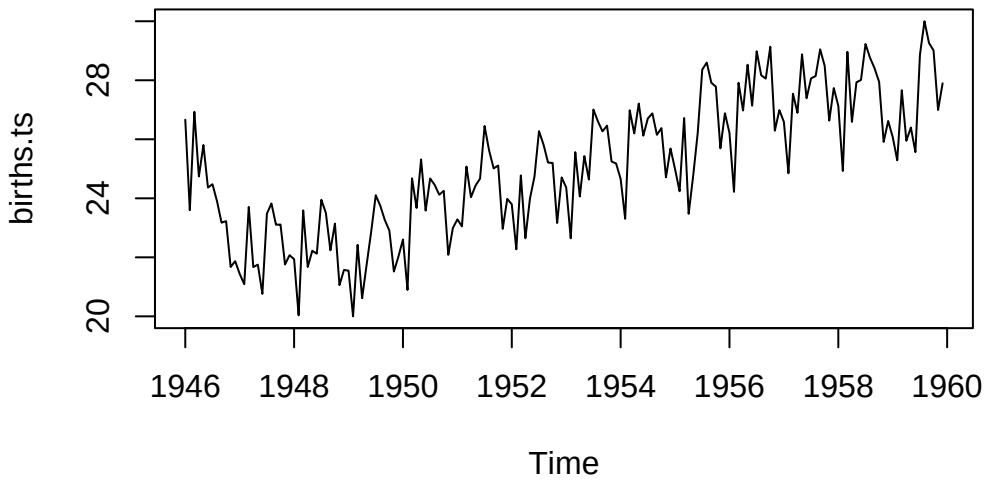
There are two forms of classical decomposition: an **additive** decomposition and a **multiplicative** decomposition. In classical decomposition, we assume that the seasonal component is constant from year to year. For multiplicative seasonality, the m values that form the seasonal component are sometimes called the *seasonal indices*.

The basic command in R for (classical) decomposition is *decompose()*. For an additive model, the code is: **decompose(name of series, type = "additive")**.

The code for a multiplicative decomposition is: **decompose(name of series, type ="multiplicative")**

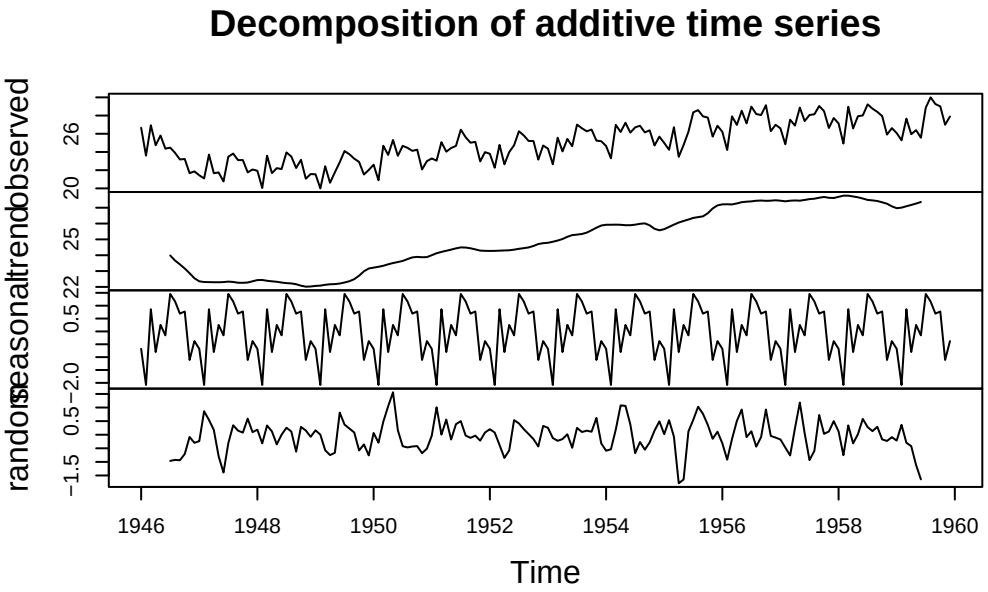
As an illustration, let us consider the dataset of number of births per month in New York city, from January 1946 to December 1959

```
births <- scan("http://robjhyndman.com/tsdldata/data/nybirths.dat")
births.ts <- ts(births,
                frequency = 12,
                start = c(1946, 1))
plot(births.ts)
```



