Time Series Analysis - A Tutorial

Rosskopf,E.; Cordes, M.; Lumiko, J.

November 25, 2014

Abstract

Tutorial for time series analysis in R...

Contents

1 Introduction

This tutorial assumes that the reader has some basic knowledge of time series analysis, and the principal focus of the tutorial is not to explain time series analysis, but rather to explain how to carry out these analyses using R. If you are new to time series analysis, and want to learn more about any of the concepts presented here, We would highly recommend the Open University book "Time series" (product code M249/02), available from from the Open University Shop.

2 Getting started

2.1 Packages

Before we get started, please make sure to set a working directory and download the necessary packages listed below.

Useful packages for time series analysis:

```
> library(tseries)
> library(nlme)
> library(car)
> library(knitr)
> library(xtable)
> library(SweaveListingUtils)
> library(stats)
> library(forecast)
> library(AICcmodavg)
> library(TTR)
> library(mgcv)
```

2.2 Functions needed lateron

function writing to organize our script:

first we can write a diagnostics function with all the tests we need to perform to check if our model is adequate enough to stop the model adaptation.
we need to be careful if we want to check for residuals or the whole model. so x will be the model and xresidualsandxfitted are the other options we need.

```
> diagnostics <- function (x)
+   {
+   normality = shapiro.test(x$residuals); #check for normal distributed values #
+   stat.res =  adf.test(x$residuals); #check both residuals and fitted of the model for s
+   stat.fit = adf.test(x$fitted);
+   x$residualsvector = as.vector(x$residuals);
+  autocorr= dwt(x$residualsvector) ; #check for autocorrelation
+  indep=  Box.test(x$residuals, type="Ljung-Box") #check for independence
+   #lag for season is df: m-1 ( 12-1)
+   #write if seasonal = TRUE lag=12-1, else write nothing
+   #there is high evidence that there are non-zero autocorr.
+   output = list(normality, stat.res, stat.fit, autocorr, indep)
+   names (output) = c("norm. distrb. of residuals", "stationarity of residuals", "statior
+   return ( output )
+ }
```

Lateron for the forecast plotting, we need the histogram with the normal distribution to see wether the errors of the forecast model are well distributed:

```
> plotForecastErrors <- function(forecasterrors)
+ {
```

```
+   # make a histogram of the forecast errors:
+   mybinsize <- IQR(forecasterrors)/4
+   mysd   <- sd(forecasterrors)
+   mymin  <- min(forecasterrors) - mysd*5
+   mymax  <- max(forecasterrors) + mysd*3
+   # generate normally distributed data with mean 0 and standard deviation mysd
+   mynorm <- rnorm(10000, mean=0, sd=mysd)
+   mymin2 <- min(mynorm)
+   mymax2 <- max(mynorm)
+   if (mymin2 < mymin) { mymin <- mymin2 }
+   if (mymax2 > mymax) { mymax <- mymax2 }
+   # make a red histogram of the forecast errors, with the normally distributed data over
+   mybins <- seq(mymin, mymax, mybinsize)
+   hist(forecasterrors, col="red", freq=FALSE, breaks=mybins)
+   # freq=FALSE ensures density
+   # generate normally distributed data with mean 0 and standard deviation mysd
+   myhist <- hist(mynorm, plot=FALSE, breaks=mybins)
+   # plot the normal curve as a blue line on top of the histogram of forecast errors:
+   points(myhist$mids, myhist$density, type="l", col="blue", lwd=2)
+ }
```

2.3 Dataset (CO2-Concentrations)

The first dataset we will work with consists of monthly CO2-concentrations [ppm] in the atmosphere, measured over time at the famous Mauna Loa Station on Hawaii.
To download this dataset, just use the code provided below.

```
> url<-"ftp://aftp.cmdl.noaa.gov/products/trends/co2/co2_mm_mlo.txt"
> dest<-"C:/Users/schnuri/Desktop/Neuer Ordner/Dataset/run.txt"
> download.file(url, dest )
> co2month=read.table(dest, skip=72)
> co2month
```

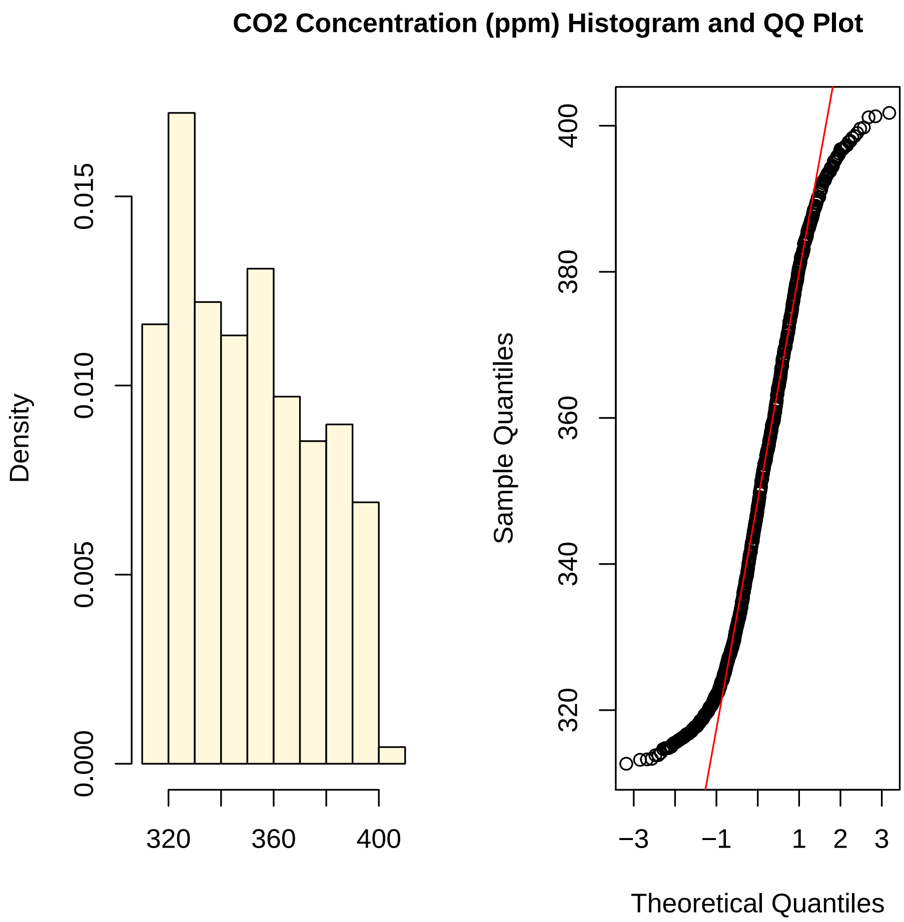
Note:"dest" represents a randomly chosen name for a text file in which the CO2-dataset will be saved. Feel free to adjust the name and directory.

