Week 3: Smoothing, Part 1

Xuan Pham

9/3/2019

# Time Series Components

library(fpp2)

## Loading required package: ggplot2

## Loading required package: forecast

## Registered S3 method overwritten by 'xts':  
## method from  
## as.zoo.xts zoo

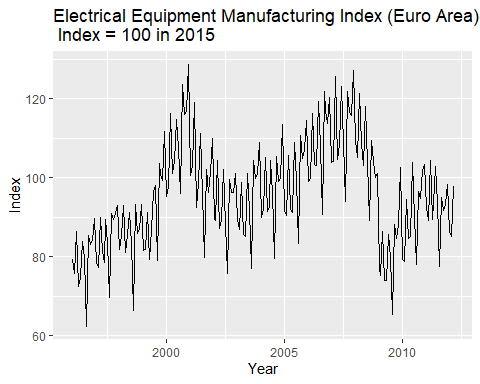
## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

## Registered S3 methods overwritten by 'forecast':  
## method from   
## fitted.fracdiff fracdiff  
## residuals.fracdiff fracdiff

## Loading required package: fma

## Loading required package: expsmooth

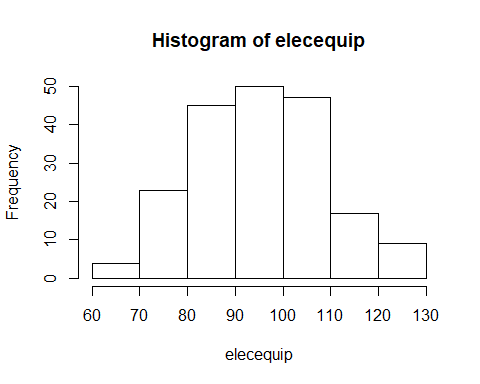
#?elecequip Always look up a preloaded data set to learn more!  
autoplot(elecequip) + xlab("Year") + ylab("Index") +  
 ggtitle("Electrical Equipment Manufacturing Index (Euro Area) \n Index = 100 in 2015")



First, some EDA…

Step 1: Viewing the time series

hist(elecequip)



library(fBasics)

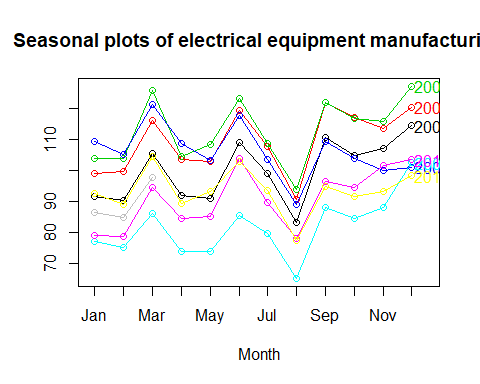
## Loading required package: timeDate

## Loading required package: timeSeries

basicStats(elecequip)

## elecequip  
## nobs 195.000000  
## NAs 0.000000  
## Minimum 62.410000  
## Maximum 128.610000  
## 1. Quartile 86.125000  
## 3. Quartile 104.045000  
## Mean 95.685487  
## Median 94.680000  
## Sum 18658.670000  
## SE Mean 0.945718  
## LCL Mean 93.820279  
## UCL Mean 97.550696  
## Variance 174.404497  
## Stdev 13.206229  
## Skewness 0.181427  
## Kurtosis -0.358091

library(forecast)  
seasonplot(window(elecequip,start = 2005), year.labels = TRUE, col = 1:10, main = "Seasonal plots of electrical equipment manufacturing") #start at base year of index.



library(xts)

## Loading required package: zoo

##   
## Attaching package: 'zoo'

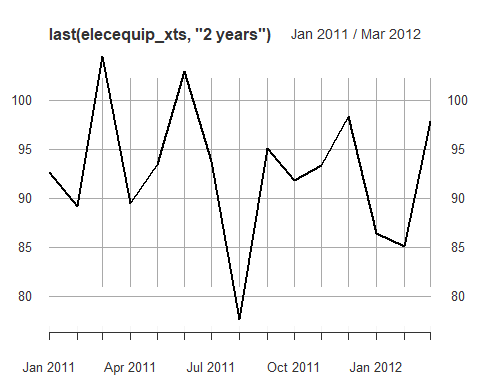
## The following object is masked from 'package:timeSeries':  
##   
## time<-

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

elecequip\_xts <- as.xts(elecequip)  
last(elecequip\_xts, '2 years')

## [,1]  
## Jan 2011 92.57  
## Feb 2011 89.16  
## Mar 2011 104.48  
## Apr 2011 89.45  
## May 2011 93.40  
## Jun 2011 102.90  
## Jul 2011 93.77  
## Aug 2011 77.58  
## Sep 2011 95.04  
## Oct 2011 91.77  
## Nov 2011 93.37  
## Dec 2011 98.34  
## Jan 2012 86.44  
## Feb 2012 85.04  
## Mar 2012 97.80

plot(last(elecequip\_xts, '2 years'))

 Side Note: Here’s the xts cheat sheet from DataCamp: <https://www.datacamp.com/community/blog/r-xts-cheat-sheet>.