We can see some seasonal variation in the number of births per month: there is a peak every summer, and a trough every winter. Seasonal fluctuations are roughly constant in size over time and do not seem to depend on the level of the time series. Hence, an additive model can be applied.

```
births.comp <- decompose(births.ts,type="additive")
plot(births.comp)
```



Then we can extract the component values from the output of the *decompose()* function.

```
trend <- births.comp$trend
seasonality <- births.comp$seasonal
random <- births.comp$random
seasonal.indices <- births.comp$figure
print(trend)
```

```
          Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
1946       NA       NA       NA       NA       NA       NA 23.98433 23.66213
1947 22.35350 22.30871 22.30258 22.29479 22.29354 22.30562 22.33483 22.31167
1948 22.43038 22.43667 22.38721 22.35242 22.32458 22.27458 22.23754 22.21988
1949 22.06375 22.08033 22.13317 22.16604 22.17542 22.21342 22.27625 22.35750
1950 23.21663 23.26967 23.33492 23.42679 23.50638 23.57017 23.63888 23.75713
1951 24.00083 24.12350 24.20917 24.28208 24.35450 24.43242 24.49496 24.48379
1952 24.27204 24.27300 24.28942 24.30129 24.31325 24.35175 24.40558 24.44475
1953 24.78646 24.84992 24.92692 25.02362 25.16308 25.26963 25.30154 25.34125
1954 25.92446 25.92317 25.92967 25.92137 25.89567 25.89458 25.92963 25.98246
1955 25.64612 25.78679 25.93192 26.06388 26.16329 26.25388 26.35471 26.40496
1956 27.21104 27.21900 27.20700 27.26925 27.35050 27.37983 27.39975 27.44150
1957 27.44221 27.40283 27.44300 27.45717 27.44429 27.48975 27.54354 27.56933
1958 27.68642 27.76067 27.75963 27.71037 27.65783 27.58125 27.49075 27.46183
1959 26.96858 27.00512 27.09250 27.17263 27.26208 27.36033       NA       NA
```

```
          Sep      Oct      Nov      Dec
1946 23.42333 23.16112 22.86425 22.54521
1947 22.26279 22.25796 22.27767 22.35400
1948 22.16983 22.07721 22.01396 22.02604
1949 22.48862 22.70992 22.98563 23.16346
1950 23.86354 23.89533 23.87342 23.88150
1951 24.43879 24.36829 24.29192 24.27642
1952 24.49325 24.58517 24.70429 24.76017
1953 25.42779 25.57588 25.73904 25.87513
1954 26.01054 25.88617 25.67087 25.57312
1955 26.45379 26.64933 26.95183 27.14683
1956 27.45229 27.43354 27.44488 27.46996
1957 27.63167 27.67804 27.62579 27.61212
1958 27.42262 27.34175 27.25129 27.08558
1959       NA       NA       NA       NA
```

```
print(seasonality)
```

```
            Jan        Feb        Mar        Apr        May        Jun
1946 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1947 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1948 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1949 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1950 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1951 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1952 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1953 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1954 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1955 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1956 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1957 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1958 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
1959 -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
            Jul        Aug        Sep        Oct        Nov        Dec
