03 Time series with trend and seasonality components

Part II

Andrius Buteikis, andrius.buteikis@mif.vu.lt
http://web.vu.lt/mif/a.buteikis/

Seasonal ARMA models

The seasonal ARIMA model incorporates both non-seasonal and seasonal factors in a multiplicative model: $SARIMA(p, d, q)(P, D, Q)_S$.

For now, we will restrict our analysis to non-differenced data $SARMA$ models (i.e. $d = 0$ and $D = 0$), where p, q are the $ARMA$ orders of the non-seasonal components and P, Q are the $ARMA$ orders of the seasonal components.

For example, our series could be described as a seasonal (e.g. quarterly) process:

$$Y_t = \Phi Y_{t-1} + w_t + \Theta w_{t-4}$$

while our shocks w_t could also be a non-seasonal MA process:

$$w_t = \epsilon_t + \theta \epsilon_{t-1}$$

So, while the seasonal term is **additive**, the combined model is **multiplicative**:

$$\begin{aligned} Y_t &= \Phi Y_{t-1} + w_t + \Theta w_{t-4} \\ &= \Phi Y_{t-1} + \epsilon_t + \theta \epsilon_{t-1} + \Theta \epsilon_{t-4} + \theta \Theta \epsilon_{t-5} \end{aligned}$$

We can write the general model formally as:

$$\Phi(L^S)\phi(L)(Y_t - \mu) = \Theta(L^S)\theta(L)\epsilon_t$$

where $\phi(z) = 0, \forall |z_i| > 1$ and $\Phi(z) = 0, \forall |z_j| > 1$, and:

- The non-seasonal components are:

$$\text{AR: } \phi(L) = 1 - \phi_1 L - ... - \phi_p L^p$$
$$\text{MA: } \theta(L) = 1 + \theta_1 L + ... + \theta_q L^q$$

- The seasonal components are:

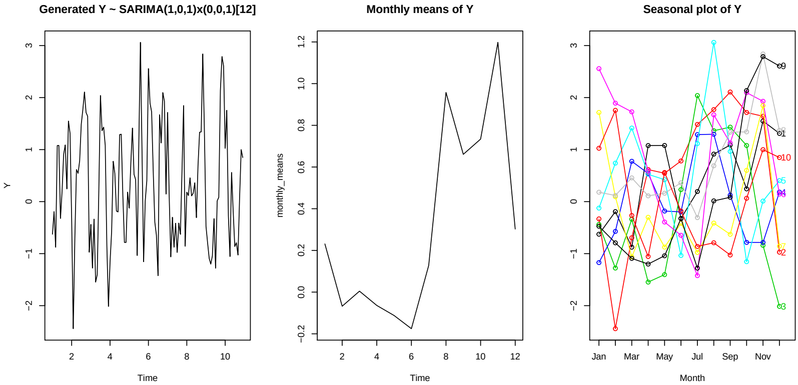
$$\text{Seasonal AR: } \Phi(L^S) = 1 - \Phi_1 L^S - ... - \Phi_p L^{S \cdot P}$$
$$\text{Seasonal MA: } \Theta(L^S) = 1 + \Theta_1 L^S + ... + \Theta_q L^{S \cdot Q}$$

Note that on the left side of equation the seasonal and non-seasonal AR components multiply each other, and on the right side of equation the seasonal and non-seasonal MA components multiply each other.

For example, a $SARIMA(1,0,1)(0,0,1)_{12}$ model could be written:

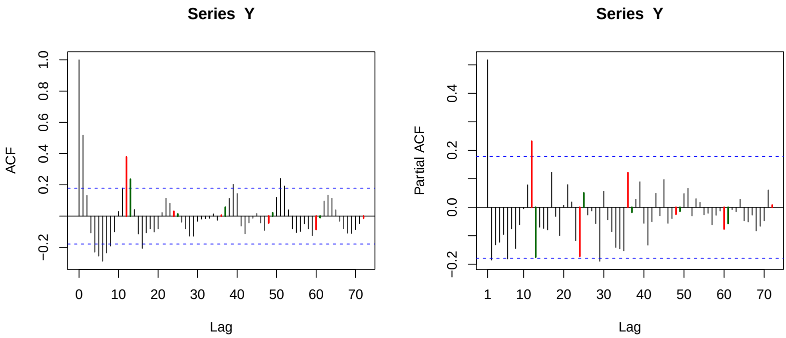
$$(1-\phi L)Y_t = (1+\theta L)\cdot(1+\Theta L^{12})\epsilon_t$$
$$(1-\phi L)Y_t = (1+\theta L+\Theta L^{12}+\theta\Theta L^{12+1})\epsilon_t$$
$$Y_t = \phi Y_{t-1}+\epsilon_t+\theta\epsilon_{t-1}+\Theta\epsilon_{t-12}+\theta\Theta\epsilon_{t-13}$$

where $\phi = 0.4$, $\theta = 0.2$ and $\Theta = 0.5$.



There is seasonality, but no trend.

Examine the ACF and PACF of the data:



Overall, both ACF and PACF plots seem to be declining - a possible $ARMA(1, 1)$ model for the non-seasonal model component.

From the ACF plot - the first 12th lag is significant and every other 12th lag (24, 36, etc.) is not (i.e. seasonal cut-off after the first period lag). From the PACF plot - the $12th$, $24th$, $36th$, etc. lags are declining. Also note the $13th$ lag, ϵ_{t-13}. This means that the seasonality could be a $MA(1)$ model.

```r
seas_mdl <- Arima(Y,
      order = c(1, 0, 1),
      seasonal = list(order = c(0, 0, 1), period = 12),
      include.mean = FALSE)
seas_mdl
```

```
## Series: Y
## ARIMA(1,0,1)(0,0,1)[12] with zero mean
##
## Coefficients:
##          ar1     ma1    sma1
##       0.4148  0.1870  0.4802
## s.e.  0.1369  0.1432  0.0902
##
## sigma^2 estimated as 0.7888:  log likelihood=-156.28
## AIC=320.56   AICc=320.91   BIC=331.71
```

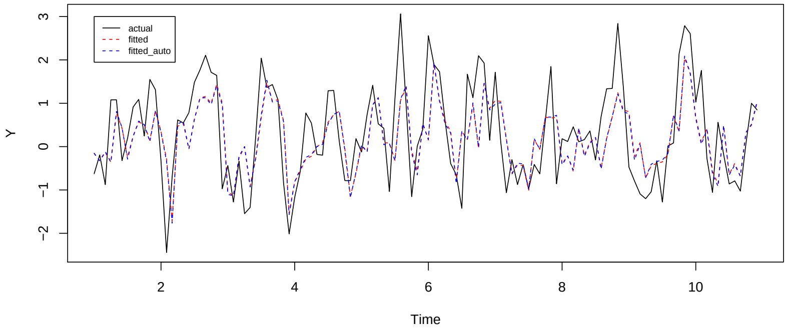
Our estimated model coefficients are: $\hat{\phi} = 0.4919$, $\hat{\theta} = 0.2058$ and $\hat{\Theta} = 0.4788$. Note `Y` is a `ts()` object, i.e. `Y <- ts(Y, freq = 12)`.

