Case Study: Modelling Airline Passengers

Arun K. Tangirala

October 10, 2017

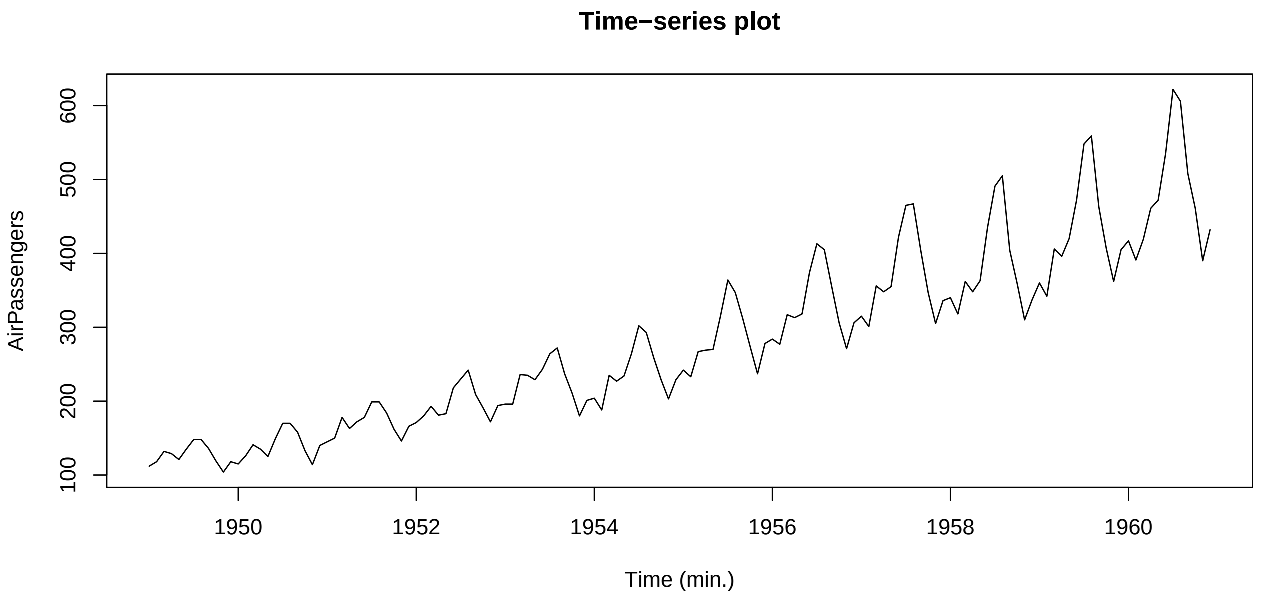
Background

This is a case study of the widely discussed AirPassengers data set that is shipped with the `datasets` package in `R`.

Preliminary analysis

We shall first load the data and perform a preliminary analysis including visualization of the series (this is important).

```
vk <- AirPassengers
# Plot the series
plot(vk, main = "Time-series plot", xlab = "Time (min.)", ylab = "AirPassengers")
```



```
# Obtain the statistical summary
summary(vk)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   104.0   180.0   265.5   280.3   360.5   622.0
```

It is clear from the plot that the series has a trend (polynomial type) and a periodic (seasonal) component in addition to some irregularities (stochastic component).

Non-parametric analysis

Based on the visual analysis (it is also possible to conduct a formal test). the first step therefore is to eliminate the trend by fitting a polynomial of appropriate order.

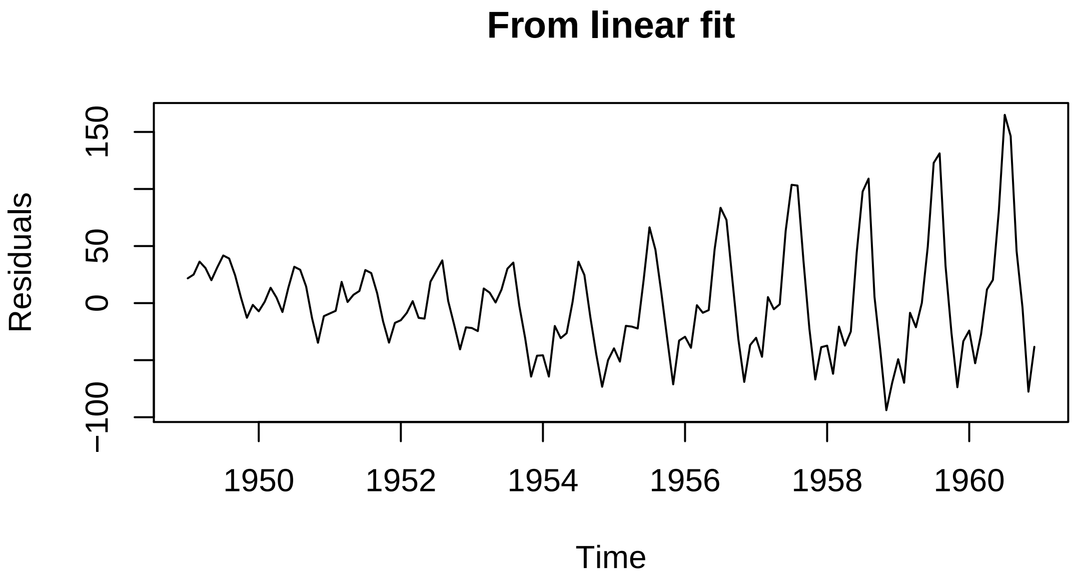
We shall begin with a linear trend.

```r
tvec <- time(vk)
modlm1 <- lm(vk ~ tvec)
summary(modlm1)
```

```
##
## Call:
## lm(formula = vk ~ tvec)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -93.858 -30.727  -5.757  24.489 164.999
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -62055.907   2166.077  -28.65   <2e-16 ***
## tvec            31.886      1.108   28.78   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 46.06 on 142 degrees of freedom
## Multiple R-squared:  0.8536, Adjusted R-squared:  0.8526
## F-statistic: 828.2 on 1 and 142 DF,  p-value: < 2.2e-16
```