1946  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1947  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1948  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1949  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1950  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1951  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1952  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1953  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
```

```
1954  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1955  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1956  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1957  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1958  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
1959  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
```

```
print(random)
```

```
              Jan          Feb          Mar          Apr          May
1946           NA           NA           NA           NA           NA
1947 -0.237305288  0.863252404  0.543893429  0.175887019 -0.793193109
1948  0.183819712 -0.318705929  0.340268429  0.121262019 -0.354234776
1949  0.161444712  0.002627404 -0.571689904 -0.749362981 -0.666068109
1950  0.064569712 -0.292705929  0.479560096  1.047887019  1.561973558
1951 -0.036638622  1.008460737  0.004310096  0.556595353 -0.176151442
1952  0.203153045  0.079960737 -0.376939904 -0.853612981 -0.576901442
1953  0.254736378 -0.122955929 -0.224439904 -0.159946314  0.016265224
1954 -0.590263622 -0.536205929  0.189810096  1.079303686  1.062681891
1955  0.021069712  0.535169071 -0.073439904 -1.787196314 -1.647943109
1956 -0.316846955 -0.918039263 -0.155523237  0.507428686  0.924848558
1957 -0.176013622 -0.471872596 -0.762523237  0.240512019  1.182056891
1958  0.122778045 -0.753705929  0.340851763 -0.319696314  0.021515224
1959 -0.215388622  0.363835737 -0.295023237 -0.419946314 -1.115734776
              Jun          Jul          Aug          Sep          Oct
1946           NA -0.963379006 -0.925718750 -0.939949519 -0.709369391
1947 -1.391369391 -0.311879006  0.347739583  0.150592147  0.076797276
1948  0.001672276  0.256412660  0.119531250 -0.623449519  0.289547276
1949  0.813838942  0.371704327  0.225906250  0.081758814 -0.578161058
1950  0.166088942 -0.423920673 -0.467718750 -0.433157853 -0.418577724
1951  0.387838942  0.499995994 -0.030385417 -0.116407853 -0.033536058
1952  0.538505609  0.414370994  0.206656250  0.025133814 -0.161411058
1953 -0.481369391  0.251412660  0.100156250  0.148592147  0.110880609
1954  0.380672276 -0.679670673 -0.269052083 -0.550157853 -0.282411058
1955  0.118380609  0.550245994  1.029447917  0.768592147  0.359422276
1956 -0.087577724  0.126204327 -0.437093750 -0.087907853  0.927213942
1957  0.053505609 -0.934587340 -0.592927083  0.724717147  0.030713942
1958  0.581005609  0.282204327  0.132572917  0.290758814 -0.171994391
1959 -1.642077724           NA           NA           NA           NA
              Nov          Dec
1946 -0.082484776 -0.298388622
1947  0.591098558  0.095819712
```

```
1948  0.154806891 -0.076221955
1949 -0.356859776 -0.761638622
1950 -0.679651442 -0.513680288
1951 -0.218151442  0.081403045
1952 -0.432526442  0.323653045
1953  0.616723558 -0.318305288
1954  0.150890224  0.491694712
1955 -0.149068109  0.110986378
1956 -0.044109776 -0.106138622
1957  0.117973558  0.499694712
1958 -0.229526442 -0.089763622
1959           NA           NA
```