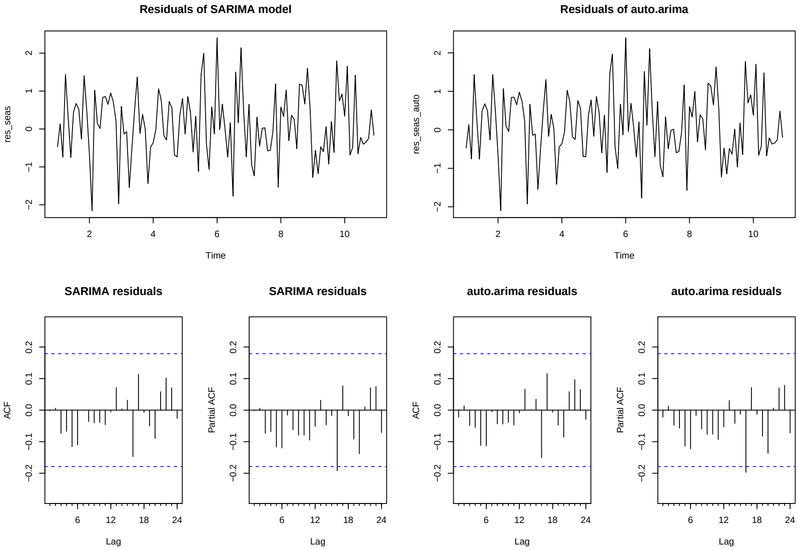
In comparison, the `auto.arima` suggests a slightly different ARMA model:

```
capture.output(summary(seas_mdl_auto <- auto.arima(Y)))[2]
```

```
## [1] "ARIMA(2,0,0)(0,0,1)[12] with zero mean "
```

```
plot.ts(Y, lwd = 1)
lines(fitted(seas_mdl), col = "red", lty = 2)
lines(fitted(seas_mdl_auto), col = "blue", lty = 2)
legend(x = 1, y = 3, c("actual", "fitted", "fitted_auto"),
  col = c("black", "red", "blue"), lty = c(1, 2, 2), cex = 0.7)
```





From the ACF and PACF plots the manually specified $SARIMA(1,0,1)(0,0,1)_{12}$ model residuals are very close to the $SARIMA(2,0,0)(0,0,1)_{12}$ residuals from the `auto.arima` function.

Example: $SARIMA(0,0,1)(0,0,1)_{12}$ model

$$Y_t = (1+\theta L)\cdot(1+\Theta L^{12})\epsilon_t \iff Y_t = (1+\theta L + \Theta L^{12} + \theta\Theta L^{12+1})\epsilon_t$$
$$\iff Y_t = \epsilon_t + \theta\epsilon_{t-1} + \Theta\epsilon_{t-12} + \theta\Theta\epsilon_{t-13}$$

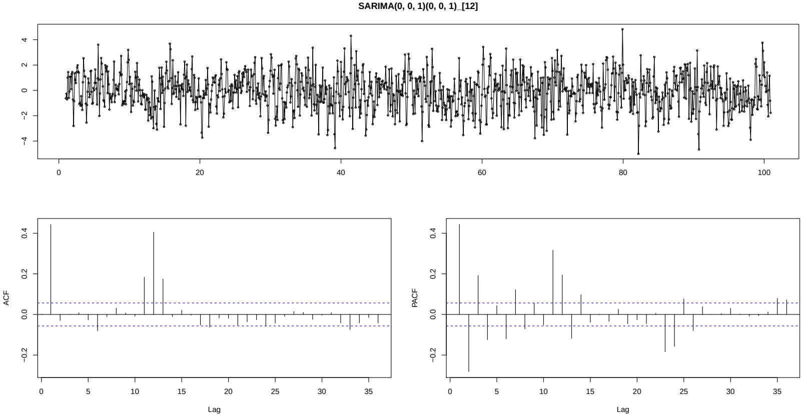
where $\theta = 0.7$ and $\Theta = 0.6$.
Note that:

$$Y_{t-11} = \epsilon_{t-11} + \theta\epsilon_{t-12} + \Theta\epsilon_{t-23} + \theta\Theta\epsilon_{t-24}$$
$$Y_{t-12} = \epsilon_{t-12} + \theta\epsilon_{t-13} + \Theta\epsilon_{t-24} + \theta\Theta\epsilon_{t-25}$$
$$Y_{t-13} = \epsilon_{t-13} + \theta\epsilon_{t-14} + \Theta\epsilon_{t-25} + \theta\Theta\epsilon_{t-26}$$

which means that the covariance between Y_t and Y_{t-11} is non-zero:

$$\mathbb{C}\text{ov}(Y_t, Y_{t-11})$$
$$= \mathbb{C}\text{ov}\left(\epsilon_t + \theta\epsilon_{t-1} + \Theta\epsilon_{t-12} + \theta\Theta\epsilon_{t-13},\ \epsilon_{t-11} + \theta\epsilon_{t-12} + \Theta\epsilon_{t-23} + \theta\Theta\epsilon_{t-24}\right)$$
$$= \mathbb{C}\text{ov}\left(\Theta\epsilon_{t-12},\ \theta\epsilon_{t-12}\right) = \Theta\theta\cdot\sigma^2 \neq 0$$

It can also be shown that $\mathbb{C}\text{ov}(Y_t, Y_{t-12}) \neq 0$, $\mathbb{C}\text{ov}(Y_t, Y_{t-13}) \neq 0$ and the remaining terms, like $\mathbb{C}\text{ov}(Y_t, Y_{t-10}) = \mathbb{C}\text{ov}(Y_t, Y_{t-14}) = 0$.

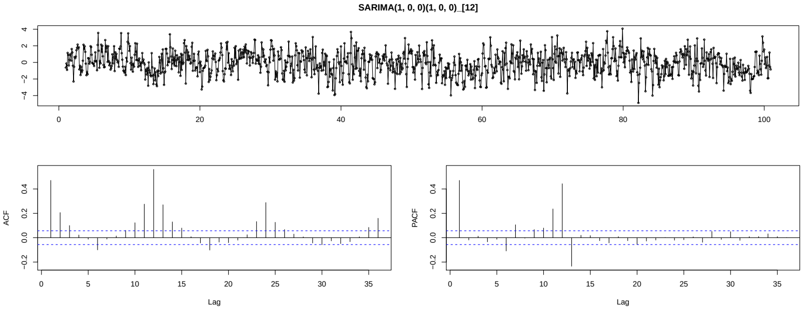


- the spikes at `lags` 1, 11, 12 and 13 in the **ACF**. Remember that the feature of a moving-average process - **ACF** has a sharp cutoff;
- this model has non-seasonal and seasonal *MA* terms, so the PACF tapers nonseasonally, following `lag` 1, and tapers seasonally that is near `lag = 12`, and again near `lag  = 2*12=24` and so on.