From the analysis of the resulting model, it is clear that both the intercept and slope estimates are significant (low p-values). In order to determine the adequacy of the fit we turn to the residuals.

```r
tattr <- tsp(vk)
reslm1 <- ts(residuals(modlm1), start = tattr[1], frequency = tattr[3])
plot(reslm1, ylab = "Residuals", main = "From linear fit")
```

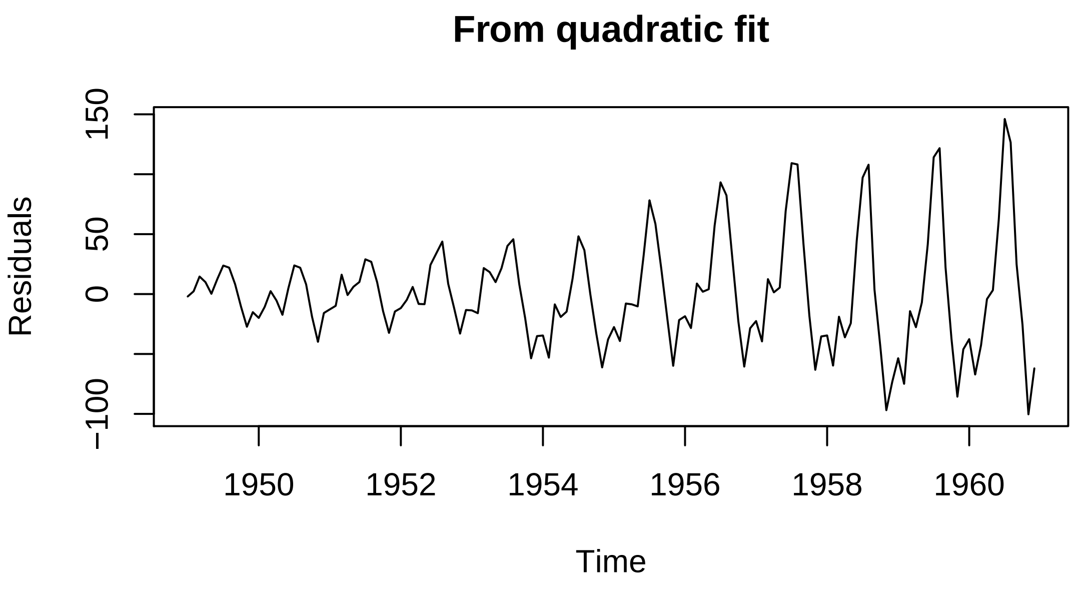


It is evident that a trend still remains. Proceeding this way iteratively, we converge to a quadratic trend.

```r
Tmat <- poly(tvec, degree = 2, raw = T)
modlm2 <- lm(vk ~ Tmat)
summary(modlm2)
```

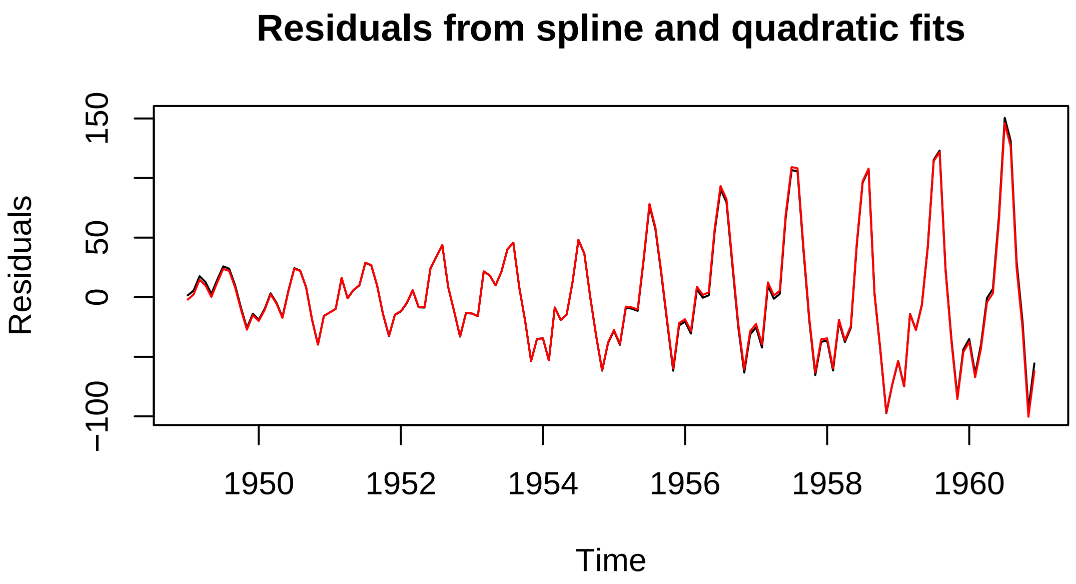
```
##
## Call:
## lm(formula = vk ~ Tmat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -100.353  -27.339   -7.442   21.603  146.116
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.795e+06  1.333e+06   2.848  0.00506 **
## Tmat1       -3.914e+03  1.363e+03  -2.871  0.00473 **
## Tmat2        1.009e+00  3.487e-01   2.894  0.00441 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 44.91 on 141 degrees of freedom
## Multiple R-squared:  0.8618, Adjusted R-squared:  0.8599
## F-statistic: 439.8 on 2 and 141 DF,  p-value: < 2.2e-16
```

```r
reslm2 <- ts(residuals(modlm2), start = tattr[1], frequency = tattr[3])
plot(reslm2, ylab = "Residuals", main = "From quadratic fit")
```