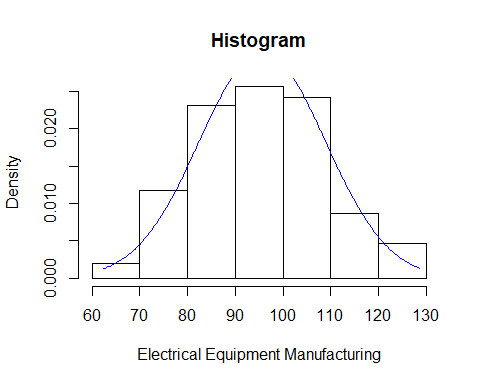
Step 2: Four moments

library(moments)

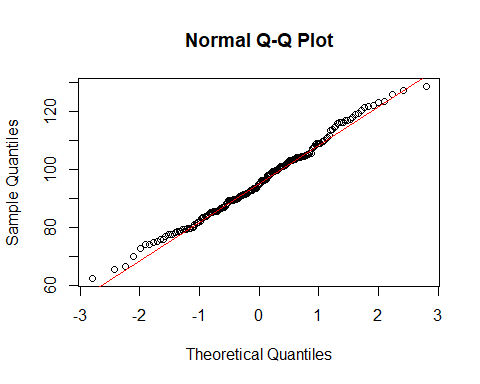
##   
## Attaching package: 'moments'

## The following objects are masked from 'package:timeDate':  
##   
## kurtosis, skewness

hist(elecequip, xlab="Electrical Equipment Manufacturing", prob=TRUE, main="Histogram")   
xfit<-seq(min(elecequip),max(elecequip), length=192)   
yfit<-dnorm(xfit,mean=mean(elecequip),sd=sd(elecequip))   
lines(xfit, yfit, col="blue", lwd=1)



qqnorm(elecequip)   
qqline(elecequip, col = 2)



skewness(elecequip)

## [1] 0.1828314

kurtosis(elecequip)

## [1] 2.669215

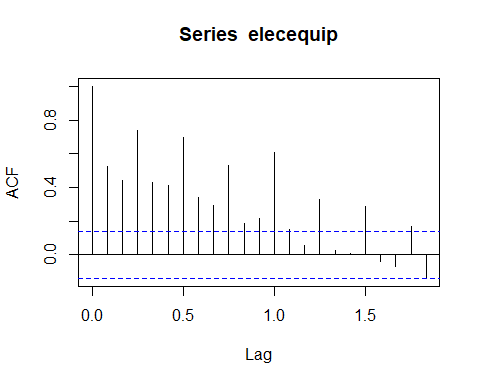
# NORMALITY TESTS   
# Perform Jarque-Bera normality test.   
#H0: Data is normally distributed  
#H1: Data is not normally distributed  
normalTest(elecequip,method=c("jb"))

##   
## Title:  
## Jarque - Bera Normalality Test  
##   
## Test Results:  
## STATISTIC:  
## X-squared: 1.9754  
## P VALUE:  
## Asymptotic p Value: 0.3724   
##   
## Description:  
## Fri Sep 27 09:57:33 2019 by user: Xuan Pham

#Since p-value > 0.05, we fail to reject H0. Data is normally distributed.

Now let’s look at the serial correlation in this time series.

acf(elecequip)



# COMPUTE LJUNG-BOX TEST FOR WHITE NOISE (NO AUTOCORRELATION)  
#H0: p(1) = p(2) = p(k) = 0  
#H1: p(k) is not equal to 0  
  
Box.test(elecequip,lag=12,type='Ljung')

##   
## Box-Ljung test  
##   
## data: elecequip  
## X-squared = 567, df = 12, p-value < 2.2e-16

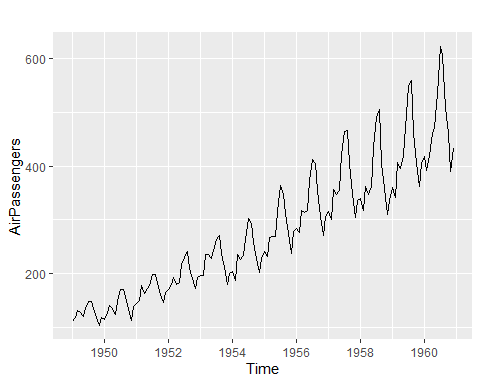
#Since p value < 0.05, we reject H0. There is serial autocorrelation!