Example: *SARIMA*$(1,0,0)(1,0,0)_{12}$ model

$$(1-\Phi L^{12})(1-\phi L)Y_t=\epsilon_t \iff (1-\phi L-\Phi L^{12}+\phi\Phi L^{13})Y_t=\epsilon_t$$
$$\iff Y_t=\phi Y_{t-1}+\Phi Y_{t-12}-\phi\Phi Y_{t-13}+\epsilon_t$$

where $\phi=0.6$ and $\Phi=0.5$.



- there are *distinct* spikes at `lags` 1, 12 and 13 in the **PACF**. Remember that the feature of an autoregressive process - **PACF** has a sharp cutoff.

Revisiting Time Series Smoothing

In some cases, linear regression cannot clarify relationships between variables and cannot detect the trend of a data series. For this reason, we can apply other regression methods in statistics.

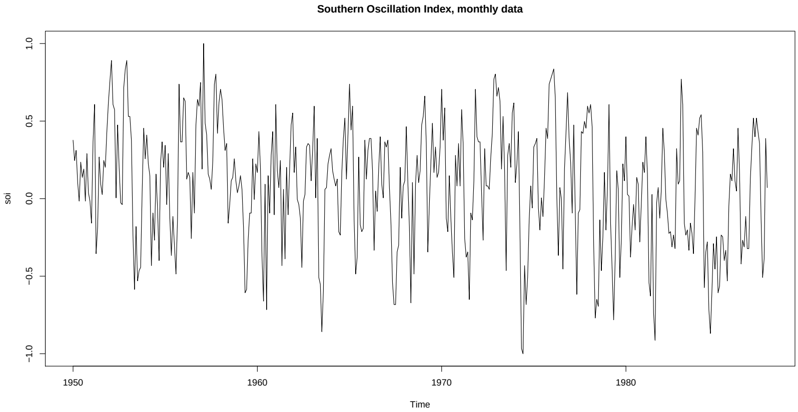
A smoother is a function or procedure for drawing a smooth curve through a scatter diagram. Similarly to linear regression (in which the “curve” is a straight line), the smooth curve is drawn in such a way as to have some desirable properties.

In general, the properties are that:

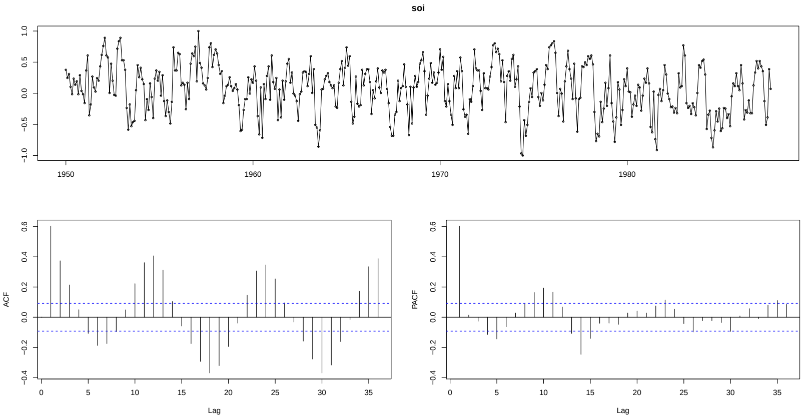
- the curve is indeed smooth;
- locally, the curve minimizes the variance of the residuals, or prediction error.

Depending on the method, we can either attempt to specify the function ourselves to control the degree of smoothing (e.g. moving average, single or double exponential smoothing), or we can estimate the optimal parameters (e.g. Holt-Winters exponential smoothing).

- The SOI measures changes in air pressure, related to sea surface temperatures in the central Pacific Ocean.
- The central Pacific warms every three to seven years due to the El *Niño* effect, which has been blamed for various global extreme weather events.
- Periodic behavior is of interest because underlying processes of interest may be regular and the rate or frequency of oscillation characterizing the behavior of the underlying series would help to identify them.



- Do you notice any trends in the data?
- Do you notice any seasonalities in the data?

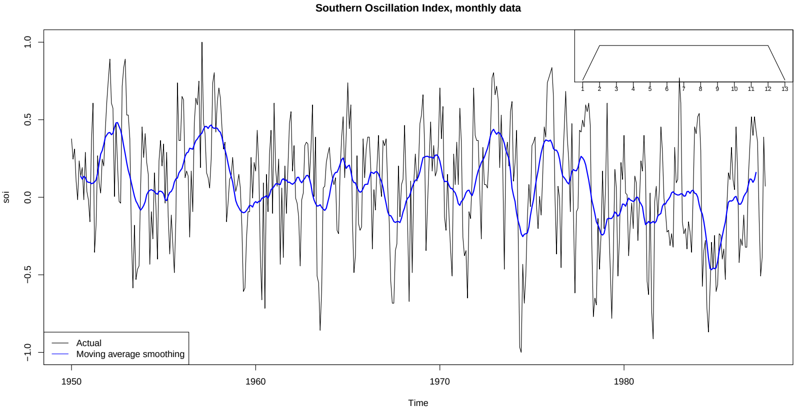


Think about the differences between:

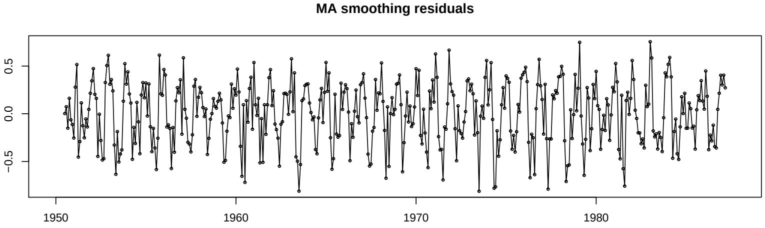
- An autoregressive process with oscillating ACF (e.g. $Y_t = 1.5 \cdot Y_{t-1} - 0.9 \cdot Y_{t-2} + \epsilon_t$);
- An ACF of a SARMA model;
- An ACF of a process with a seasonal component;
- An ACF of a process with a Trend component;

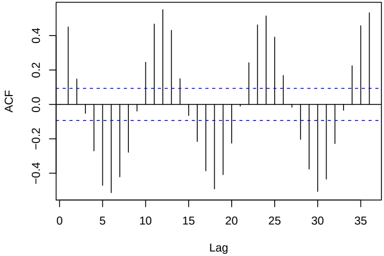
Additional Time Series Smoothing Methods

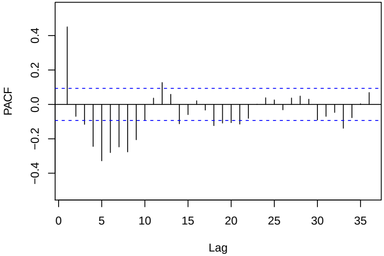
Previously we discussed using a moving average smoothing method, which is useful for **discovering** certain traits in a time series, such as trend and/or seasonal components.