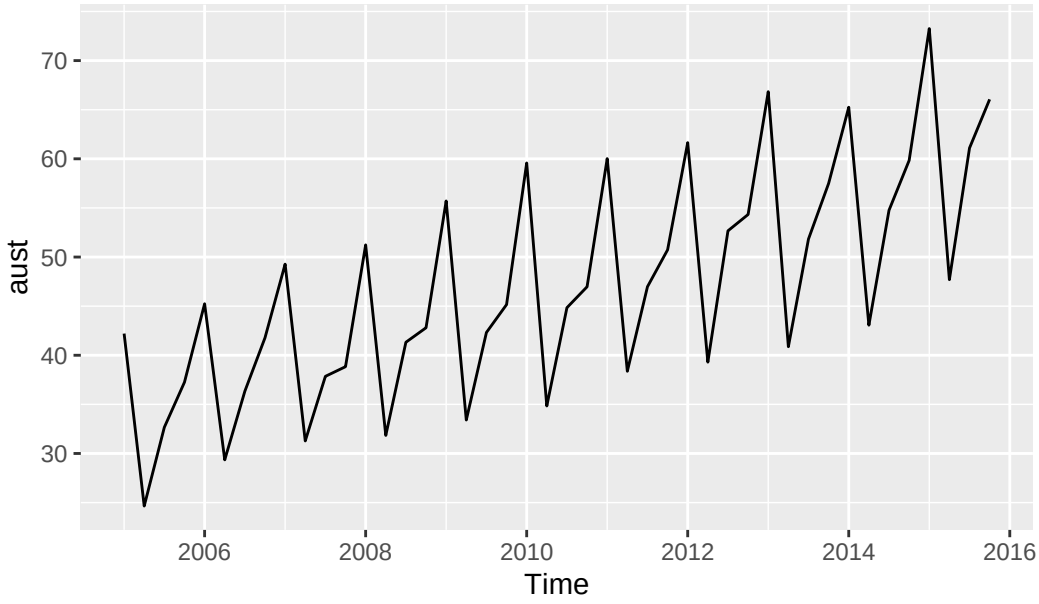
```r
print(seasonal.indices)
```

```
 [1] -0.6771947 -2.0829607  0.8625232 -0.8016787  0.2516514 -0.1532556
 [7]  1.4560457  1.1645938  0.6916162  0.7752444 -1.1097652 -0.3768197
```

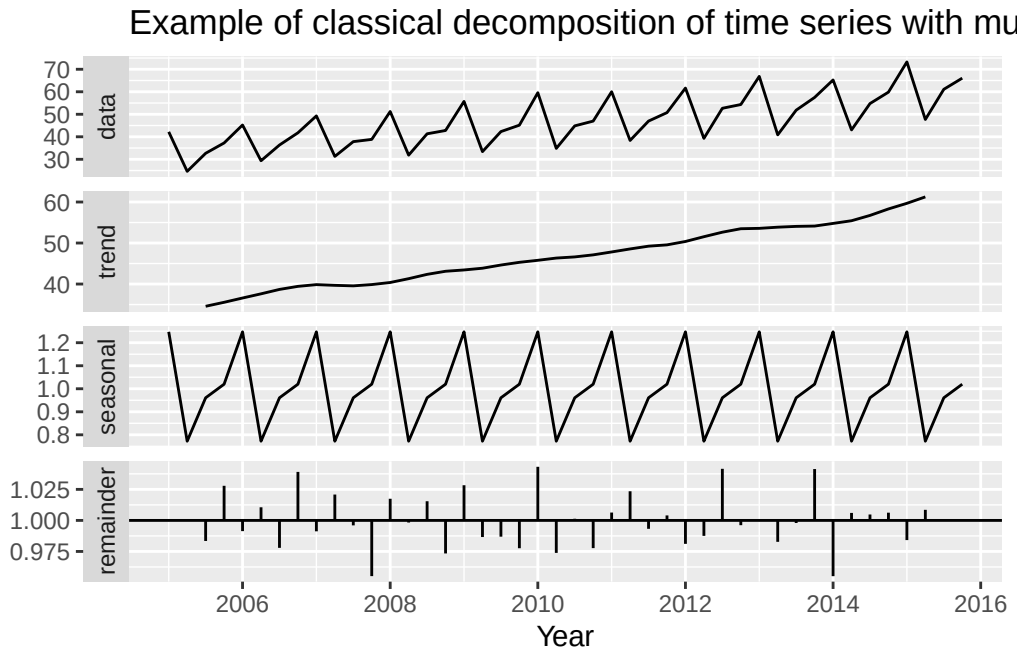
```r
sum(seasonal.indices)
```

```
[1] 1.665335e-16
```

```r
library(fpp2)
aust <- window(austourists,start=2005)
autoplot(aust)
```




```
aust.comp <- decompose(aust,type = "multiplicative")
autoplot(aust.comp) + xlab("Year") +
  ggtitle("Example of classical decomposition of time series with multipicative seasonality")
```



```
print(aust.comp$figure)
```

```
[1] 1.2472720 0.7721434 0.9607437 1.0198409
```

```
sum(aust.comp$figure)
```

```
[1] 4
```