## Digression

What would have happened if the JB test shows the data is not normally distributed? What can we do?

* logarithmic transformation
* power transformation (squared, cubed, etc.)
* Box-Cox transformation <https://otexts.com/fpp2/transformations.html>

autoplot(AirPassengers)



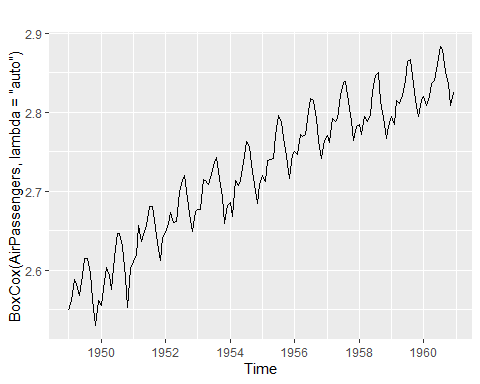
normalTest(AirPassengers,method=c("jb")) #reject H0. Not normal dist.

##   
## Title:  
## Jarque - Bera Normalality Test  
##   
## Test Results:  
## STATISTIC:  
## X-squared: 8.9225  
## P VALUE:  
## Asymptotic p Value: 0.01155   
##   
## Description:  
## Fri Sep 27 09:57:34 2019 by user: Xuan Pham

BoxCox.lambda(AirPassengers)

## [1] -0.2947156

autoplot(BoxCox(AirPassengers, lambda="auto")) #automate



normalTest(BoxCox(AirPassengers,lambda="auto"), method=c("jb")) #reject H0

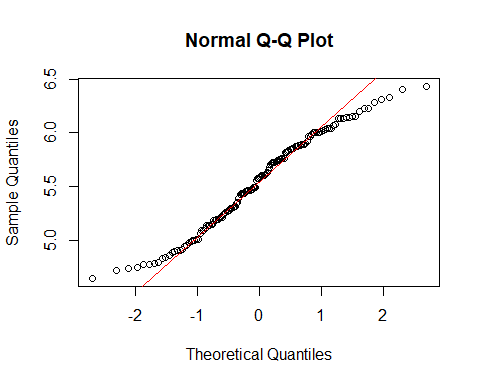
##   
## Title:  
## Jarque - Bera Normalality Test  
##   
## Test Results:  
## STATISTIC:  
## X-squared: 7.2475  
## P VALUE:  
## Asymptotic p Value: 0.02668   
##   
## Description:  
## Fri Sep 27 09:57:34 2019 by user: Xuan Pham

How does this compare to taking the natural log?

#?AirPassengers  
AirPassengers\_log <- log(AirPassengers)  
normalTest(AirPassengers\_log, method=c("jb")) #perhaps normal now?

##   
## Title:  
## Jarque - Bera Normalality Test  
##   
## Test Results:  
## STATISTIC:  
## X-squared: 6.0199  
## P VALUE:  
## Asymptotic p Value: 0.04929   
##   
## Description:  
## Fri Sep 27 09:57:34 2019 by user: Xuan Pham

qqnorm(AirPassengers\_log)  
qqline(AirPassengers\_log, col = 2)



Can you identify the components of this time series?

Trend: Pattern exists when there is a long-term increase or decrease in the data.

Cyclical: Pattern exists when data exhibit rises and falls that are *not of fixed period* (duration usually of at least 2 years).

Seasonal: Pattern exists when a series is influenced by seasonal factors (e.g., the quarter of the year, the month, or day of the week).

# Smoothing

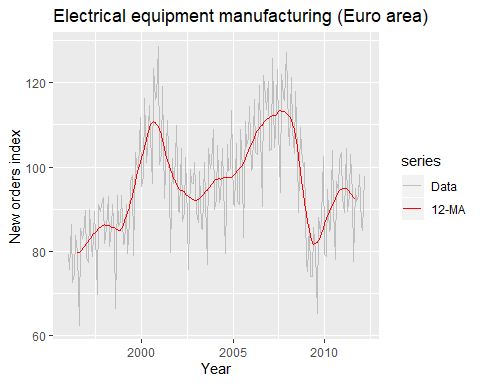
## Approach 1: Moving Average (MA)

ma(elecequip, order=12)