2.4 Dataset Visualization

It can be really useful to visualize your dataset before you transform it into a timeseries (ts) in order to detect potential errors.

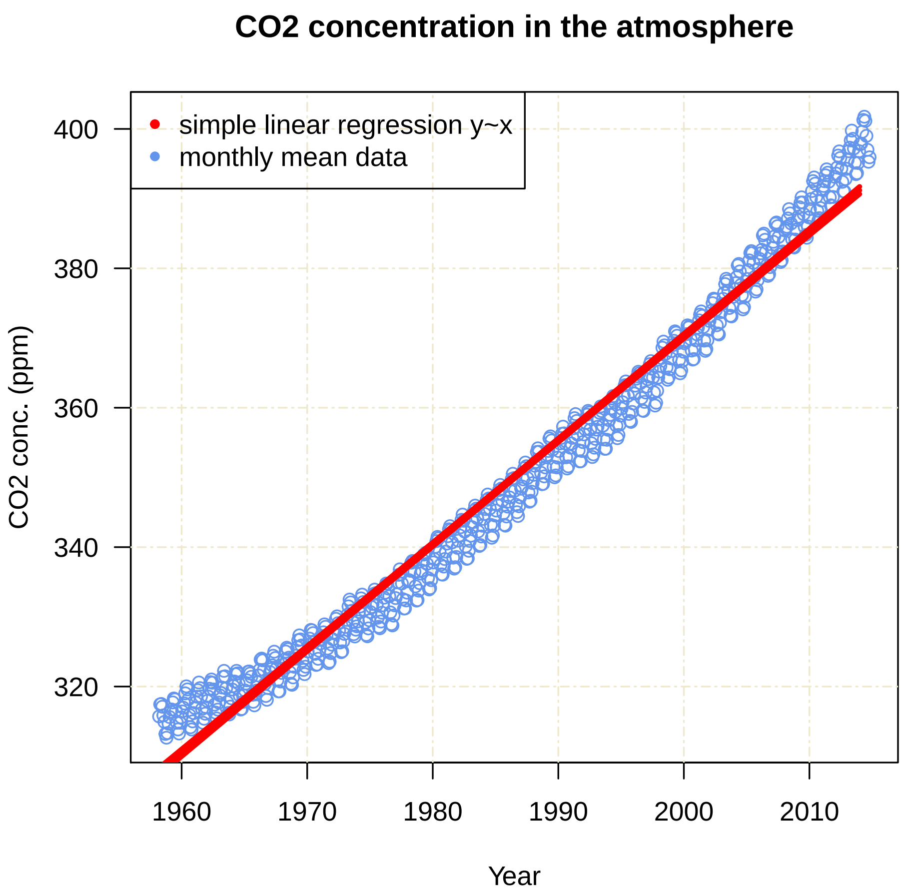
2.4.1 Histogram & QQ-Plot

```
> data = co2month[,c(3,5)]
> colnames(data)= c("year", "co2")
> attach(data)
> x = co2
> op = par(mfrow = c(1,2),
+          mar = c(5,4,1,2)+.1,
+          oma = c(0,0,2,0))
> hist(co2, freq=F, col = "cornsilk",xlab = "", main = "")
> qqnorm(x, main = ""); qqline(x,col = 'red')
> par(op)
> mtext("CO2 Concentration (ppm) Histogram and QQ Plot", line = 2.5,font = 2,cex = 1.0)
```


CO2 Concentration (ppm) Histogram and QQ Plot
Density
0.000
0.005
0.010
0.015
320
360
400
Sample Quantiles
Theoretical Quantiles
−3
−1
1
2
3
320
340
360
380
400

2.4.2 Plotting the fitted values

```
> # Run a linear model
> datalm = lm( co2 ~ year)
> # Fit predict values
> MyData=data.frame(year=seq(from=(1958),to=2014, by=0.1))
> pred=predict(datalm, newdata=MyData, type="response", se=T)
> # Plot the fitted values
> plot(year, co2, type="n",las=1, xlab="Year", ylab="CO2 conc. (ppm)", main="CO2 concer
> grid (NULL,NULL, lty = 6, col = "cornsilk2")
> points(year, co2, col="cornflowerblue" )
> # Write confidence interval
> F=(pred$fit)
> FSUP=(pred$fit+1.96*pred$se.fit) # make upper conf. int.
> FSLOW=(pred$fit-1.96*pred$se.fit) # make lower conf. int.
> lines(MyData$year, F, lty=1, col="red", lwd=3)
> lines(MyData$year, FSUP,lty=1, col="red", lwd=3)
> lines(MyData$year, FSLOW,lty=1, col="red", lwd=3)
> legend("topleft",c("simple linear regression y~x", "monthly mean data"),
+        pch=c(20,20), col=c("red", "cornflowerblue"))
```