Alternatively one could fit a spline with the regularization parameter $\lambda = 1$, which ensures that a piecewise polynomial of first-order (line) is fit to the given series over short intervals of time.

```r
splinefit <- smooth.spline(time(vk), vk, spar = 1)
# Compute the residual and coerce it to a TS object
res_spfit <- vk - ts(splinefit$y, start = tattr[1], frequency = tattr[3])
# Plot the residual and compare with that from quadratic fit.
plot(res_spfit, ylab = "Residuals", main = "Residuals from spline and quadratic fits")
lines(reslm2, col = "red")
```

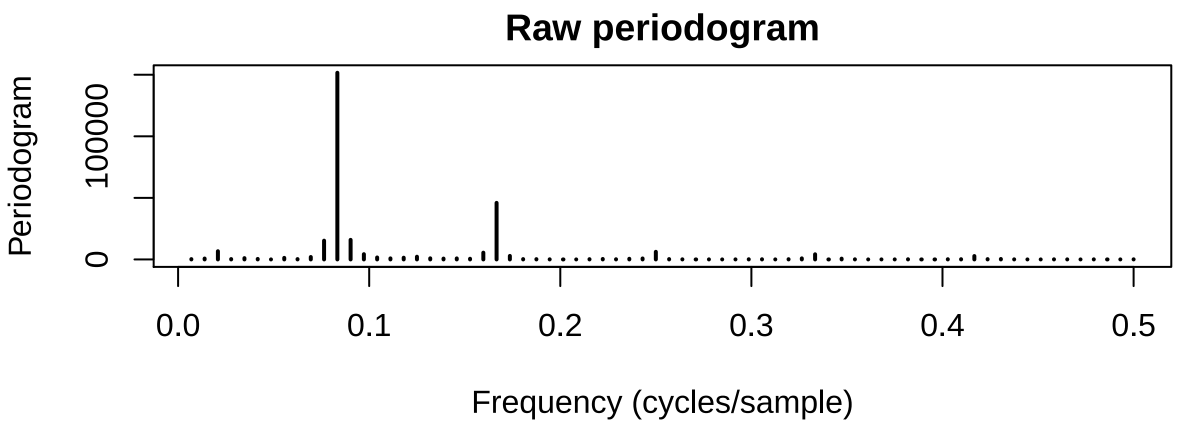


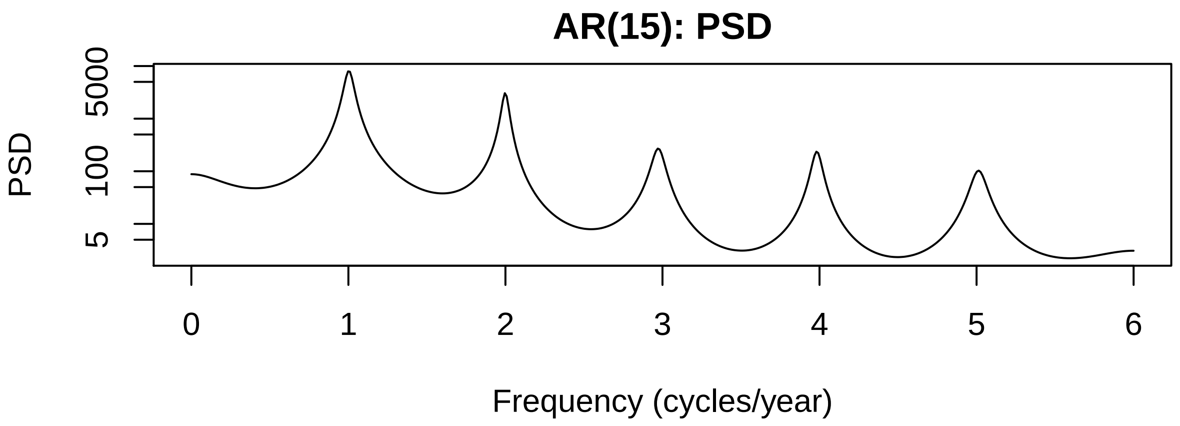
Observe that the residuals obtained from both approaches are nearly identical. We shall work with the residual series from the spline fit for the rest of the development. It is clear from the time-series plot that a seasonal component is present in the series, i.e., peaks at regular intervals in time. This is to be expected given the nature of the phenomenon.