## Jan Feb Mar Apr May Jun Jul  
## 1996 NA NA NA NA NA NA 79.75042  
## 1997 81.82958 82.29875 82.84333 83.34333 83.89125 84.30792 84.57458  
## 1998 86.32292 86.17125 86.12208 86.07000 85.74792 85.58958 85.59583  
## 1999 86.17625 87.31333 88.27083 89.32000 90.46792 91.76333 93.10542  
## 2000 102.21458 103.32375 104.86250 106.32792 107.69000 109.11792 110.03750  
## 2001 109.00875 107.84667 106.27042 104.54667 103.06583 101.63417 100.39250  
## 2002 95.24417 94.74250 94.46625 94.36750 94.16417 93.58292 93.20667  
## 2003 92.14708 92.35417 92.61042 92.95083 93.29333 93.83083 94.18667  
## 2004 96.73042 97.02250 97.17667 97.20000 97.12375 97.25958 97.52292  
## 2005 97.95125 98.23583 98.61708 99.08125 99.62583 99.96542 100.31375  
## 2006 105.43417 106.09708 106.85667 107.82167 108.60500 109.13042 109.57792  
## 2007 111.83375 112.01833 112.16667 112.15750 112.23500 112.60708 113.11917  
## 2008 112.33375 111.92000 111.20417 110.15542 108.96625 107.22833 104.81042  
## 2009 89.06083 87.07042 85.18208 83.48083 82.17042 81.72500 81.86000  
## 2010 86.66542 87.61333 88.49625 89.26667 90.25667 90.86667 91.46042  
## 2011 94.86250 95.00167 94.91750 94.73542 94.26417 93.70000 93.23042  
## 2012 NA NA NA   
## Aug Sep Oct Nov Dec  
## 1996 79.78917 80.00500 80.51375 81.03417 81.43083  
## 1997 85.04000 85.50083 85.62333 85.90583 86.26375  
## 1998 85.43458 85.21250 85.07167 85.05958 85.34333  
## 1999 94.34333 96.05125 97.98958 99.59833 101.06167  
## 2000 110.47667 110.81458 110.57750 110.00083 109.63333  
## 2001 99.35833 98.17125 97.33292 96.71500 95.95000  
## 2002 93.10083 92.77542 92.48542 92.25250 92.03833  
## 2003 94.43250 94.91458 95.42167 95.97458 96.41292  
## 2004 97.52542 97.47125 97.51667 97.48917 97.62875  
## 2005 101.01417 101.84417 102.75750 103.72208 104.64417  
## 2006 109.95333 110.52917 110.97917 111.25417 111.63625  
## 2007 113.39625 113.26000 113.24667 113.19792 112.76333  
## 2008 102.22250 99.51583 96.61708 93.96083 91.40167  
## 2009 82.08708 82.57750 83.35875 84.26000 85.48625  
## 2010 92.44833 93.29333 93.90375 94.44500 94.74333  
## 2011 92.80333 92.35333 NA NA NA  
## 2012

autoplot(elecequip, series="Data") +  
 autolayer(ma(elecequip, 12), series="12-MA") +  
 xlab("Year") + ylab("New orders index") +  
 ggtitle("Electrical equipment manufacturing (Euro area)") +  
 scale\_colour\_manual(values=c("Data"="grey","12-MA"="red"),  
 breaks=c("Data","12-MA"))

## Warning: Removed 12 rows containing missing values (geom\_path).



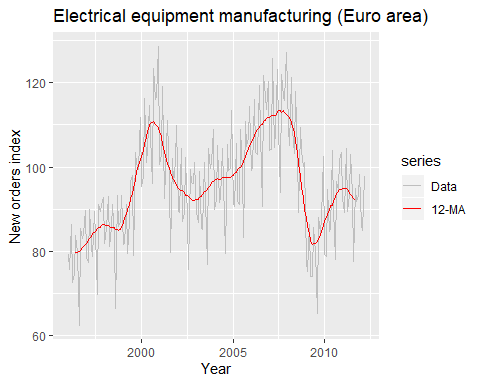
What difference(s) do you see with this smoothing method?

ma(elecequip, 12, centre = FALSE)