Remarks

1. In class decomposition, the estimate of the trend-cycle is unavailable for the first few and last few observations. For example, if $m = 12$, there is no trend-cycle estimate for the first 6 or the last 6 observations. Consequently, there is also no estimate of the remainder component for the same time periods.
2. The trend-cycle estimate tends to over-smooth rapid rises and falls in the data.
3. Classical decomposition methods assume that the seasonal component repeats from year to year. Hence, they are unable to capture seasonal changes over time.
4. The classical method is not robust to small sudden shifts in a time series.

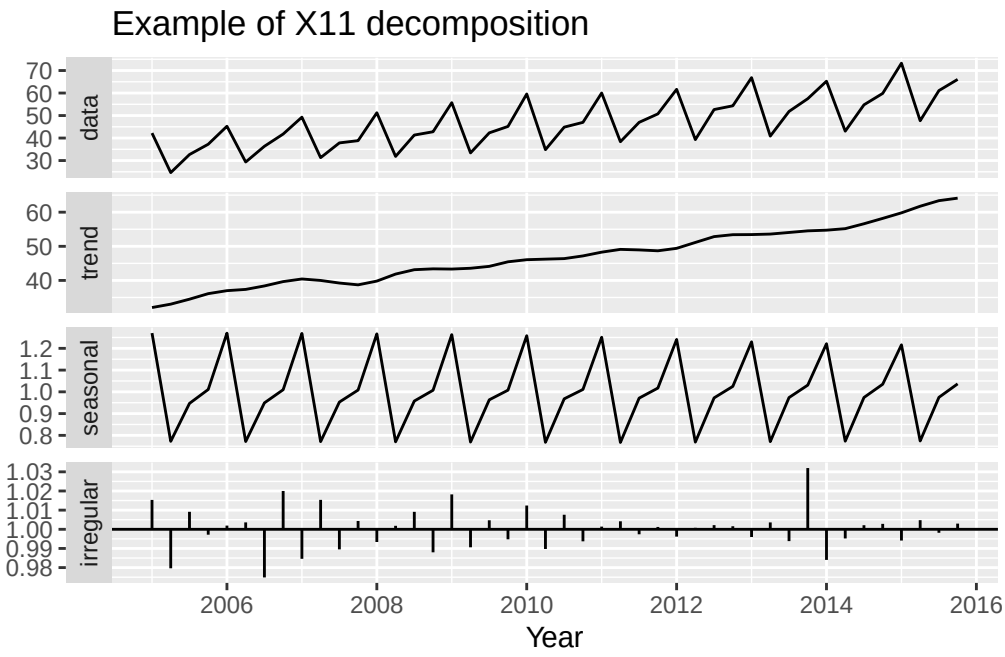
X11 decomposition

Another popular method for decomposing quarterly and monthly data is the X11 method which originated in the US Census Bureau and Statistics Canada. It involves many extra steps and features in order to overcome the drawbacks of classical decomposition. Consequently, trend-cycle estimates are available for all time periods and the seasonal component is allowed to vary slowly over time. Finally, it is highly robust to outliers and level shifts in the time series.

It has sophisticated methods for handling trading day variation, holiday effects and the effects of known predictors. It can also handle both additive and multiplicative decomposition.