While the moving average seems to work quite well, it appears to be too *choppy*. **We can obtain a smoother fit using the normal distribution for the weights**







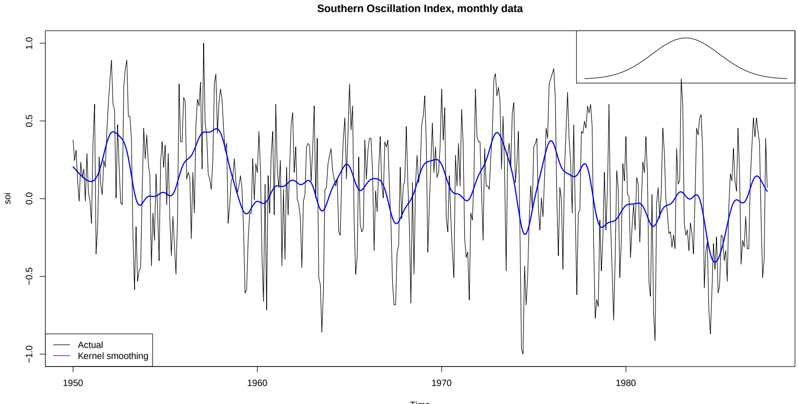
Since we have eliminated the trend component - the seasonality is more pronounced in the residuals.

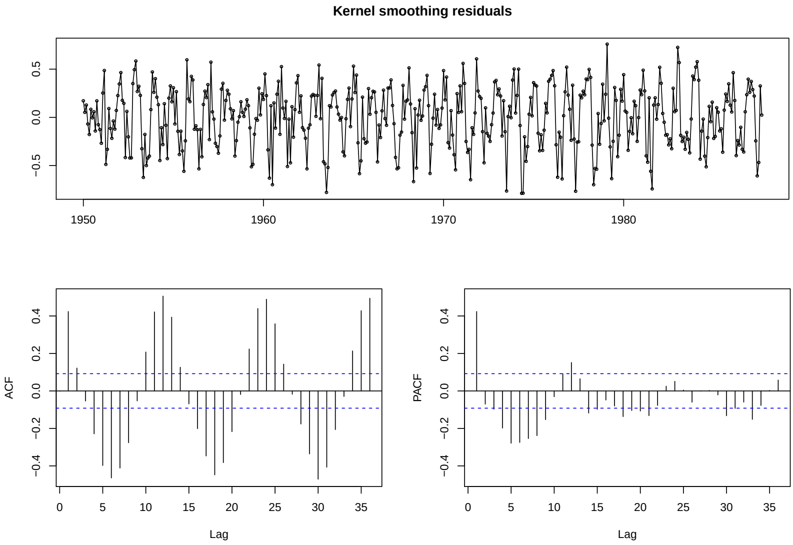
Kernel smoothing

Kernel smoothing - a moving average smoothing method that uses a kernel as the weight function to average the observations.

$$\widehat{T}_t = \sum_{i=1}^{T} w_i(t) Y_i, \quad w_i(t) = K\left(\frac{t-i}{b}\right) \Big/ \sum_{j=1}^{T} K\left(\frac{t-j}{b}\right)$$

where $K(\cdot)$ is a **kernel** function. This estimator is often called the **Nadaraya-Watson estimator**.





Since we have eliminated the trend component - the seasonality is more pronounced in the residuals.

Lowess Smoothing

Another approach to smoothing a time plot is nearest neighbor regression. The technique is based on k-nearest neighbors ($k - NN$) regression, where we use only data points in the vicinity of Y_t – namely $Y_{t-k/2}, ..., Y_t, ..., Y_{t+k/2}$ – and predict Y_t via regression, then set $\widehat{T}_t = \widehat{Y}_t$.

The **lo**cally **we**ighted **s**catterplot **s**moothing (**LOWESS**) method makes no assumptions about the form of the relationship, and allows the form to be discovered using the data itself.

Note: the next slide will have a general outline of the process - try to get the basic idea behind it.

The basic idea of LOWESS is close to the nearest neighbor regression (see here and here):

- start with a local polynomial (i.e. $k - NN$) least squares fit and then to use robust methods to obtain the final fit. Specifically, one can first fit a polynomial regression in a neighborhood of Y_t:

$$\frac{1}{T}\sum_{i=1}^{T} W_{k,i}(Y_t)\left(Y_i - \sum_{j=0}^{p}\beta_j Y_t^j\right)^2 \longrightarrow \min_{\beta_0,\dots,\beta_p}$$

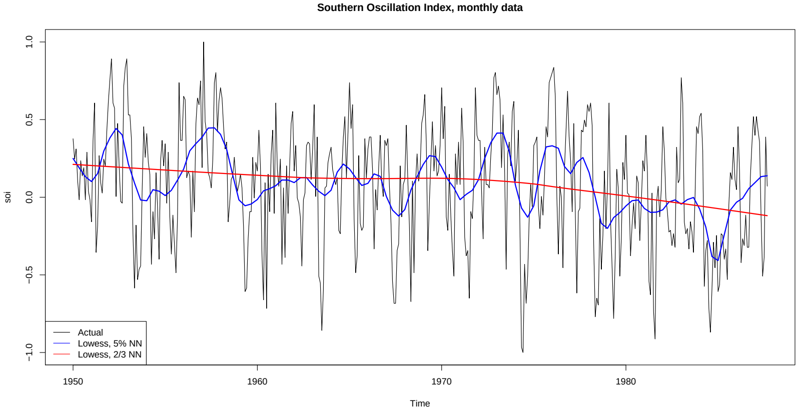
  where $W_{k,i}(Y_t)$ are the $k - NN$ weights:

$$W_{k,i}(Y_t) = Tri\left(\frac{Y_i - Y_t}{h}\right)$$

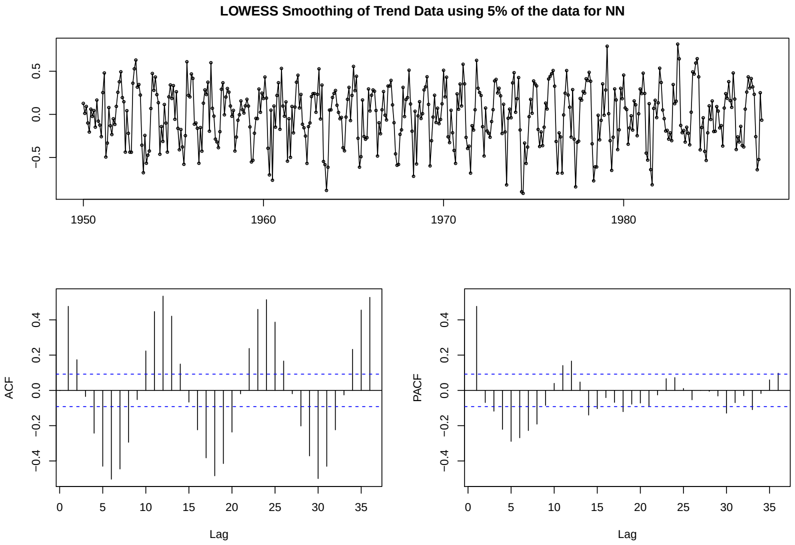
  where h is the k^{th} smallest distance $|Y_i - Y_t|$, $i = 1, ..., T$ and $Tri(x) = (1 - |x|^3)^3$ if $|x| < 1$ and $Tri = 0$ otherwise.
- Calculate the residuals $\widehat{\epsilon}_t$ and the scale parameter $\widehat{\sigma} = Median(\widehat{\epsilon}_t)$;
- Define robustness weights $\delta_i = K\left(\frac{\widehat{\epsilon}_i}{6\widehat{\sigma}}\right)$, where $K(u) = (15/16)(1-u)^2$, if $|u| \leq 1$ and $K(u) = 0$ otherwise.
- Use $\delta_i W_{k,i}(Y_t)$ instead of $W_{k,i}(Y_t)$ and estimate new parameters $\widehat{\beta}_0, ..., \widehat{\beta}_p$;
- Repeat the process a predefined number of times.