## Jan Feb Mar Apr May Jun Jul  
## 1996 NA NA NA NA NA 79.78000 79.72083  
## 1997 81.99083 82.60667 83.08000 83.60667 84.17583 84.44000 84.70917  
## 1998 86.31417 86.02833 86.21583 85.92417 85.57167 85.60750 85.58417  
## 1999 86.78167 87.84500 88.69667 89.94333 90.99250 92.53417 93.67667  
## 2000 102.61583 104.03167 105.69333 106.96250 108.41750 109.81833 110.25667  
## 2001 108.53000 107.16333 105.37750 103.71583 102.41583 100.85250 99.93250  
## 2002 94.91083 94.57417 94.35833 94.37667 93.95167 93.21417 93.19917  
## 2003 92.30000 92.40833 92.81250 93.08917 93.49750 94.16417 94.20917  
## 2004 96.91417 97.13083 97.22250 97.17750 97.07000 97.44917 97.59667  
## 2005 98.07833 98.39333 98.84083 99.32167 99.93000 100.00083 100.62667  
## 2006 105.79583 106.39833 107.31500 108.32833 108.88167 109.37917 109.77667  
## 2007 111.87333 112.16333 112.17000 112.14500 112.32500 112.88917 113.34917  
## 2008 112.12000 111.72000 110.68833 109.62250 108.31000 106.14667 103.47417  
## 2009 88.06667 86.07417 84.29000 82.67167 81.66917 81.78083 81.93917  
## 2010 87.08083 88.14583 88.84667 89.68667 90.82667 90.90667 92.01417  
## 2011 95.02500 94.97833 94.85667 94.61417 93.91417 93.48583 92.97500  
## 2012 NA NA NA   
## Aug Sep Oct Nov Dec  
## 1996 79.85750 80.15250 80.87500 81.19333 81.66833  
## 1997 85.37083 85.63083 85.61583 86.19583 86.33167  
## 1998 85.28500 85.14000 85.00333 85.11583 85.57083  
## 1999 95.01000 97.09250 98.88667 100.31000 101.81333  
## 2000 110.69667 110.93250 110.22250 109.77917 109.48750  
## 2001 98.78417 97.55833 97.10750 96.32250 95.57750  
## 2002 93.00250 92.54833 92.42250 92.08250 91.99417  
## 2003 94.65583 95.17333 95.67000 96.27917 96.54667  
## 2004 97.45417 97.48833 97.54500 97.43333 97.82417  
## 2005 101.40167 102.28667 103.22833 104.21583 105.07250  
## 2006 110.13000 110.92833 111.03000 111.47833 111.79417  
## 2007 113.44333 113.07667 113.41667 112.97917 112.54750  
## 2008 100.97083 98.06083 95.17333 92.74833 90.05500  
## 2009 82.23500 82.92000 83.79750 84.72250 86.25000  
## 2010 92.88250 93.70417 94.10333 94.78667 94.70000  
## 2011 92.63167 92.07500 NA NA NA  
## 2012

autoplot(elecequip, series="Data") +  
 autolayer(ma(elecequip, 12, centre=FALSE), series="12-MA") +  
 xlab("Year") + ylab("New orders index") +  
 ggtitle("Electrical equipment manufacturing (Euro area)") +  
 scale\_colour\_manual(values=c("Data"="grey","12-MA"="red"),  
 breaks=c("Data","12-MA"))

## Warning: Removed 11 rows containing missing values (geom\_path).



## Quick Exercise

Change the MA to 3, 6, 9, and 24 periods. What differences do you see?

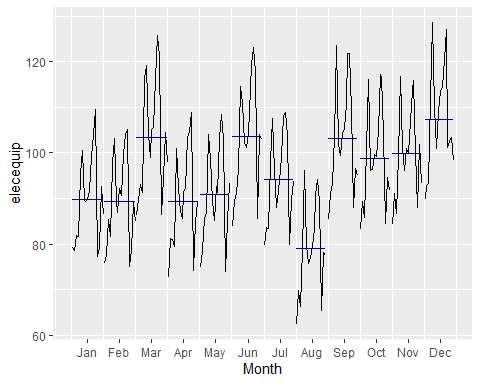
## Time series decomposition

* Additive model appropriate if magnitude of seasonal fluctuations does not vary with level.
* If seasonal are proportional to level of series, then multiplicative model appropriate.
* Multiplicative decomposition more prevalent with economic series
* Logs turn multiplicative relationship into an additive relationship:

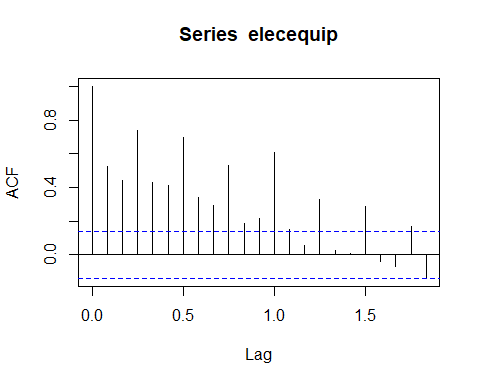
Are seasonal fluctuations a constant value or proportionally different?  
Additive: e.g. Manufacturing order index is always 110 in March. Multiplicative: e.g. Manufacturing is 10% higher (or lower) in March.