The X11 method is available using the *seas()* function from the **seasonal** package

```
library(seasonal)
aust.comp.x11 <- seas(aust, x11 = "")
autoplot(aust.comp.x11) + xlab("Year") +
  ggtitle("Example of X11 decomposition")
```

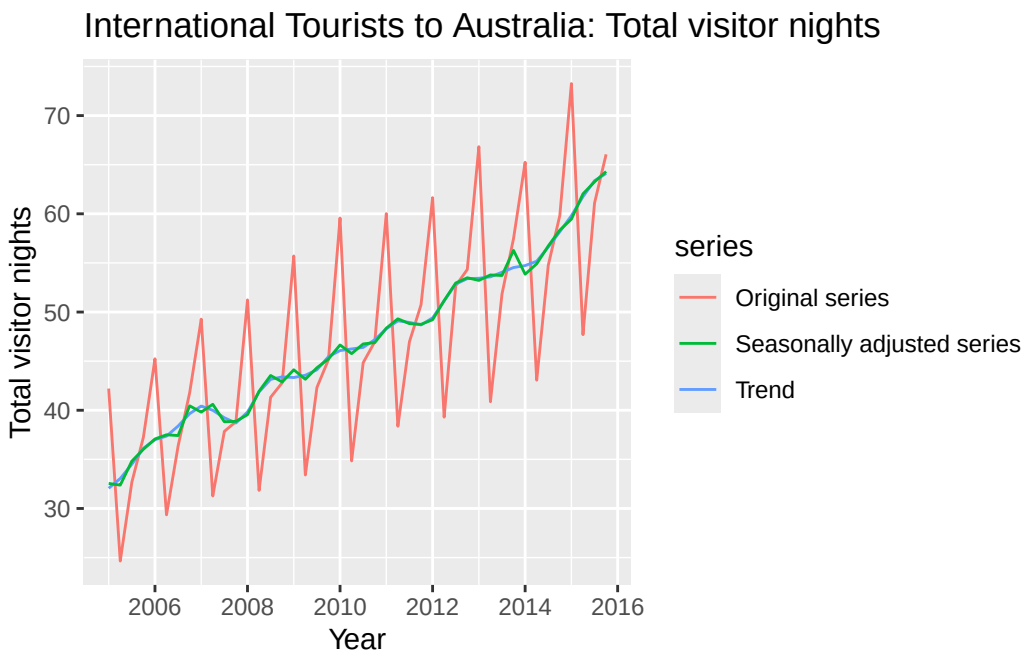


Given the output from the *seas()* function, we can use

- *seasonal()* to extract the seasonal component
- *trendcycle()* to extract the trend-cycle component
- *remainder()* to extract the remainder component
- *seasadj()* to compute the seasonally adjusted time series

The following code chunk will plot the original series together with the trend component and the seasonally adjusted series

```
autoplot(aust, series="Original series") +
  autolayer(trendcycle(aust.comp.x11), series="Trend") +
  autolayer(seasadj(aust.comp.x11), series="Seasonally adjusted series") +
  xlab("Year") +
  ylab("Total visitor nights") +
  ggtitle("International Tourists to Australia: Total visitor nights")
```



SEATS decomposition

Another method of dexcomposing a time series is SEATS. It stands for "Seasonal Extraction in ARIMA Time Series". Unfortunately, it only works with quarterly and monthly data.

It uses the same functions to extract the components and the seasonally adjusted series

```
aust.comp.seats <- seas(aust)
autoplot(aust.comp.seats) +
  xlab("Year") +
  ggtitle("Example of SEATS decomposition")
```

STL decomposition

STL ("Seasonal and Trend decomposition using Loess") is a versatile and robust method for decomposing time series. Loess is a method for estimating nonlinear relationships.

STL has several advantages over the classical, SEATS and X11 decomposition methods.