Seasonality analysis

A seasonality of 12 (months) is noticed in the residual series. One can plot the periodogram or the spectral density to check for seasonality. These plots are shown in the figure.

```
.pardefault <- par(no.readonly = T)
par(mfrow = c(2, 1), mai = c(0.8, 1, 0.4, 0.2))
periodogram(res_spfit, ylab = "Periodogram", main = "Raw periodogram",
    xlab = "Frequency (cycles/sample)")
spec.ar(x = res_spfit, ylab = "PSD", main = "AR(15): PSD", xlab = "Frequency (cycles/year)")
```





The top panel shows the raw periodogram (obtained via FFT) using the `TSA` package, while the bottom panel is the *parametric* spectral density estimate obtained by fitting an AR(15) model. Both plots carry the trademark signature of seasonality, which is that of peaks in the PSD recurring at multiples of (1/12) cycles/sample or 1 cycle/year (observe the difference in the units of the x-axis of these plots).

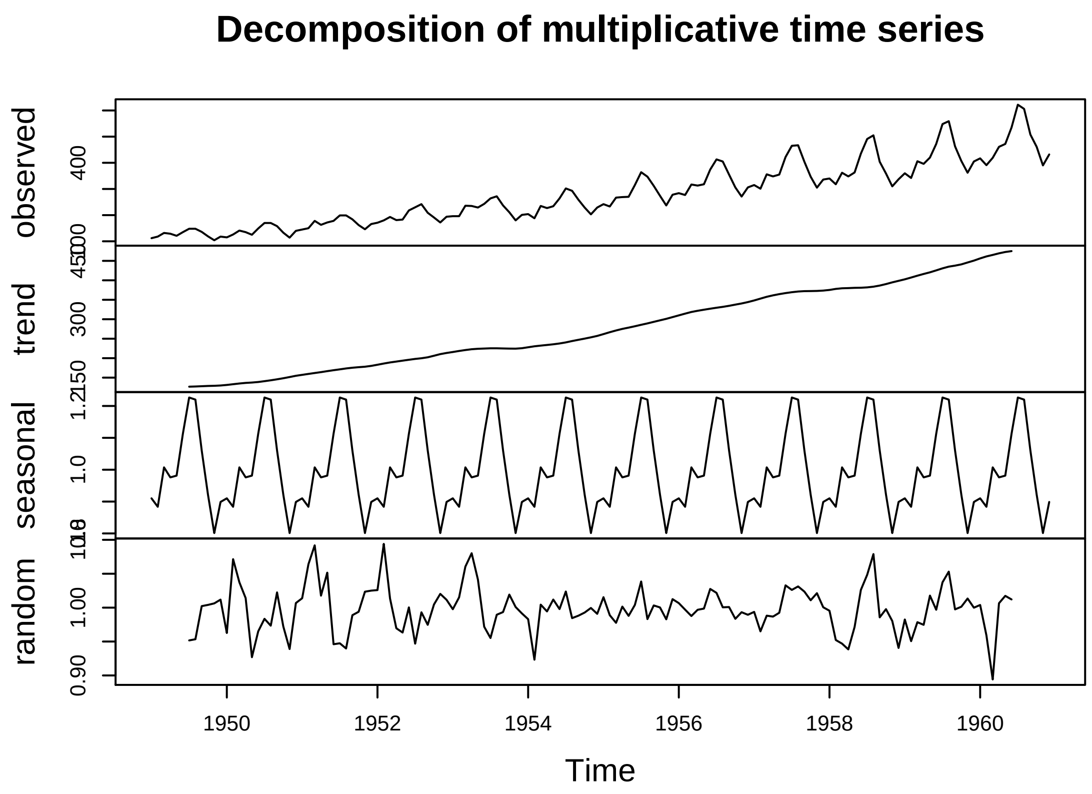
Extracting the seasonal, trend and irregular (level) components

Our next step is a decomposition of the series into its respective seasonal, trend and irregular components. The decision to be made at this stage is whether we seek an *additive* or *multiplicative* decomposition. By subtracting the trend from the series earlier, we have implicitly assumed the decomposition to be additive. However, if this were to be true, the amplitude of the seasonal component would not change with time, which is not the case with this series. Therefore, a **multiplicative** decomposition is more appropriate for the given series.

$$v[k] = S[k] * T[k] * w[k]$$

The decomposition can be done manually as we have already done for the trend $T[k]$. The seasonal part can be obtained by taking appropriate averages of the $v[k]/T[k]$ followed by the computation $v[k]/(T[k] * S[k])$ to obtain an estimate of the irregular component $w[k]$. Alternatively, one can turn to the `decompose` routine in R, which by default assumes a seasonality of 12 months (it suits our series!).

```
# Seasonality is retrieved by the routine from the frequency
# attribute
vkdec <- decompose(vk, type = "multiplicative")
plot(vkdec)
```