Let’s look at some plots.

ggsubseriesplot(elecequip)

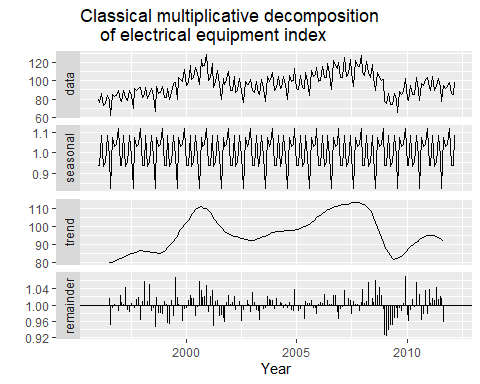


acf(elecequip)



#Multiplicative Decomposition

elecequip %>% decompose(type="multiplicative") %>%  
 autoplot() + xlab("Year") +  
 ggtitle("Classical multiplicative decomposition  
 of electrical equipment index")



How does the Multiplicative Decomposition Model do in “smoothing” the time series? Do you see any problem in the remainder component? Is it moving with any other component?

What’s wrong with classical decomposition?

1. No estimates for the first few and last few observations in time series. This is due to the MA technique being used to do the decomposition.
2. Trend-cycle component is overreactive (rises and falls too rapidly).
3. Seasonal components repeat year to year.
4. Cannot deal with “shocks”.

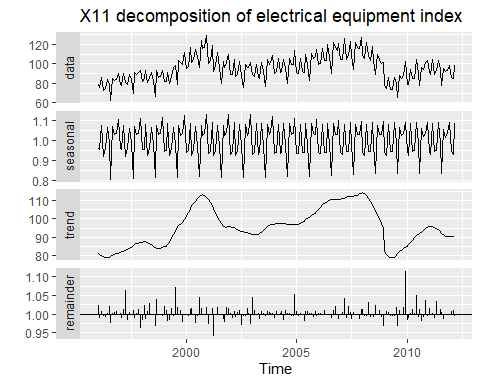
## X11 & SEATS Decomposition

library(seasonal)

##   
## Attaching package: 'seasonal'

## The following objects are masked from 'package:timeSeries':  
##   
## outlier, series

fit.x11 <- seas(elecequip, x11="")  
autoplot(fit.x11) + ggtitle("X11 decomposition of electrical equipment index")



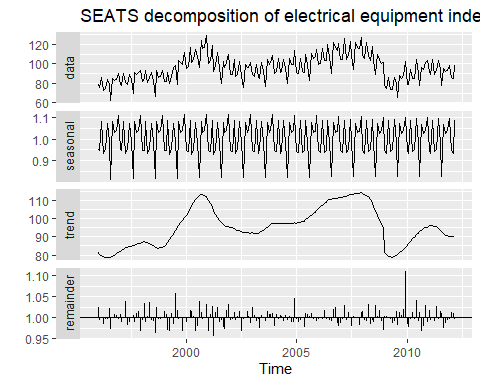
Advantages:

* Relatively robust to outliers
* Completely automated choices for trend and seasonal changes
* Very widely tested on economic data over a long period of time.

Disadvantages:

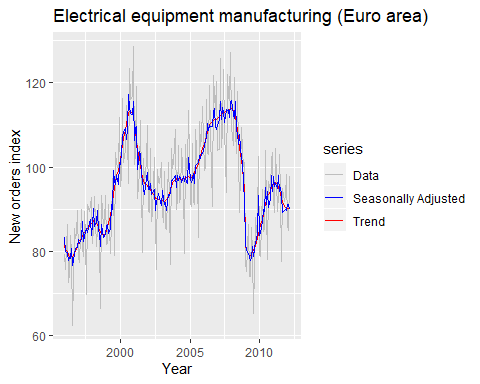
* No prediction/confidence intervals
* Ad hoc method with no underlying model
* Only developed for quarterly and monthly data

library(seasonal)  
fit.seats <- seas(elecequip)  
autoplot(fit.seats) +  
 ggtitle("SEATS decomposition of electrical equipment index")

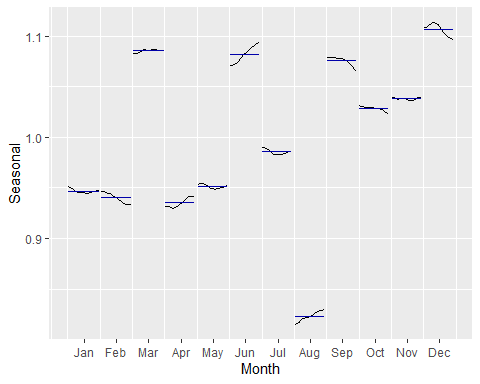


### SEATS (SEASONAL EXTRACTION IN ARIMA Time Series)

autoplot(elecequip, series="Data") +  
 autolayer(trendcycle(fit.seats), series="Trend") +  
 autolayer(seasadj(fit.seats), series="Seasonally Adjusted") +  
 xlab("Year") + ylab("New orders index") +  
 ggtitle("Electrical equipment manufacturing (Euro area)") +  
 scale\_colour\_manual(values=c("gray","blue","red"),  
 breaks=c("Data","Seasonally Adjusted","Trend"))



ggsubseriesplot(seasonal(fit.seats)) + ylab("Seasonal")