Look at standard errors, highly underestimated, standard errors are higher in true ! p value is not right evidence , misleading :

```
> xtable(summary(datalm))
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-2618.5566	16.9782	-154.23	0.0000
year	1.4944	0.0085	174.86	0.0000

This plot (**??** can be used to observe if there are outliers which could possibly bias the model.
However the poly-1 linear regression is not accurate in fitting the CO2-dataset. This is due to the present autocorrelation that not yet has been taken into account. Neglecting this factor will always effect the accuracy of the model results. The standard errors are lower than their true values thus giving high statistical significance with a p-value lower than it should be. The clue in statistical modelling is to present the correct statistical evidence, which would be highly biased with a linear model.

2.5 Dataset Transformation

It is essential to transform your dataset into a timeseries (ts) if you seek for an accurate and extensive analysis of the data.
The data stored as a dataframe needs to be transformed with the important columns into the class of a time series to continue working on it properly. If you have monthly data you have to set the deltat of the function ts() to deltat=1/12 describing the sampling period parts between successive values xt and xt+1. Your time series should somehow look like table 1.

Original Data

```
> xtable(head(data), caption="Original CO2-Data")
```

	year	co2
1	1958.21	315.71
2	1958.29	317.45
3	1958.38	317.50
4	1958.46	317.10
5	1958.54	315.86
6	1958.62	314.93

Table 1: Original CO2-Data

Transformation

```
> yourts=ts(co2, c(1958,3),c(2014,10), deltat=1/12)
> class(yourts)
```

[1] ”ts”

2.6 Time-Series Visualization

It is important to get a quick overview of your data. Some simple plots for visualization are quite helpful.

2.6.1 Time-Series Plot

```
> par(mfrow=c(1,1))
> plot.ts(yourts,las=1, xlab="Year", ylab="CO2 conc. (ppm)", main="CO2 concentration in
> grid (NULL,NULL, lty = 6, col = "cornsilk2")
> points(yourts, col="cornflowerblue" )
> k <- 5
> lines(year,filter(co2, rep(1/k,k) ),col = 'red', type="l", lwd = 3)
> legend("topleft",c("simple moving average", "monthly mean data"),
+ pch=c(20,20), col=c("red", "cornflowerblue"))
```

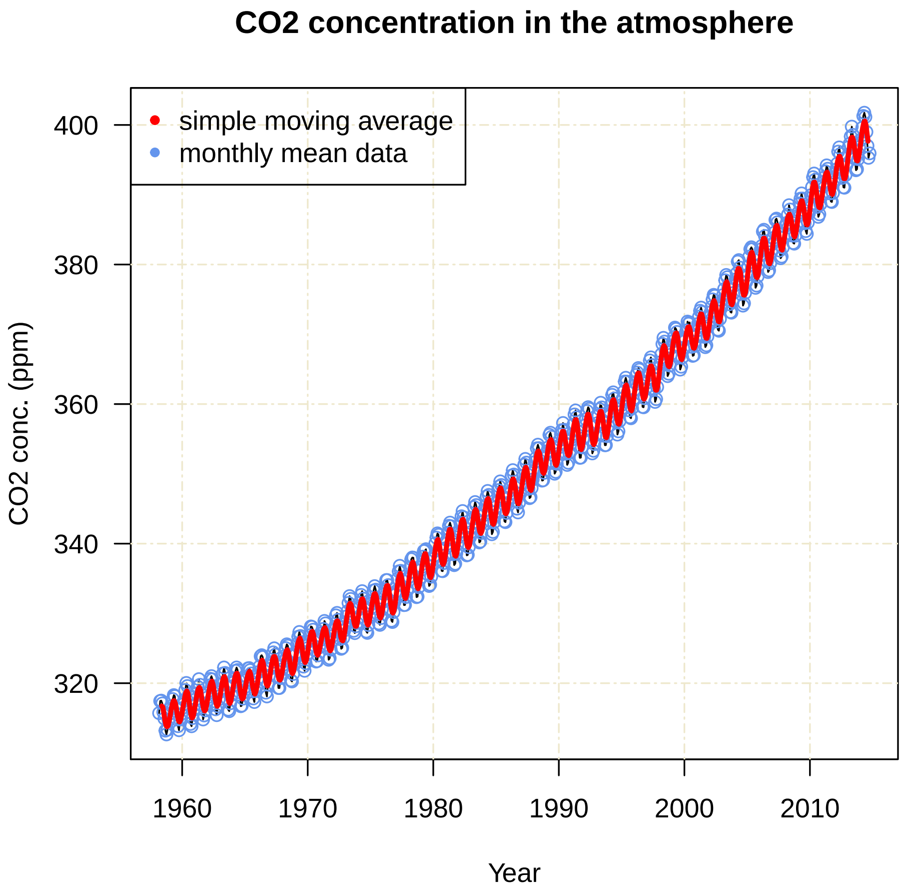


Figure 1: Visualization of the CO2 Concentrations

Note: The red line in plot 1 was computed with a simple moving average. It is not enough to just run a MA.

2.6.2 ACF, PACF, SPECTRUM

Since time-series data usually violates the independence assumption of the model, the standard error is potentially too small. As the data are regularly spaced in time, we can easily employ the autocorrelation function to investigate residuals correlations in the model errors. **The acf() function** can be used for that, which produces a plot of the correlogram.
Another nice procedure is to run the **autocorrelation function** with its complementary **partial acf** and the **spectrum** showing the spectral density of your time series at frequencies corresponding to the possibly approx. Fourier frequencies.