Recommended to use $p = 1$ for computational efficiency.

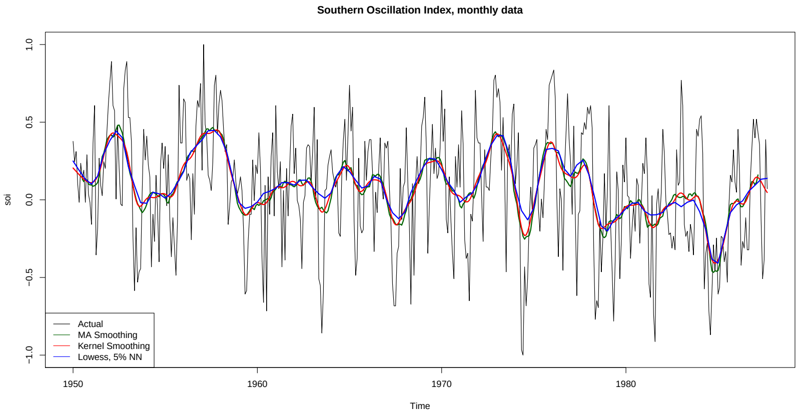
The larger the fraction of nearest neighbors included, the smoother the fit will be.



Insight from the decomposition: a decreasing (or, negative) trend in SOI indicates the long-term warming of the Pacific Ocean.



Since we have eliminated the trend component - the seasonality is more pronounced in the residuals.



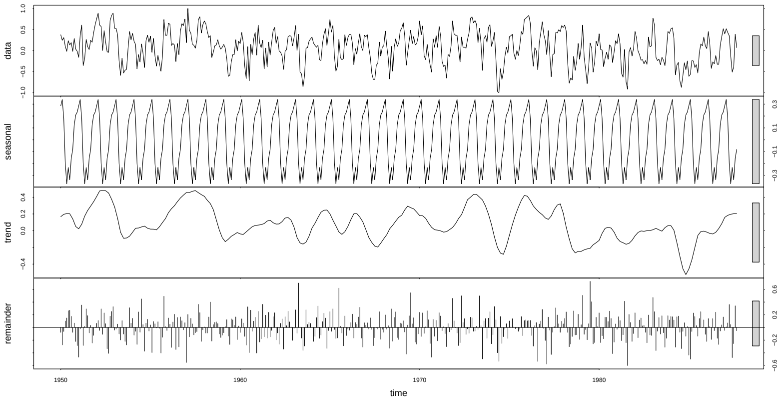
Q: Which method is better?

A: Depends on whether we can decompose seasonality after removing the trend.

Seasonal Decomposition of Time Series by Loess

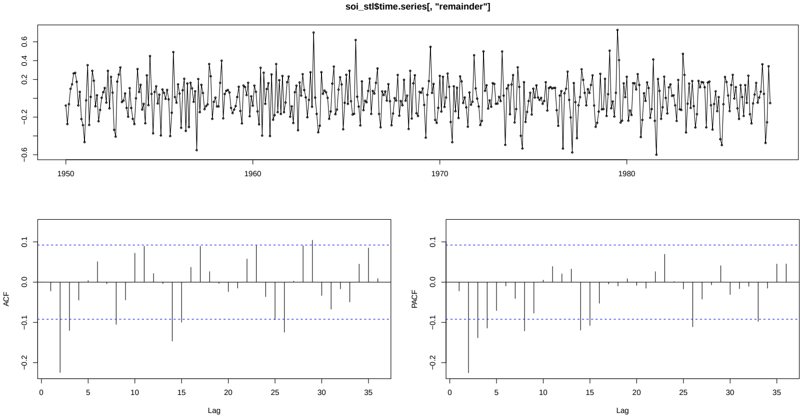
To decompose the seasonal component, use the `stl()` function in R.

`t.window` controls the ‘wiggliness’ of the trend components and `s.window` controls the variation on the seasonal component.



It is a very versatile and robust decomposition method - the seasonal component is allowed to change over time, the smoothness of the trend cycle can also be controlled by the user.

This method is robust to outliers, however it is only available for additive decompositions.



- No more seasonality;
- No more trend;

In R the two main robust "user-friendly" methods are:

- stl() - does not allow forecasting. To remedy this forecast::forecast() can be used, which applies a non-seasonal forecasting method to the seasonally adjusted data and re-seasonalizes using **the last year of the seasonal component**. See here.
- ets() - Holt-Winters Exponential smoothing, allows forecasting.

Remarks on LOWESS (or LOESS)

Advantages:

- LOWESS does not require specification of a functional form to fit a model to the data sample;
- LOWESS is very flexible, making it ideal for modeling complex processes for which no theoretical models exist.

Disadvantages:

- LOWESS makes less efficient use of data than other least squares methods. It requires fairly large, densely sampled data sets in order to produce good models;
- LOWESS does not produce a regression function that is easily represented by a mathematical formula. This can make it difficult to transfer the results of an analysis to other people;
- LOWESS is a computationally intensive method that is also prone to the effects of outliers in the data set, like other least squares methods.

Local Linear Forecast Using Smoothing (Cubic) Splines

Suppose that our time series Y_t, $t = 1, ..., T$ exhibits a non-linear trend. We are interested in forecasting this series by extrapolating the trend using a linear function, which we estimate from the historical data.

An obvious way to smooth data would be to fit a polynomial regression in terms of time. For example, a cubic polynomial would have $Y_t = T_t + E_t$, where

$$T_t = \beta_0 + \beta_1 \cdot t + \beta_2 \cdot t^2 + \beta_3 \cdot t^3$$

In practice we would fit this cubic polynomial on Y_t via OLS to obtain $\widehat{T}_t$.

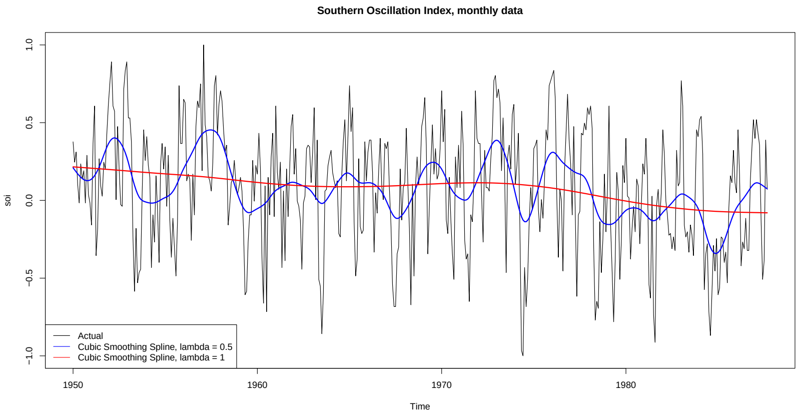
- An extension of polynomial regression is to first divide time $t = 1, ..., T$ into k intervals: $[t_0, t_1], [t_1 + 1, t_2], ..., [t_{k-1} + 1, t_k]$ with $t_0 = 1$ and $t_k = T$. The values $t_0, ..., t_k$ are called **knots**. Each interval fits a polynomial regression, typically of order 3, and this is called **cubic splines**.
- A similar method is called **smoothing splines**, which minimizes a compromise between the fit and the degree of smoothness.