Advantages: \* Model-based \* Smooth trend estimate \* Allows estimates at end points \* Allows changing seasonality \* Developed for economic data

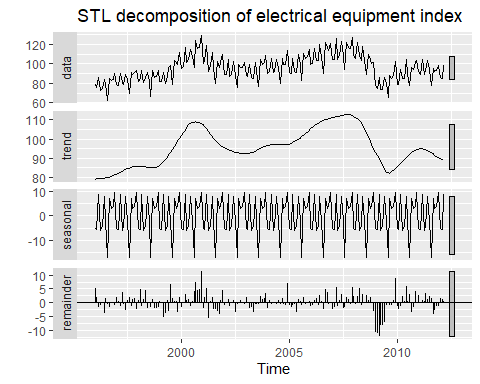
Disadvantage:  
\* Only developed for quarterly and monthly data

## STL decomposition

* STL: “Seasonal and Trend decomposition using Loess”
* Very versatile and robust.
* STL will handle any type of seasonality.
* Seasonal component allowed to change over time, and rate of change controlled by user.
* Smoothness of trend-cycle also controlled by user.
* Robust to outliers
* Not trading day or calendar adjustments.
* Only additive.
* Take logs to get multiplicative decomposition.
* Use Box-Cox transformations to get other decompositions.

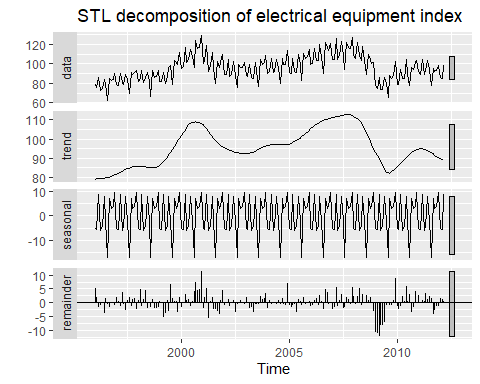
On STL algorithm: <http://www.gardner.fyi/blog/STL-Part-II/>  
On LOESS: <http://r-statistics.co/Loess-Regression-With-R.html> and <http://varianceexplained.org/files/loess.html>. ## STL decomposition (SEASONAL AND TREND DECOMPOSITION USING LOESS)

#?stl   
#look at the help file first  
#`t.window` controls wiggliness of trend component. Default is defined in help file.  
#`s.window` controls variation on seasonal component. periodic means to use the default period (month in this case) to calculate the seasonal components  
fit <- stl(elecequip, s.window="periodic", robust=TRUE)  
autoplot(fit) +  
 ggtitle("STL decomposition of electrical equipment index")



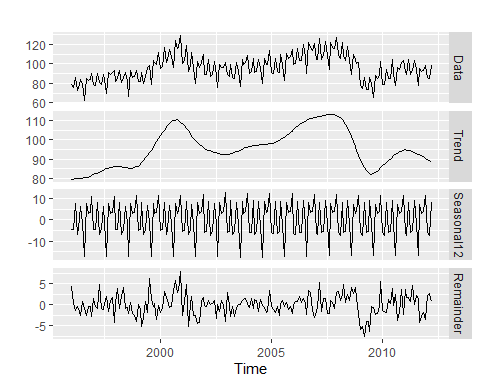
Try out different s and t window values to see the different. Be sure to look at the ACF plot to see if you can identify the number of lags to use for the windows.

fit <- stl(elecequip, s.window="periodic", t.window=NULL, robust=TRUE)  
autoplot(fit) +  
 ggtitle("STL decomposition of electrical equipment index")



Or we can try to “automate” the process:

elecequip %>% mstl() %>% autoplot()



For more information, see link: <https://www.scb.se/contentassets/ca21efb41fee47d293bbee5bf7be7fb3/stl-a-seasonal-trend-decomposition-procedure-based-on-loess.pdf>

## Forecasting and decomposition

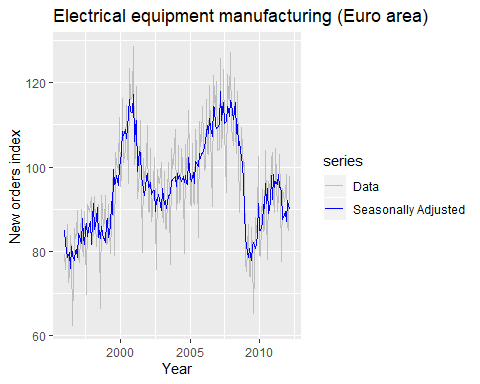
* Forecast seasonal component by repeating the last year
* Forecast seasonally adjusted data using non-seasonal time series method.
* Combine forecasts of seasonal component with forecasts of seasonally adjusted data to get forecasts of original data.
* Sometimes a decomposition is useful just for understanding the data before building a separate forecasting model.