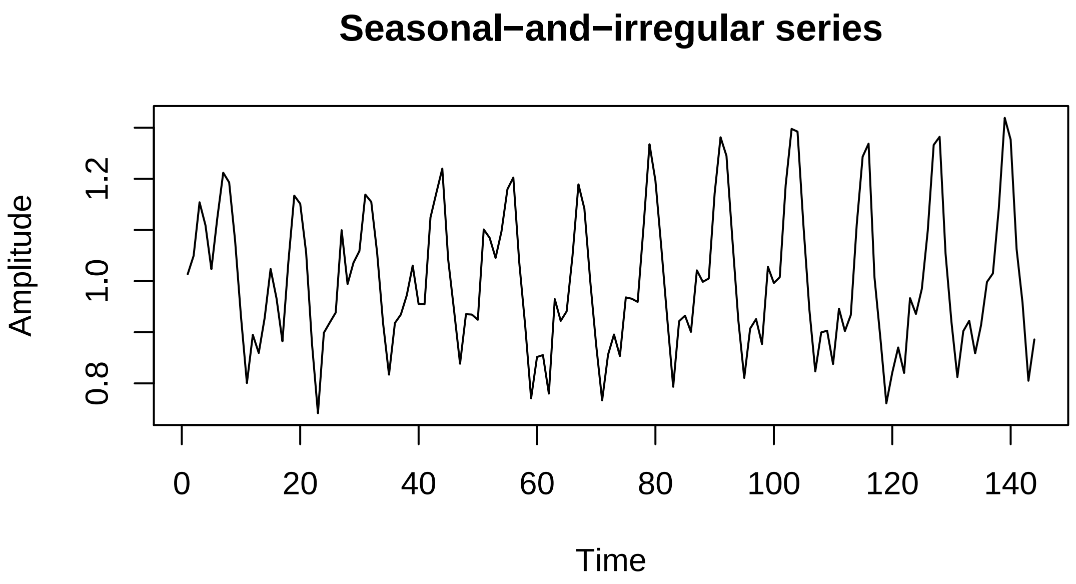


The decomposition has extracted the seasonal, trend and irregular component fairly well. We could now fit a model for the irregular component or fit a SARIMA model for the seasonal and irregular component. We shall adopt the latter strategy.

Fitting a SARIMA model for the trend-removed series

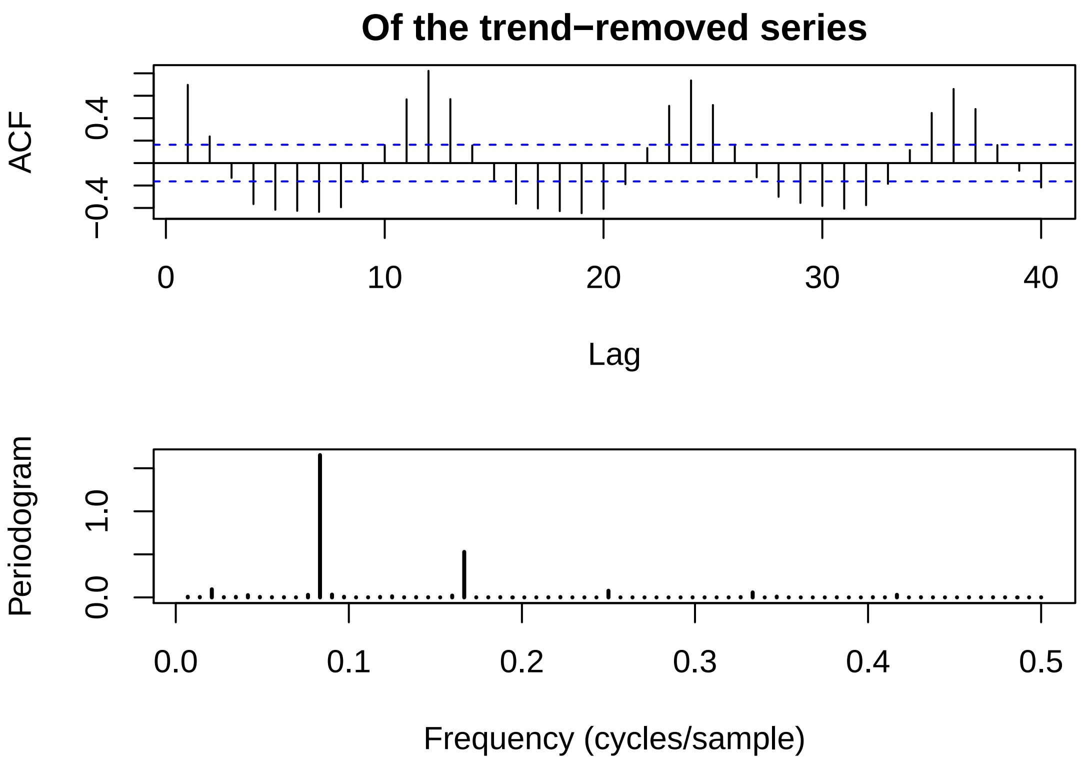
To begin with, we shall generate the seasonal and irregular component by dividing the original series with the trend that we fit earlier. Subsequently we shall fit a SARIMA model of appropriate order (after we have verified for any integrating effects).

```r
# Extract the series assuming multiplicative model
swk <- as.ts(as.vector(vk)/splinefit$y, start = tattr[1], frequency = tattr[3])
# Plot the series
plot(swk, xlab = "Time", ylab = "Amplitude", main = "Seasonal-and-irregular series")
```



Observe that the series has a non-zero mean. It may be a good idea to examine the ACF and periodogram of the obtained series before proceeding to model development.

```r
par(mfrow = c(2, 1), mai = c(0.8, 1, 0.4, 0.2))
acf(swk, lag.max = 40, main = "Of the trend-removed series")
periodogram(swk, ylab = "Periodogram", xlab = "Frequency (cycles/sample)")
```



It is clear from the plots that the series does not exhibit any integrating effects (conduct a formal test and check) and that the seasonality is of 12 months / cycle.