```
> op <- par(mfrow = c(3,1),
+           mar = c(2,4,1,2)+.1,
+           oma = c(0,0,2,0))
> acf(x, xlab = "")
> pacf(x, xlab = "")
> spectrum(x, xlab = "", main = "")
> par(op)
> mtext("CO2 Concentration (ppm) correlogram",
+       line = 2.5,
+       font = 2,
+       cex = 0.8)
> op <- par(mfrow = c(3,1),
+           mar = c(2,4,1,2)+.1,
+           oma = c(0,0,2,0))
> acf(resid(datalm), xlab = "")
> pacf(resid(datalm),xlab = "")
> spectrum(resid(datalm), xlab = "", main = "")
> par(op)
> mtext("Model residual correlogram",
+       line = 2.5,
+       font = 2,
+       cex = 1.2)
```

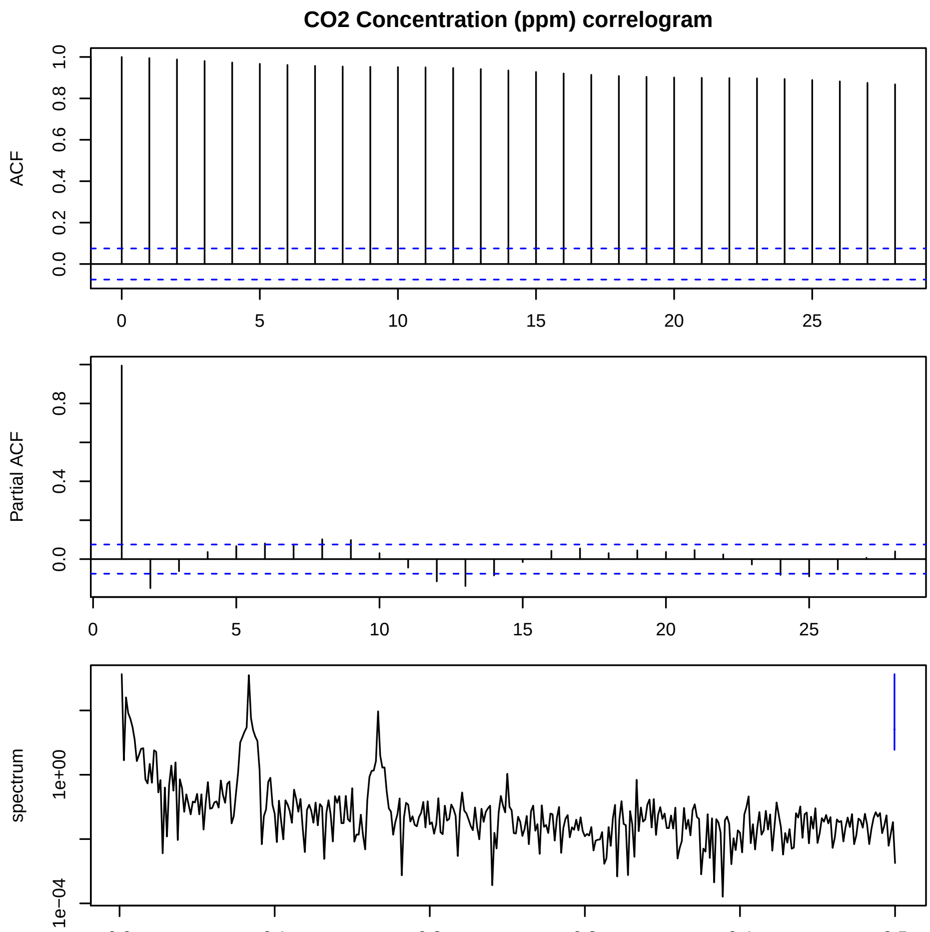


Figure 2: Correlogram of time series and residuals of lm

Explain the acf , pacf, spectrum here.
ACF & PACF:
The generated correlogram reveals that there are major autocorrelations.
There is a strong correlation at lag 1,a weaker correlation at lag 2, and a noticeable correlation at lag 3. Such a correlation pattern is typical for an autoregressive process where most of the sequential dependence can be explained as a flow-on effect from a dependence at lag 1.

In an autoregressive time series, an independent error component, or "innovation" is associated with each time point. For an order p autoregressive time series, the error for any time point is obtained by taking the innovation for that time point, and adding a linear combination of the innovations at the p previous time points. (For the present time series, initial indications are that p = 1 might capture most of the correlation structure.)"
(autosmooth.pdf)

Spectrum: easier to interpret the acf / log scaled / strong cycles where spectrum max. occurs Here the highest maximum is at about 0.75. 1/0.75 = 12 meaning a 12 month cycle is occuring here .

The residuals in a time series are serially correlated. The ACF is waving and decreases only slowly, which could be an identification of non-stationarity (If the ACF would drop to zero quickly, the time series would be stationary). We stop all diagnostics here for our clearly wrong model and go on to investigate the different components of our time series.

3 Decomposition of Time Series

A time serie consists of 3 components; a trend component, an irregular (random) component and (if it is a seosonal time series) seasonal component.

3.1 Decomposing Seasonal Data

We can decompose the ts and plot these components:

```
> plot(decompose(yourts))
>
```

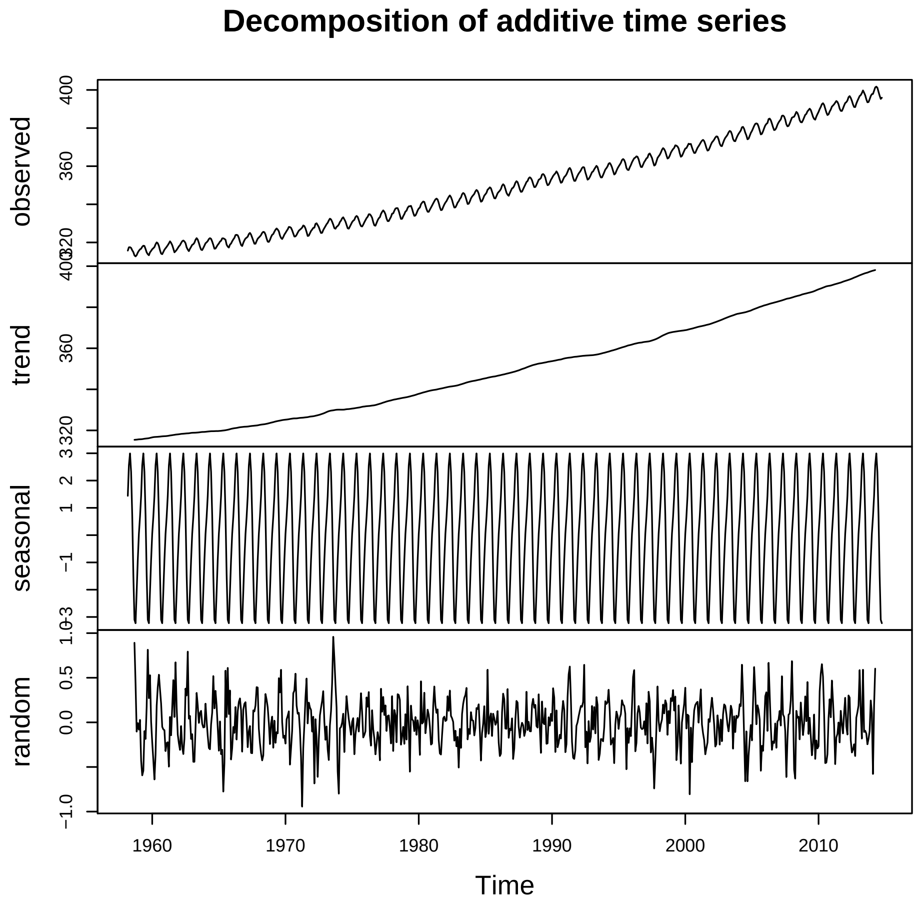


Figure 3: Decomposition of the CO2 Time Series

We can see each component with:

```
> yourts_components<- decompose(yourts)
```

```
> yourts_components$seasonal
```

```
> #we can see the trend for the first year:
> par(mfrow=c(1,2))
> ts.plot(yourts_components$seasonal[1:12])
> ts.plot(aggregate(yourts_components$seasonal))
>
>
```

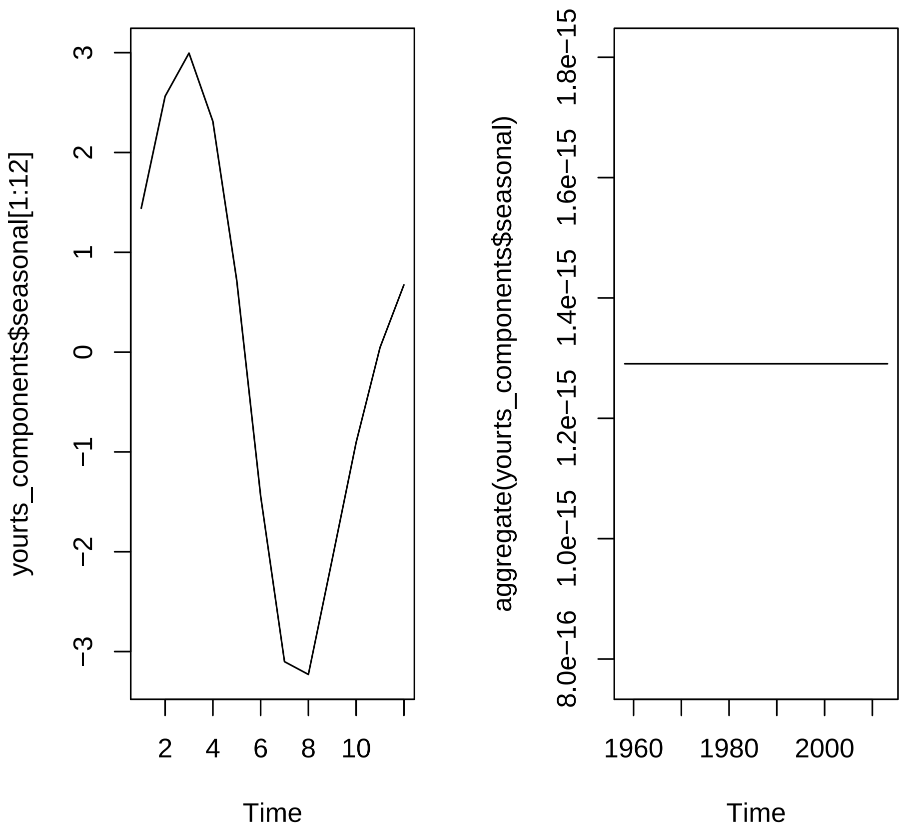


Figure 4: The seasonal component across the time

It seems that our seasonal component is positive until the summer months, were it turns to be negative and turning to be positiv again in winter (see fig. 4). And we can see in the right plot that this seasonal component is constant over all the years (see fig. 4).