1. Unlike SEATS and X11, STL will handle any type of seasonality, not only monthly and quarterly data.

2. The seasonal component is allowed to change over time, and the rate of change can be controlled by the user.
3. The smoothness of the trend-cycle can also be controlled by the user.
4. It can be robust to outliers (occasional unusual observations will not affect the estimates of the trend-cycle and seasonal components but can affect the remainder component).
5. It can not handle trading day or calendar variation automatically, and it only provides facilities for additive decomposition.
6. It is possible to obtain a multiplicative decomposition by first taking logarithms of the data, then back-transforming the components

Decompositions between additive and multiplicative can be obtained using a Box-Cox transformation of the data with $0 < \lambda < 1$

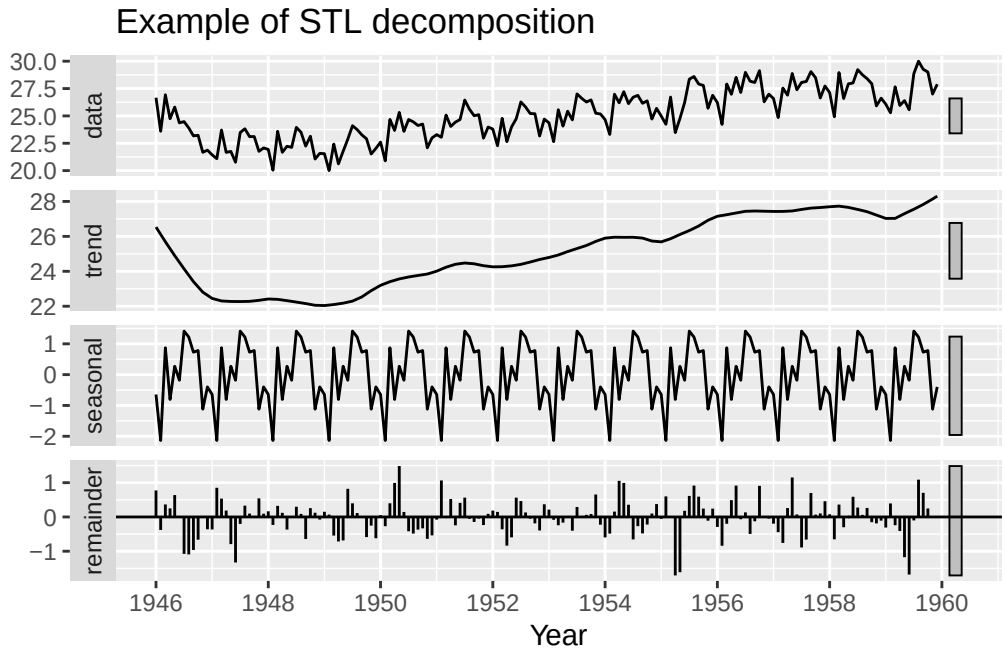
- A value of $\lambda = 0$ corresponds to the multiplicative decomposition
- A value of $\lambda = 1$ corresponds to the additive decomposition

The two main parameters to be chosen when using STL are the trend-cycle window (*t.window*) and the seasonal window (*s.window*). The *t.window* is the number of consecutive observations to be used when estimating the trend-cycle and *s.window* is the number of consecutive years to be used in estimating each value in the seasonal component.

These parameters control how rapidly the trend-cycle and seasonal components can change. Smaller values allow for more rapid changes. Both parameters must be odd numbers. The user must specify *s.window* as there is no default but specifying *t.window* is optional, and a default value will be used if it is omitted.

To extract the time series components after application of STL decomposition we use the same functions as with X11 and SEATS methods.

```
births.stl <- stl(births.ts, s.window="periodic")
autoplot(births.stl) +
  xlab("Year") +
  ggtitle("Example of STL decomposition")
```


Example of STL decomposition
data
trend
seasonal
remainder
30.0
27.5
25.0
22.5
20.0
28
26
24
22
1
0
−1
−2
1
0
−1
1946
1948
1950
1952
1954
1956
1958
1960
Year