Before we proceed to developing the time-series model, it is useful to have a function that performs the routine diagnostics that one generates from an ARIMA model. There exists the `tsdiag` routine in the `stats` package, but it is useful to write a customized one so that we are fully aware of the proceedings. For this purpose, I have written a function named `mytsdiag.R`, the code for which is provided below.

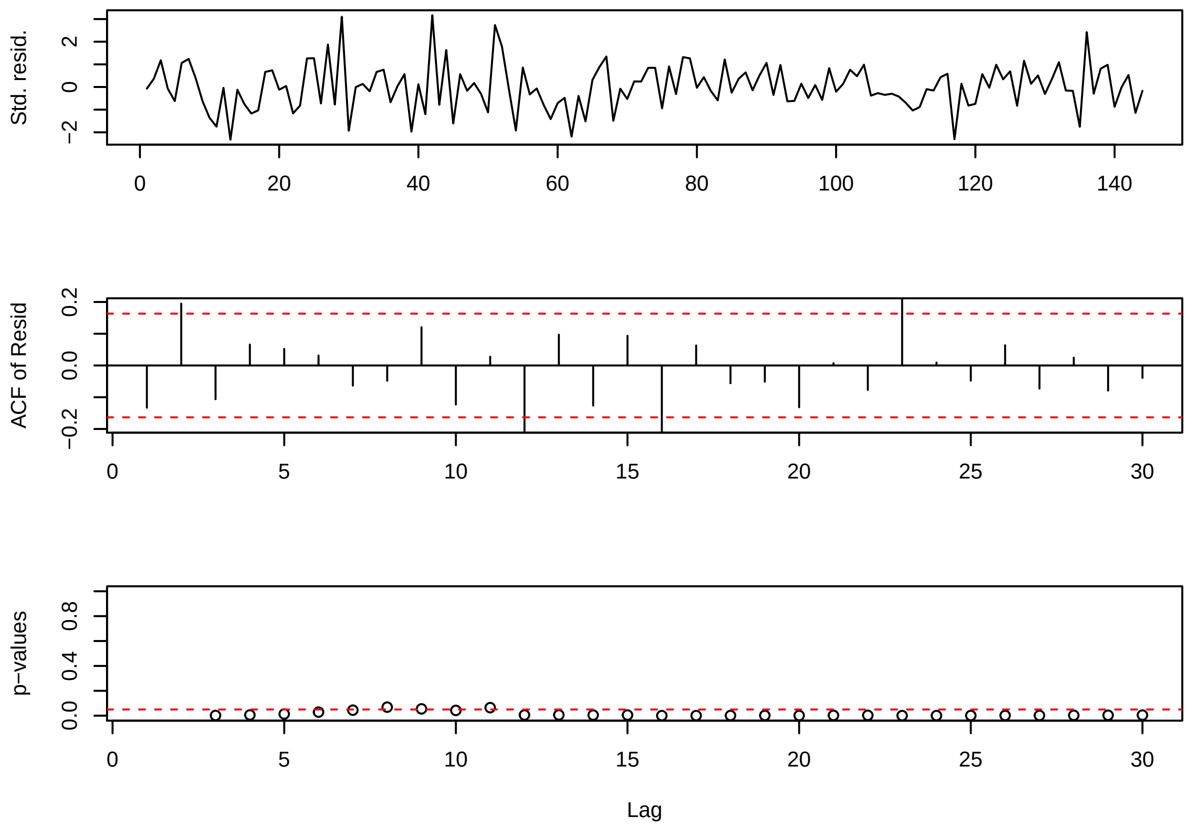
```r
mytsdiag <- function(modarima, Lmax = 30) {
    # Print summary
    summary(modarima)
    # Extract residuals and plot them
    err_mod <- modarima$residuals
    N = length(err_mod)
    # layout(matrix(c(1,2,3),3,1,byrow=TRUE),heights=c(1,1,1))
    par(mfrow = c(3, 1), mai = c(0.6, 0.7, 0.2, 0.2))
    plot(scale(err_mod), type = "l", ylab = "Std. resid.", xlab = "")
    # Compute ACF of residuals
    acf_err <- acf(err_mod, lag.max = Lmax, main = "", plot = F)
    lowsig = -1.96/sqrt(N)
    upsig = 1.96/sqrt(N)
    plot(acf_err$lag * tsp(err_mod)[3], acf_err$acf, type = "h",
        main = "", ylab = "ACF of Resid", xlab = "", ylim = c(1.2 *
            lowsig, 1.2 * upsig))
    abline(h = upsig, col = "red", lty = "dashed")
    abline(h = lowsig, col = "red", lty = "dashed")
    abline(h = 0, col = "black")
    # Compute BLP statistic
    blpval <- NULL
    npar <- sum(modarima$arma[1:4])
    Lval <- (npar + 1):Lmax
    for (L in Lval) {
        blpval <- c(blpval, Box.test(modarima$residuals, lag = L,
            fitdf = npar)$p.value)
    }
    # Plot BLP statistic
    plot(1:Lmax, c(rep(NA, npar), blpval), ylab = "p-values",
        xlab = "Lag", ylim = c(0, 1))
    abline(h = 0.05, col = "red", lty = "dashed")
}
```