```
> yourts_seasonallyadjusted <- yourts - yourts_components$seasonal
> par(mfrow=c(1,2))
> plot(yourts, main="TS with seasonal fl.", las=1)
> plot(yourts_seasonallyadjusted, las=1, main="removed seasonal fluctuation")
>
```

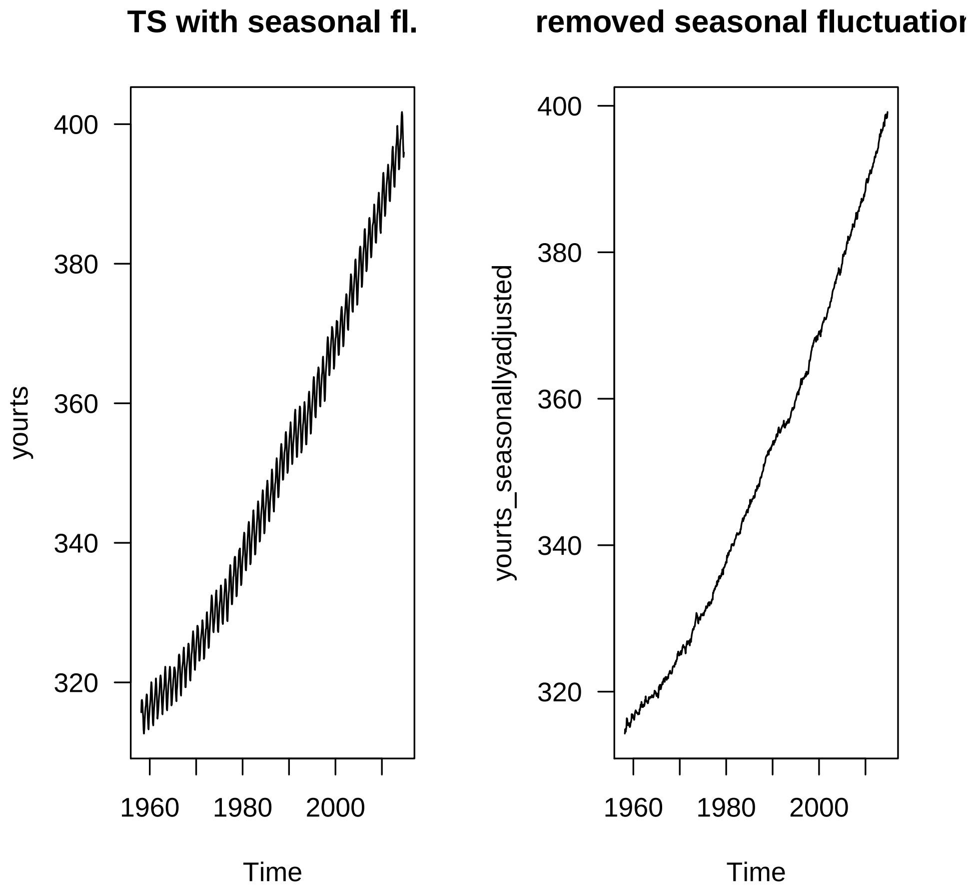


Figure 5: Comparison of seasonal vs. seasonally adjusted model

It seems that our data can probably be described using an additive model, since the random fluctuations in the data are roughly constant in size over time (constant seasonal component). In some cases it might be handy to have the model without the seasonal fluctuations to depict change in the trend and local extremes easier (see fig. 5).

4 Analysis of Seasonal Data

After looking at the simple linear regression datalm, we were facing some serious problems with our model. To be sure about non-stationarity of our time series, wen can run the adf.test, giving us the result that our data is non-stationary and we need to fix it:

```
        Augmented Dickey-Fuller Test

data:  yourts
Dickey-Fuller = -1.0663, Lag order = 8, p-value = 0.928
alternative hypothesis: stationary
```

Also we need a model which is covering the serial correlation of our residuals. The ACF, PACF, spectrum above gives us certainty an the autocorrelation and the seasonality. The standard errors are highly underestimated, thus in the summary the p-value is too small and misleading.

A nice model to try is GLS, which will allow for correlation of standard errors and unequel variances.

In the GLS we have different options to choose, though our data is not spatially correlated we are not discussing spatial autocorrelation here (further reading on:...)

Our first try on gls will be simple:

```
> data.glsAR = gls(co2 ~ year,cor= corAR1(acf(resid(datalm))$acf[2]))
>
```

The difference will be made in the correlation structure. There are generally (for temporal corr. interesting) five options you have:

1. corAR1: in ACF exponential decreasing values of correlation with time distance

2. corARMA: either autoregressive order or moving average order or both

3. corCAR1: continuous time (time index does nto have to be integers)

4. corCompSymm: correlation does not decrease with higher distance

5. corSymm: general correlation only for few observations only, often overparameterized

Our first gls model accounts for the AR1, which is clearly visible in the PACF.

```
> acf(data.glsAR$residuals)
```

We have still a lot of problems concerning the autocorrelation and the seasonality. One option is to allow the AR to use more parameters and/or to include a moving average or error variance to the model. This can be handled via the corARMA. We tried 2 versions, one with 1 lag and 1 moving average, the other with 2 lags and 2 moving averages. The 0.2 are starting values for Phi, which are in the modelling process optimized.

The next models are thus:

```
> data.glsARMA1 = gls ( co2 ~ year, cor = corARMA (c(0.2,0.2),p=1, q=1 ))
> data.glsARMA2 = gls (co2 ~year, cor=corARMA(c(0.2,0.2,0.2, 0.2), p=2, q=2))
> xtable(anova(data.glsAR, data.glsARMA1, data.glsARMA2))
```

	call
data.glsAR	gls(model = co2 ~ year, correlation = corAR1(acf(resid(datalm))$acf[2]))
data.glsARMA1	gls(model = co2 ~ year, correlation = corARMA(c(0.2, 0.2), p = 1, q = 1))
data.glsARMA2	gls(model = co2 ~ year, correlation = corARMA(c(0.2, 0.2, 0.2, 0.2), p = 2, q = 2))

To compare all the models we use anova and the best model is so far the data.glsARMA2 with the lowest AIC and significantly better than the ARMA1, which is itself significantly better than the data.glsAR. ‘$norm.distrb.ofresiduals$‘

Shapiro-Wilk normality test

data: x$residualsW = 0.9785, p - value = 1.861e - 08$

‘$stationarityofresiduals$‘

Augmented Dickey-Fuller Test

data: x$residualsDickey{-}Fuller = -1.0665, Lagorder = 8, p{-}value = 0.928alternativehypothesis :$ $stationary$

‘$stationarityoffittedvalues$‘

Augmented Dickey-Fuller Test

data: x$fittedDickey{-}Fuller = -72220583361, Lagorder = 8, p{-}value = 0.01alternativehypothesis :$ $stationary$

‘$autocorrelationofresiduals$‘[1]0.1111791

‘$independenceofresiduals$‘

Box-Ljung test

data: $xresidualsX - squared = 603.7779, \mathrm{df} = 1, p - value < 2.2e - 16$

function writing: if anova$AIClowest, choosethisasbestmodelButstillwehaveseasonalproblemshere.Wesho$