## Electrical equipment

fit <- stl(elecequip, t.window=13, s.window="periodic")

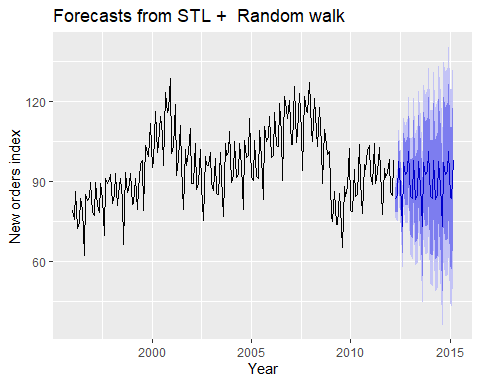
We use the naive method to forecast the seasonally adjusted component (Trend, Cyclical, and Random) and seasonality component. The two components are added together to create a “reseasonalized” forecast.

autoplot(elecequip, series="Data") +  
 autolayer(seasadj(fit), series="Seasonally Adjusted") +  
 xlab("Year") + ylab("New orders index") +  
 ggtitle("Electrical equipment manufacturing (Euro area)") +  
 scale\_colour\_manual(values=c("gray","blue"),  
 breaks=c("Data","Seasonally Adjusted"))



Now let’s bring everything together to get forecasts

fit %>% forecast(method='naive', h=36) %>%  
 autoplot() + ylab("New orders index") + xlab("Year")



summary(fit %>% forecast(method='naive'))

##   
## Forecast method: STL + Random walk  
##   
## Model Information:  
## Call: rwf(y = x, h = h, drift = FALSE, level = level)   
##   
## Residual sd: 3.5154   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set 0.02566723 3.506388 2.723322 -0.02484186 2.874973 0.333245  
## ACF1  
## Training set -0.3373772  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## Apr 2012 83.45798 78.96436 87.95159 76.58558 90.33037  
## May 2012 85.04087 78.68593 91.39580 75.32184 94.75990  
## Jun 2012 97.68151 89.89834 105.46469 85.77818 109.58485  
## Jul 2012 88.46028 79.47305 97.44751 74.71549 102.20507  
## Aug 2012 73.19696 63.14893 83.24499 57.82982 88.56410  
## Sep 2012 97.41490 86.40783 108.42196 80.58104 114.24875  
## Oct 2012 92.87455 80.98555 104.76354 74.69190 111.05719  
## Nov 2012 93.74607 81.03621 106.45594 74.30801 113.18414  
## Dec 2012 101.17630 87.69545 114.65715 80.55912 121.79348  
## Jan 2013 84.32944 70.11938 98.53951 62.59703 106.06186  
## Feb 2013 83.80002 68.89638 98.70366 61.00686 106.59317  
## Mar 2013 97.80000 82.23366 113.36634 73.99333 121.60667  
## Apr 2013 83.45798 67.25601 99.65994 58.67921 108.23674  
## May 2013 85.04087 68.22729 101.85444 59.32673 110.75501  
## Jun 2013 97.68151 80.27781 115.08522 71.06485 124.29818  
## Jul 2013 88.46028 70.48581 106.43475 60.97071 115.94985  
## Aug 2013 73.19696 54.66931 91.72462 44.86136 101.53257  
## Sep 2013 97.41490 78.35009 116.47970 68.25780 126.57199  
## Oct 2013 92.87455 73.28733 112.46177 62.91848 122.83062  
## Nov 2013 93.74607 73.65001 113.84214 63.01180 124.48035  
## Dec 2013 101.17630 80.58396 121.76864 69.68303 132.66956  
## Jan 2014 84.32944 63.25251 105.40637 52.09506 116.56383  
## Feb 2014 83.80002 62.24939 105.35064 50.84117 116.75886  
## Mar 2014 97.80000 75.78586 119.81414 64.13229 131.46771

## Decomposition and prediction intervals

* It is common to take the prediction intervals from the seasonally adjusted forecasts and modify them with the seasonal component.
* This ignores the uncertainty in the seasonal component estimate.
* It also ignores the uncertainty in the future seasonal pattern.