For equally spaced time series, a **cubic smoothing spline** can be defined as the function $\hat{f}(t)$, which minimizes:

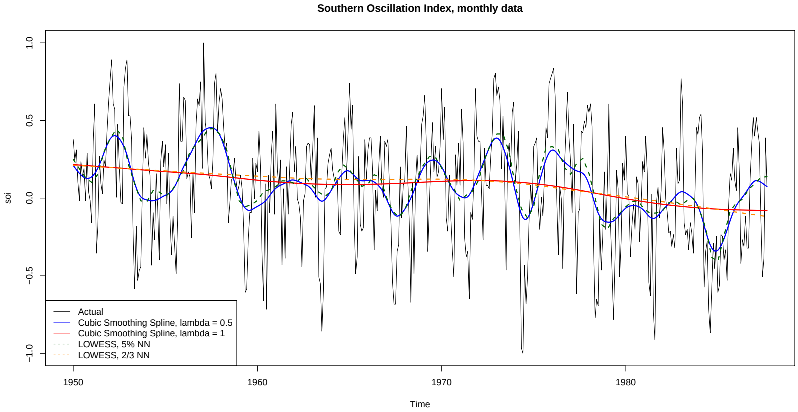
$$\sum_{t=1}^{T}(Y_t - f(t))^2 + \lambda \int_S (f''(u))^2 du$$

over all twice differentiable functions f on S where $[1, T] \subseteq S \subseteq \mathbb{R}$. The smoothing parameter λ is controlling the trade-off between fidelity to the data and roughness of the function estimate. The larger the value of λ, the smoother the fit.

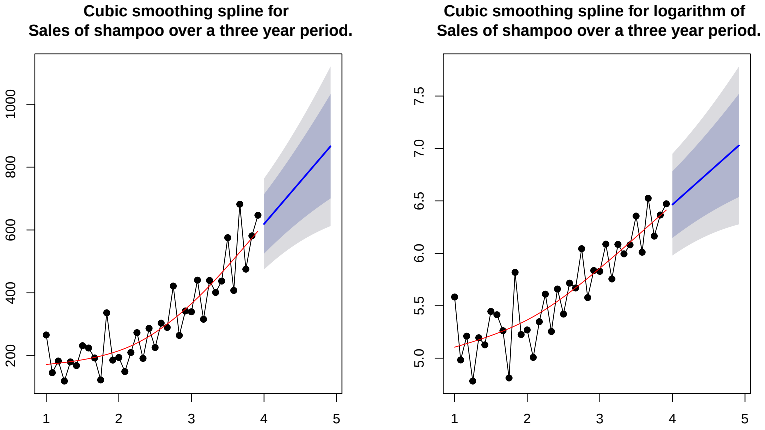
- Link to the paper presenting this method can be found `[here]`.
- The cubic smoothing spline model is equivalent to an $ARIMA(0, 2, 2)$ model (this model will be presented later) but with a restricted parameter space.
- The advantage of the cubic smoothing spline approach over the full *ARIMA* model is that it provides a smooth historical trend as well as a linear forecast function.



Results appear similar to other trend estimation methods.


Southern Oscillation Index, monthly data
soi
1.0
0.5
0.0
−0.5
−1.0
1950
1960
1970
1980
Time
Actual
Cubic Smoothing Spline, lambda = 0.5
Cubic Smoothing Spline, lambda = 1
LOWESS, 5% NN
LOWESS, 2/3 NN

```r
data(shampoo)
fcast <- splinef(shampoo, h = 12)
fcast.l <- splinef(log(shampoo), h = 12)
par(mfrow = c(1, 2))
plot(fcast, main = "Cubic smoothing spline for \n Sales of shamp
plot(fcast.l, main = "Cubic smoothing spline for logarithm of \n
```



The X-12-ARIMA or X-13-ARIMA-SEATS Seasonal Adjustment

Link to R package documentation [here] and [here].

X-13ARIMA-SEATS is a seasonal adjustment software produced, distributed, and maintained by the United States Census Bureau.

X-13ARIMA-SEATS combines the current filters used in X-12-ARIMA with ARIMA-model-based adjustment as implemented in the program SEATS.

In SEATS, the seasonal and trend filters are estimated simultaneously based on the ARIMA model.

The new program still provides access to all of X-12-ARIMA's seasonal and trend filters and to the diagnostics.

```r
X_13 <- seasonal::seas(x = AirPassengers)
capture.output(summary(X_13))[6:11]
```

```
[1] "                    Estimate Std. Error z value Pr(>|z|)
[2] "Weekday           -0.0029497  0.0005232  -5.638 1.72e-08 **
[3] "Easter[1]          0.0177674  0.0071580   2.482   0.0131 *
[4] "AO1951.May         0.1001558  0.0204387   4.900 9.57e-07 **
[5] "MA-Nonseasonal-01  0.1156205  0.0858588   1.347   0.1781
[6] "MA-Seasonal-12     0.4973600  0.0774677   6.420 1.36e-10 **
```

We can generate a nice `.html` output of our model with:

```r
seasonal::out(X_13)
```

where (using the [documentation, Tables 4.1 and 7.28]):

- `Weekday` - One Coefficient Trading Day, the difference between the number of weekdays and the 2.5 times the number of Saturdays and Sundays
- `AO1951.May` - Additive (point) outlier variable, AO, for the given date or observation number. In this case it is the *regARIMA* (regression model with ARIMA residuals) outlier factor for the point at time 1951-May of the series;
- `Easter[1]` - Easter holiday regression variable for monthly or quarterly flow data which assumes the level of daily activity changes on the `[1]`-st day before Easter and remains at the new level through the day before Easter.
- `MA-Nonseasonal-01` - coefficients of the non-seasonal components of the ARMA model for the *differenced* residuals, $\nabla\epsilon_t$.
- `MA-Seasonal-12` - coefficients of the seasonal components of the ARMA model for the *differenced* residuals $\nabla_{12}\epsilon_t$.