We begin our hunt with a SARIMA model of first order in AR and seasonal components.

```r
modsarima <- stats::arima(swk, order = c(1, 0, 0), seasonal = list(order = c(1,
    0, 0), period = 12))
# Examine the diagnostics
mytsdiag(modsarima)
```

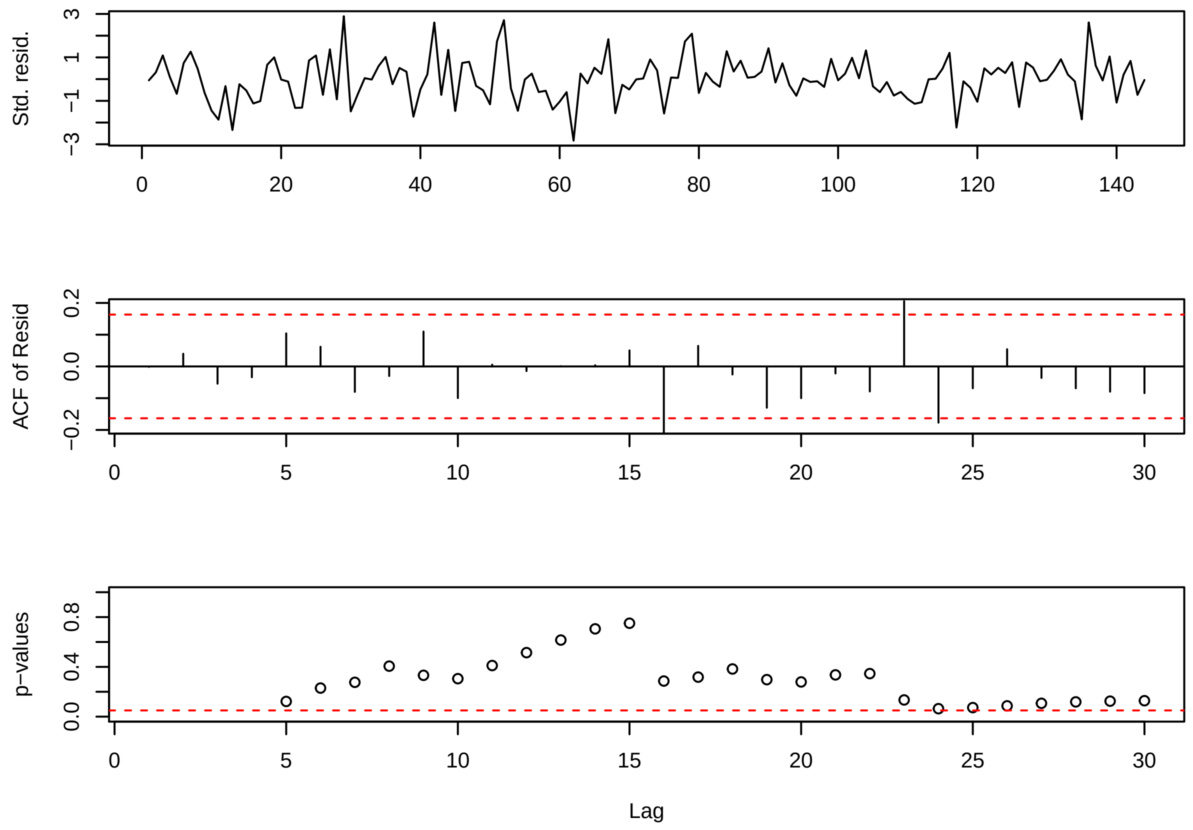
```
##
## Call:
## stats::arima(x = swk, order = c(1, 0, 0), seasonal = list(order = c(1, 0, 0),
##     period = 12))
##
## Coefficients:
##          ar1    sar1  intercept
##       0.7279  0.9079     1.0289
## s.e.  0.0579  0.0270     0.0822
##
## sigma^2 estimated as 0.00163:  log likelihood = 247,  aic = -486
##
## Training set error measures:
```

```
##                        ME       RMSE       MAE        MPE     MAPE
## Training set -0.001395622 0.04037893 0.0310575 -0.3354768 3.137096
##                   MASE       ACF1
## Training set 0.3478675 -0.1331768
```



```r
# Converge to a SARIMA(2,0,0)(2,0,0) model
modsarima <- stats::arima(swk, order = c(2, 0, 0), seasonal = list(order = c(2,
    0, 0), period = 12))
# Construct the diagnostics
mytsdiag(modsarima)
```

```
##
## Call:
## stats::arima(x = swk, order = c(2, 0, 0), seasonal = list(order = c(2, 0, 0),
##     period = 12))
##
## Coefficients:
##          ar1     ar2    sar1    sar2  intercept
##       0.5470  0.2480  0.5510  0.4113     1.0303
## s.e.  0.0825  0.0854  0.0808  0.0835     0.1446
##
## sigma^2 estimated as 0.001282:  log likelihood = 259.86,  aic = -507.72
##
## Training set error measures:
##                        ME       RMSE        MAE        MPE     MAPE
## Training set -0.001133149 0.03580943 0.02701732 -0.2864275 2.729358
##                   MASE        ACF1
## Training set 0.3026144 -0.00139632
```



Using the diagonstic tools, we arrive at a SARIMA(2,0,0)×(2,0,0) model. As the plots show, the model is satisfactory in all respects. The standard errors in the estimates imply that the coefficient estimates are significant. A formal check can be done by computing the confidence intervals.