```
> seas = cycle(yourts)
> dataseason.gls = gls(co2 ~ year + factor(seas), cor=corARMA( c(0.2,0.2,0.2, 0.2),p=2, q=
\end{Sinput}
\end{Schunk}




\subsection{Decomposing Non-Seasonal Data (Annual Flow Of The River Nile)}

Our second dataset contains the measurements of the annual flow of the river Nile at Ashwa

\noindent First we visualize our data:\\
\begin{figure}[H]
\centering
\begin{Schunk}
\begin{Sinput}
> str(Nile)
\end{Sinput}
\begin{Soutput}
 Time-Series [1:100] from 1871 to 1970: 1120 1160 963 1210 1160 1160 813 1230 1370 1140 ..
\end{Soutput}
\begin{Sinput}
> plot(Nile, main="Annual flow of teh Nile", ylab="Flow [V/a]")
\end{Sinput}
\end{Schunk}
\caption{Annual Flow of the Nile}
\end{figure}

\noindent A non-seasonal time series consists of a trend and an irregular component. To es

\noindent The SMA() function in the ■TTR■ R package can be used to smooth time series data
\begin{figure}[H]
\centering
\begin{Schunk}
\begin{Sinput}
> library(TTR)
> plot(SMA(Nile,n=20))
\end{Sinput}
\end{Schunk}
\includegraphics{sweaveclean-025}
\caption{Trend of the Annual Flow of the Nile}
\end{figure}

\noindent We can see that there was a negative trend until 1920 and that it becomes positi


\noindent Now we can procede plotting the correlograms:
\begin{figure}[H]
\centering
\begin{Schunk}
\begin{Sinput}
> par(mfrow = c(1, 2))
> acf(Nile)
```

```
> pacf(Nile)
> par(mfrow = c(1, 1))
\end{Sinput}
\end{Schunk}
\includegraphics{sweaveclean-026}
\caption{Autocorrelations and Partial autocorrelations.}
\end{figure}

\noindent We find a significant autocorrelation at the first lags only. This means that th
The  partial autocorrelation plot does not show any significant autocorrelation

\section{Modelling the time series}
The best model for a time series needs to have residuals as white noise terms.
There are different ways to approach this task.


\section{Forecasts}%---------------------------------------------------------------------
We have three different options to make ( up to now)
\begin{enumerate}
  \item predict
  \item Holt Winters
  \item Arima forecasts
\end{enumerate}


\subsection{Holt-Winters Exponential Smoothing}%-----------------------------------------

If we have a time series that can be described using an additive model,we can short-time f
Preconditions:forecast errors are uncorrelated and are normally distributed with mean zero
\begin{Schunk}
\begin{Sinput}
> hw<-HoltWinters(yourts)
> #the alpha value tells us the weight of the previous values for the forecasting
> #values of alpha that are close to 0 mean that little weight is placed on the most recen
> #gamma is for the seasonality
> plot(hw)
>
\end{Sinput}
\end{Schunk}

\noindent Holtwinters just makes forecasts for the time period covered by the original dat

\begin{figure}[H]
\centering
\begin{Schunk}
\begin{Sinput}
> hw1<- forecast.HoltWinters(hw, h=12)
> #for the next year
> plot.forecast(hw1, main="Prediction for the next year",shadecols = "oldstyle")
> #for next 10 years
> hw10<- forecast.HoltWinters(hw, h=120)
> plot.forecast(hw10, main="Prediction for the next 10 years", shadecols = "oldstyle")
\end{Sinput}
\end{Schunk}
\includegraphics{sweaveclean-028}
\caption{Forecasting using Holt Winters exponential smoothing}
\end{figure}

\noindent Here the forecasts for 1913-1920 are plotted as a blue line, the 80% prediction
```

```
\noindent The ‘forecast errors’ are calculated as the observed values minus the predicted

\noindent To calculate a correlogram of the in-sample forecast errors for the CO2 Time ser

\begin{figure}[H]
\centering
\begin{Schunk}
\begin{Sinput}
> acf(hw10$residuals, lag.max=20)
\end{Sinput}
\end{Schunk}
\includegraphics{sweaveclean-029}
\caption{Correlogram of the residuals.}
\end{figure}

\begin{Schunk}
\begin{Sinput}
> Box.test(hw10$residuals, lag=20, type="Ljung-Box")
\end{Sinput}
\begin{Soutput}
        Box-Ljung test

data:  hw10$residuals
X-squared = 37.5445, df = 20, p-value = 0.01006
\end{Soutput}
\end{Schunk}
There is little evidence of non-zero autocorrelations at lags 1-20.