Looking at ?series, we can extract different data:

```
#Estimate of the Seasonal factors:
X_13.seas <- seasonal::series(X_13, "history.sfestimates")
```

```
## specs have been added to the model: history
```

```
#Estimate of the seasonally adjusted data
X_13.deseas <- seasonal::series(X_13, "history.saestimates")
```

```
## specs have been added to the model: history
```

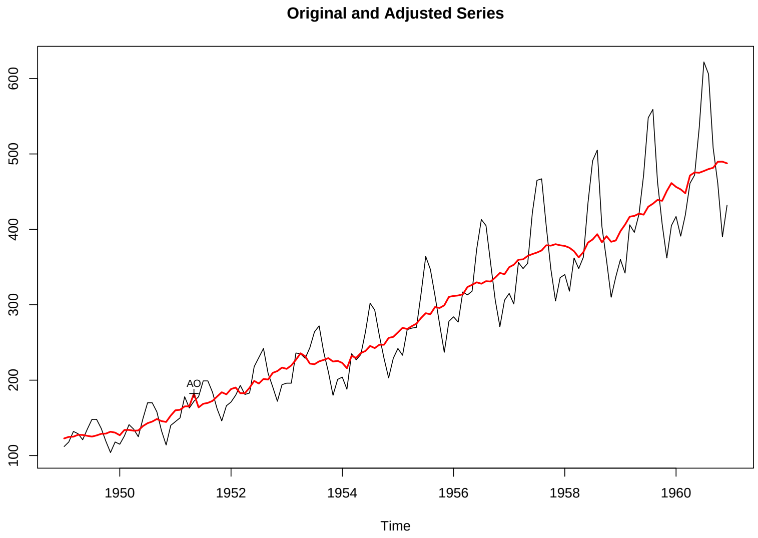
```
#Estimate of the trend component
X_13.trend <- seasonal::series(X_13, "history.trendestimates")
```

```
## specs have been added to the model: history
```

```
#Forecasts:
X_13.forc <- seasonal::series(X_13, "forecast.forecasts")
```

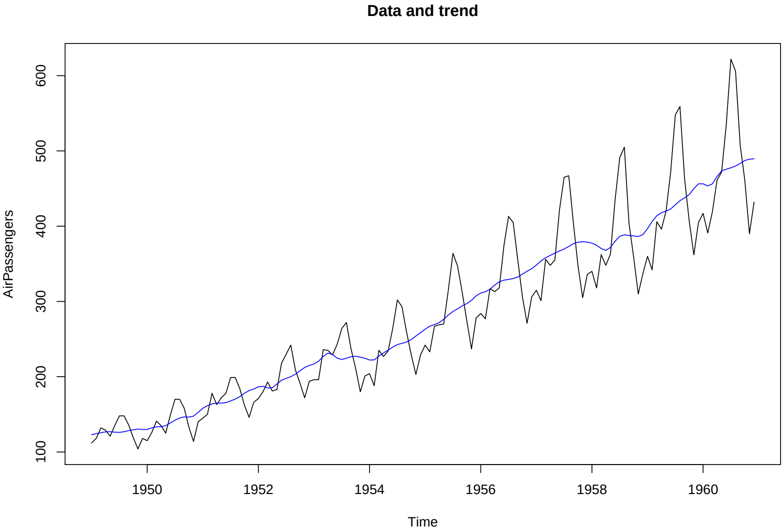
```
## specs have been added to the model: forecast
```

```r
plot(X_13)
```

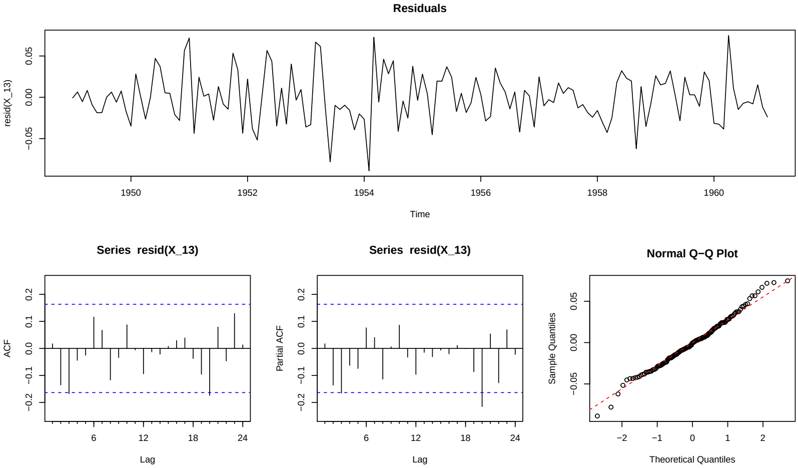


Note: The series is adjusted for seasonality only.

```r
plot(AirPassengers, main = "Data and trend")
lines(X_13$data[, "trend"], col = "blue")
```



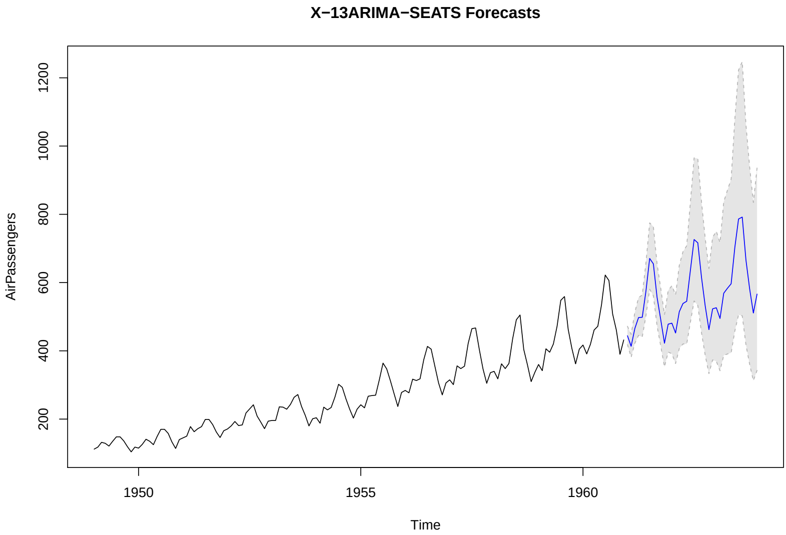
```
layout(matrix(c(1, 1, 1, 2, 3, 4), 2, 3, byrow = TRUE))
plot.ts(resid(X_13), main = "Residuals")
forecast::Acf(resid(X_13)); forecast::Pacf(resid(X_13))
qqnorm(resid(X_13)); qqline(resid(X_13), lty = 2, col = "red")
```



We can also plot the forecasts along with their confidence intervals:

```
#Set the x and y axis separtely
x.lim = c(head(time(AirPassengers), 1), tail(time(X_13.forc), 1))
y.lim = c(min(AirPassengers), max(X_13.forc[,"upperci"]))

#Plot the time series:
plot.ts(AirPassengers,  xlim = x.lim, ylim = y.lim,
        main = "X-13ARIMA-SEATS Forecasts")
#Plot the shaded forecast confidence area:
polygon(c(time(X_13.forc), rev(time(X_13.forc))),
        c(X_13.forc[,"upperci"], rev(X_13.forc[,"lowerci"])),
        col = "grey90", border = NA)
#Plot the forecasts along with their lwoer and upper bounds:
lines(X_13.forc[,"forecast"], col = "blue")
lines(X_13.forc[,"lowerci"], col = "grey70", lty = 2)
lines(X_13.forc[,"upperci"], col = "grey70", lty = 2)
```