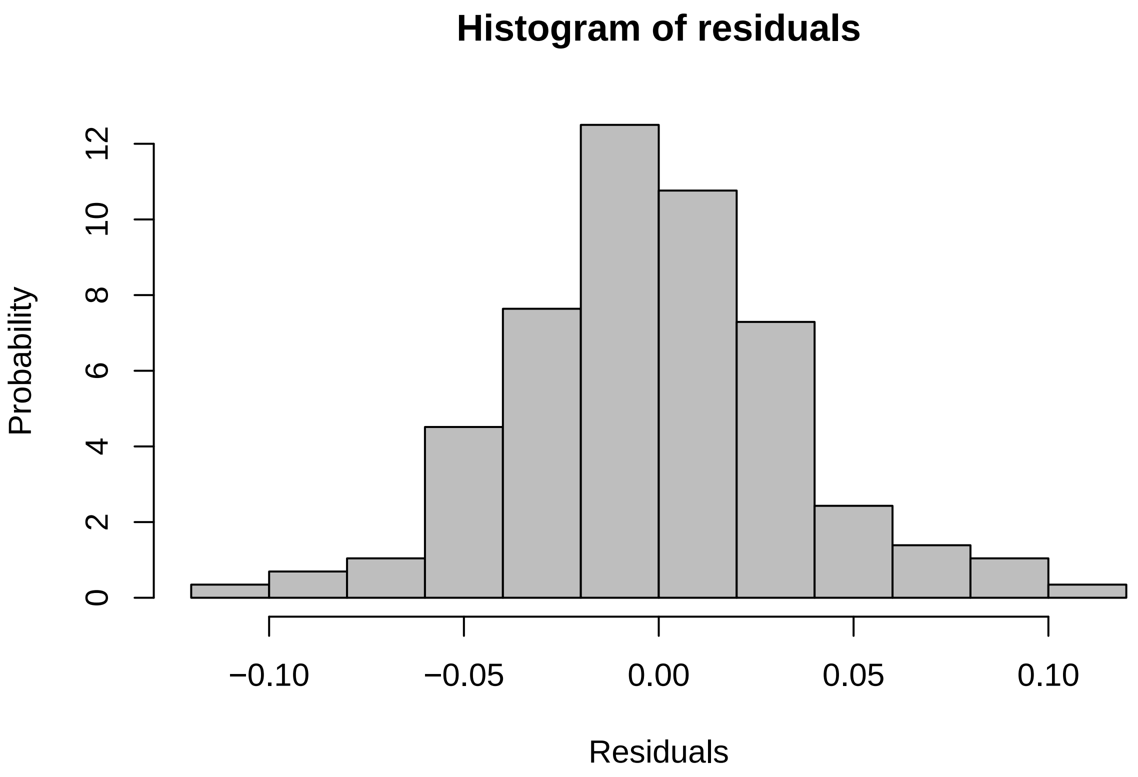
```
# Compute the confidence intervals
confint(modsarima)
```

```
##                 2.5 %    97.5 %
## ar1       0.38529641 0.7086132
## ar2       0.08062861 0.4152731
## sar1      0.39263344 0.7094550
## sar2      0.24760333 0.5749001
## intercept 0.74691461 1.3136390
```

The C.I. for model scoefficients are wide, obviously not a desirable result. However, given the small sample size, this is not unexpected. We should expect to see better estimates with larger samples.

To complete the model development we examine the histogram of errors obtained from the SARIMA model.

```
# Histogram of errors
hist(modsarima$residuals, probability = T, xlab = "Residuals",
    ylab = "Probability", main = "Histogram of residuals", col = "gray")
```



The plot strongly indicates Gaussian distributed errors with zero-mean. The variance of the errors is already estimated to be .

Thus the final model is

$$v[k] = T[k] * Sw[k]$$
$$Sw[k] = \left(\frac{1}{1 + d_1 q^{-1} + d_2 q^{-2}}\right)\left(\frac{1}{1 + d_{1,s} q^{-12} + d_{2,s} q^{-24}}\right) e[k], \quad \sigma_e^2 = 0.00128$$

with the coefficient estimates reported earlier.

Forecasting the series would involve computing the forecasts from the SARIMA model combined with that from the trend (using the associated `predict` routines).

As an **exercise** compute the 12-step ahead, i.e., yearly forecasts with this model.

Concluding remarks: Box and co-workers used natural logarithmic transformation and seasonal differencing to remove non-stationarity from the series. Following which they fit SARIMA$(0,1,1)(0,1,1)_{12}$ model, which according to them is the best stochastic model for the airline data.