\begin{Schunk}
\begin{Sinput}
> plotForecastErrors <- function(forecasterrors)
+   {
+      # make a histogram of the forecast errors:
+      mybinsize <- IQR(forecasterrors)/4
+      mysd   <- sd(forecasterrors)
+      mymin  <- min(forecasterrors) - mysd*5
+      mymax  <- max(forecasterrors) + mysd*3
+      # generate normally distributed data with mean 0 and standard deviation mysd
+      mynorm <- rnorm(10000, mean=0, sd=mysd)
+      mymin2 <- min(mynorm)
+      mymax2 <- max(mynorm)
+      if (mymin2 < mymin) { mymin <- mymin2 }
+      if (mymax2 > mymax) { mymax <- mymax2 }
+      # make a red histogram of the forecast errors, with the normally distributed data o
+      mybins <- seq(mymin, mymax, mybinsize)
+      hist(forecasterrors, col="red", freq=FALSE, breaks=mybins)
+      # freq=FALSE ensures the area under the histogram = 1
+      # generate normally distributed data with mean 0 and standard deviation mysd
+      myhist <- hist(mynorm, plot=FALSE, breaks=mybins)
+      # plot the normal curve as a blue line on top of the histogram of forecast errors:
+      points(myhist$mids, myhist$density, type="l", col="blue", lwd=2)
+   }
\end{Sinput}
\end{Schunk}


\begin{figure}[H]
\centering
```

```
\begin{Schunk}
\begin{Sinput}
> plotForecastErrors(hw10$residuals)
\end{Sinput}
\end{Schunk}
\caption{Histogram of the errors}
\end{figure}

\noindent The histogram of the time series shows that the forecast errors are roughly norm


\subsection{Seasonal Decomposition of Time Series by Loess}
Forecasting using stl objects:
\begin{figure}[H]
\centering
\begin{Schunk}
\begin{Sinput}
> plot(stlf(yourts, lambda=0, h =120))
> (tslm(yourts~time(yourts)))
\end{Sinput}
\begin{Soutput}
Call:
lm(formula = formula, data = "yourts", na.action = na.exclude)

Coefficients:
 (Intercept)  time(yourts)
   -2618.494         1.494
\end{Soutput}
\end{Schunk}
\includegraphics{sweaveclean-033}
\caption{Forecasting using Loess}
\end{figure}


\subsection{ARIMA Models}%-----------------------------------------------------------------
For the CO2 time series:
\begin{Schunk}
\begin{Sinput}
> au=auto.arima(yourts, ic = "bic")
> arima1=arima(yourts, order = c(au$arma[1],au$arma[6],au$arma[2]))
> fore1=forecast.Arima(arima1,h=10)
> plot.forecast(fore1)
>
\end{Sinput}
\end{Schunk}
\includegraphics{sweaveclean-034}

We can first fit an autoregression model to the Nile Time Series:
\begin{figure}[H]
\centering
\begin{Schunk}
\begin{Sinput}
> ar(Nile) # selects order 2
\end{Sinput}
\begin{Soutput}
Call:
ar(x = Nile)

Coefficients:
```

```
     1       2
0.4081  0.1812

Order selected 2  sigma^2 estimated as  21247
\end{Soutput}
\begin{Sinput}
> cpgram(ar(Nile)$resid)
> arima(Nile, c(2, 0, 0))
\end{Sinput}
\begin{Soutput}
Series: Nile
ARIMA(2,0,0) with non-zero mean

Coefficients:
         ar1     ar2  intercept
      0.4096  0.1987   919.8397
s.e.  0.0974  0.0990    35.6410

sigma^2 estimated as 20291:  log likelihood=-637.98
AIC=1283.96   AICc=1284.38   BIC=1294.38
\end{Soutput}
\begin{Sinput}
>
\end{Sinput}
\end{Schunk}
\includegraphics{sweaveclean-035}
\caption{Cumulative Peridiogram of the residuals}
\end{figure}
Fitting a autoregressive model, we can see that the residuals are well placed.

Structural time series models:
\begin{figure}[H]
\centering
\begin{Schunk}
\begin{Sinput}
> par(mfrow = c(3, 1))
> plot(Nile)
> ## local level model
> (fit <- StructTS(Nile, type = "level"))
\end{Sinput}
\begin{Soutput}
Call:
StructTS(x = Nile, type = "level")

Variances:
  level  epsilon
   1469    15099
\end{Soutput}
\begin{Sinput}
> lines(fitted(fit), lty = 2)              # contemporaneous smoothing
> lines(tsSmooth(fit), lty = 2, col = 4)   # fixed-interval smoothing
> plot(residuals(fit)); abline(h = 0, lty = 3)
> ## local trend model
> (fit2 <- StructTS(Nile, type = "trend")) ## constant trend fitted
\end{Sinput}
\begin{Soutput}
Call:
StructTS(x = Nile, type = "trend")
```

```
Variances:
  level    slope  epsilon
   1427        0    15047
\end{Soutput}
\begin{Sinput}
> pred <- predict(fit, n.ahead = 30)
> ## with 50% confidence interval
> ts.plot(Nile, pred$pred,
+         pred$pred + 0.67*pred$se, pred$pred -0.67*pred$se, col=1:3)
>
>
\end{Sinput}
\end{Schunk}
\includegraphics{sweaveclean-036}
\caption{Fitting a Structural Model}
\end{figure}


\subsection{Selecting a Candidate ARIMA Model}%-----------------------------------------

TEXT

\subsection{Forecasting Using an ARIMA Model}%------------------------------------------

compare both functions of forecasting:
\begin{figure}[H]
\centering
\begin{Schunk}
\begin{Sinput}
> par(mfrow=c(1,2))
> plot(forecast.arima, xlim=c(2010,2025), ylim=c(385,430))
> plot.forecast(forecasts2 ,xlim=c(2010,2025), ylim=c(385,430))
\end{Sinput}
\end{Schunk}
\caption{Model Comparisons}
\end{figure}



\section{Links and Further Reading}%----------------------------------------------------

and in:\\

http://cran.r-project.org/web/views/TimeSeries.html\\

Another Exemple Datasets are avaliable at:\\

http://www.comp-engine.org/timeseries/browse-data-by-category\\
https://datamarket.com/data/list/?q=provider:tsdl\\

\section{Acknowledgements}%-------------------------------------------------------------
Don't forget to thank TeX and R and other opensource communities if you use their products
```