X−13ARIMA−SEATS Forecasts
AirPassengers
200
400
600
800
1000
1200
1950
1955
1960
Time

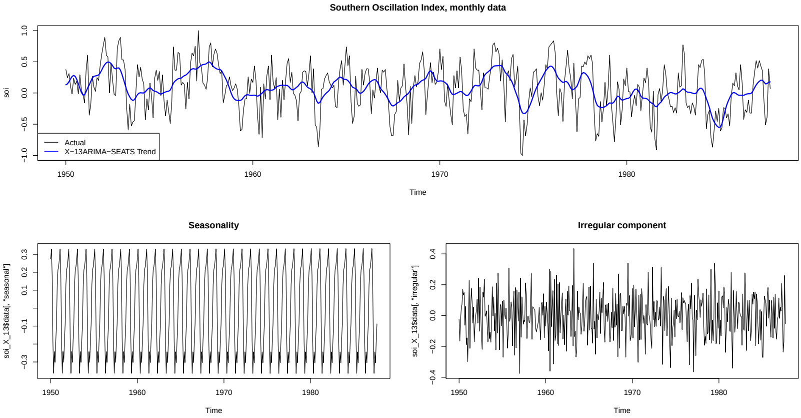
Looking back at our SOI example

```
soi_X_13 <- seasonal::seas(x = soi)
```

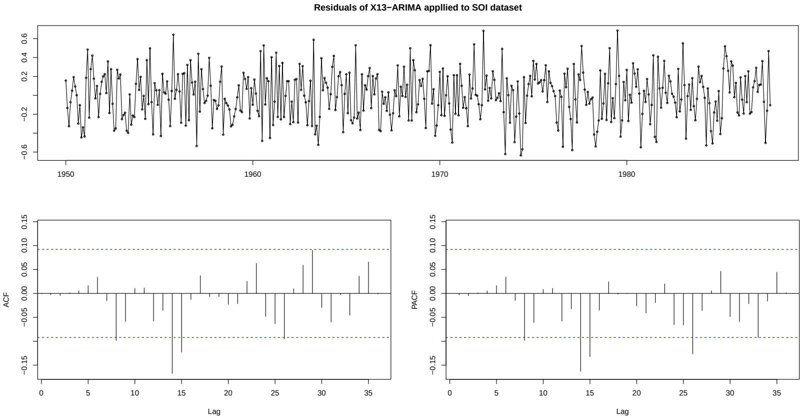
It may sometimes be the case that `seasonal::series(...)` will not work, so the relevant results can always be extracted directly.

```
print(head(soi_X_13$data))
```

```
               final   seasonal seasonaladj     trend   irregular
Jan 1950  0.10145039  0.2755496  0.10145039 0.1316089 -0.02457600
Feb 1950 -0.08329878  0.3292988 -0.08329878 0.1452373 -0.16458940
Mar 1950  0.13144671  0.1795533  0.13144671 0.1808291 -0.02679486
Apr 1950  0.28609804 -0.1820980  0.28609804 0.2264334  0.04469685
May 1950  0.34707990 -0.3630799  0.34707990 0.2641907  0.07335916
Jun 1950  0.47731379 -0.2423138  0.47731379 0.2810873  0.17002810
          adjustfac
Jan 1950  0.2755496
Feb 1950  0.3292988
Mar 1950  0.1795533
Apr 1950 -0.1820980
May 1950 -0.3630799
Jun 1950 -0.2423138
```



Southern Oscillation Index, monthly data
soi
1.0
0.5
0.0
−0.5
−1.0
Actual
X−13ARIMA−SEATS Trend
1950
1960
1970
1980
Time
Seasonality
soi_X_13$data[, "seasonal"]
0.3
0.2
0.1
−0.1
−0.3
1950
1960
1970
1980
Time
Irregular component
soi_X_13$data[, "irregular"]
0.4
0.2
0.0
−0.2
−0.4
1950
1960
1970
1980
Time



Residuals of X13−ARIMA appllied to SOI dataset
0.6
0.4
0.2
−0.2
−0.6
1950
1960
1970
1980
ACF
PACF
0.15
0.10
0.05
0.00
−0.05
−0.15
0
5
10
15
20
25
30
35
Lag

Recap: Seasonality

- Seasonality in a time series is a regular pattern, which repeats over d time units, where d is called the (seasonality) period - the number of time units/periods, until the pattern repeats again. This can be written as $S_t = S_{t+d}$;
- Seasonality usually causes the series to be nonstationary, since the average values from particular seasonal periods (e.g. summer) may be different than the average values at other times (e.g. winter);
  - We can remove the trend by differencing the series, e.g. by transforming the series to $(1 - L)Y_t$;
  - We can remove the seasonality by seasonally differencing the series, e.g. by transforming the series to $(1 - L^d)Y_t$;
  - To remove both trend and seasonality by applying both non-seasonal and seasonal differencing, e.g. $(1 - L)(1 - L^d)Y_t$;
- The seasonal ARMA model incorporates both non-seasonal and seasonal factors in a multiplicative model. In a seasonal ARMA model, the seasonal AR and MA terms predict Y_t using lagged values, where the lags are multiples of d.

Recap: Time Series Decomposition

- Decomposition procedures are used in time series to describe the trend and seasonal factors in a time series. More extensive decompositions might also include long-run cycles, holiday effects, day of week effects and so on (see $X - 13ARIMA - SEATS$).
- Decomposition can be used to estimate and remove seasonality in order to calculate seasonally adjusted values. These adjusted values allow to see the trend more clearly. For instance, U.S. unemployment tends to decrease in the summer due to increased employment in agricultural areas. So, it would appear that unemployment decreased from winter to summer, however, this does not indicate that there is a trend toward lower unemployment in the country.
- Decomposition is usually done in three steps: (i) trend estimation; (ii) seasonality estimation on the de-trended series; (ii) remainder (i.e. random) component estimation by removing the seasonal and trend component from the original series. The decomposition method depends on whether the series is additive or multiplicative.
- The random component can be analyzed for such things as the mean, variance, or possibly even for whether the component is actually random or might be modeled with an ARIMA model.

Recap: Time Series Smoothing

- Smoothing is usually done to help us better see patterns in the time series. The term **filter** is sometimes used to describe a smoothing procedure.
  - For Series with a trend, we may smooth out the irregular roughness to see a clearer signal.
    - For non-seasonal data you should experiment with moving averages of different spans. Those spans of time could be relatively short. The objective is to knock off the rough edges to see what trend or pattern might be there.
  - For seasonal data, we might smooth out the seasonality so that we can identify the trend.
    - To take away seasonality from a series, so we can better see the trend, we would use a moving average with a length equal to d (the seasonal length). If d is even, then a centered moving average is needed.
- Some possible smoothing methods include - moving average smoothing; single, double or triple (Holt-Winters) exponential smoothing. Additional methods include kernel smoothing, lo(w)ess smoothing, cubic smoothing splines and more.
- Smoothing doesn't provide us with a model, but it can be a good first step in describing various components of the series.