```
library(fpp)


# the total quarterly beer production in Australia, in megalitres from 1956 to 2008
summary(ausbeer)  #check out the statistics

beer2 <- window(ausbeer, start = 1992)
summary(beer2)  #and now

# plot(forecast(beer2), xlab = "Year", ylab = "megaliters", main = "")  #let's plot


# ploting the data for the Total monthly scripts for pharmaceutical products
# falling under ATC code A10, as recorded by the Australian Health Insurance Commission.
plot(a10)   #increasing trend

#can we see better that seasonal pattern
seasonplot(a10,ylab="$ million", xlab="Year",
           main="Seasonal plot: antidiabetic drug sales",
           year.labels=TRUE, year.labels.left=TRUE, col=1:20, pch=19)

#and the month plot
monthplot(a10,ylab="$ million",xlab="Month",xaxt="n",
          main="Seasonal deviation plot: antidiabetic drug sales")
axis(1,at=1:12,labels=month.abb,cex=0.8)

#and a scatterplot for the fuel economy data on 2009 vehicles in the US
plot(jitter(fuel[,5]), jitter(fuel[,8]), xlab="City mpg", ylab="Carbon footprint")

pairs(fuel[,-c(1:2,4,7)], pch=19)

##############################################
#Data summaries, correlation and autocorrelation

fuel2 <- fuel[fuel$Litres<2,]
summary(fuel2[,"Carbon"])
sd(fuel2[,"Carbon"])   # the standart deviation function


cor(fuel2[,"Carbon"], fuel2[,"City"])   #a simple correlation fucntion among carbon and city emitions?

beer2 <- window(ausbeer, start=1992, end=2006)   #select the data since 1992 up to 2006
lag.plot(beer2, lags=16, do.lines=FALSE)     #show the autocorrelation for 6 different lags (quarterly)
Acf(beer2)   #the autocorrelation function (ACF)

#how does the autocorrelation of white noise looks like
set.seed(30)
x <- ts(rnorm(50))
plot(x, main="White noise")  #producting white noise

Acf(x)    #the ACF function

##############################################
#Simple forecasting methods

#creating the prediction series for 11 quarters
#for four different simple forecasting methods
beer2 <- window(ausbeer,start=1992,end=2006-.1)
beerfit1 <- meanf(beer2, h=11)
beerfit2 <- naive(beer2, h=11)
beerfit3 <- snaive(beer2, h=11)
beerfit4 <-rwf(beer2, h=11,drift=TRUE)

#and ploting
plot(beerfit1, plot.conf=FALSE,
     main="Forecasts for quarterly beer production")
lines(beerfit2$mean, col=2)
lines(beerfit3$mean, col=3)
lines(beerfit4$mean, col=5)
legend("topright",lty=1,cex = 0.5,col=c(4,2,3,5),
       legend=c("Mean method","Naive method","Seasonal naive method","Drift method"))



##############################################
#Transfortations and adjustments

#Australian monthly electricity production: Jan 1956 – Aug 1995.
plot(elec, ylab="Electricity demand",
     xlab="Year", main="Monthly electricity demand")
```

```
#logarithmic transformation
plot(log(elec), ylab="Transformed electricity demand",
     xlab="Year", main="Transformed monthly electricity demand")
title(main="Log",line=-1)

# the Box-Cox transformation
lambda <- BoxCox.lambda(elec) # = 0.27
plot(BoxCox(elec,lambda))


# monthdays <- rep(c(31,28,31,30,31,30,31,31,30,31,30,31),14)
# monthdays[26 + (4*12)*(0:2)] <- 29
#
# par(mfrow=c(2,1))
# plot(milk, main="Monthly milk production per cow",
#      ylab="Pounds",xlab="Years")
# plot(milk/monthdays, main="Average milk production per cow per day",
#      ylab="Pounds", xlab="Years")


##################################################
#Evaluating forecast accuracy


#recall the time series forcast for the quartely beer production with data up to 2006
beer2 <- window(ausbeer,start=1992,end=2006-.1)
beerfit1 <- meanf(beer2,h=11)
beerfit2 <- rwf(beer2,h=11)
beerfit3 <- snaive(beer2,h=11)

par(mfrow=c(1,1))
plot(beerfit1, plot.conf=FALSE,
     main="Forecasts for quarterly beer production")
lines(beerfit2$mean, col=2)
lines(beerfit3$mean, col=3)
lines(ausbeer)
legend("topright", lty=1, cex=0.5, col=c(4,2,3),
       legend=c("Mean method","Naive method","Seasonal naive method"))


beer3 <- window(ausbeer, start=2006)
accuracy(beerfit1, beer3)
accuracy(beerfit2, beer3)
accuracy(beerfit3, beer3)

##################################################
#Residual diagnostics

#taking the Dow-Jones example
dj2 <- window(dj, end=250)
plot(dj2, main="Dow Jones Index (daily ending 15 Jul 94)",
     ylab="", xlab="Day")

res <- residuals(naive(dj2))
plot(res, main="Residuals from naive method",
     ylab="", xlab="Day")

Acf(res, main="ACF of residuals")

hist(res, nclass="FD", main="Histogram of residuals")

#tests for group autocorrelations
#for up to lag 10 and K=0 (because in the Dow-Jones no parameters)
#applying the Box-Pierce test
Box.test(res, lag = 10, fitdf = 0)
#applying the Box-Ljung testfi
Box.test(res, lag = 10, fitdf = 0, type = "Lj")


##################################################
#Time series decomposition
#Moving averages

#Plotting the annual electricity sales for South Australia in GWh from 1989 to 2008.
plot(elecsales, main="Residential electricity sales", ylab="GWh",xlab="Year")

# a simple moving average smoother
ma(elecsales, order=5)
```

```
#ploting the actual electricity data together with the average smoohted data
plot(elecsales, main="Residential electricity sales", ylab="GWh", xlab="Year")
lines(ma(elecsales,5),col="red")

#let's experiment with the k periods of time t, a.k.a the order in the ma function
par(mfrow=c(2,2))
plot(elecsales, main="Residential electricity sales", ylab="GWh", xlab="Year")
lines(ma(elecsales,3),col="red")
legend("topleft", lty=1, cex=0.5, col=c("red"),
       legend=c("3-MA"))
plot(elecsales, main="Residential electricity sales", ylab="GWh", xlab="Year")
lines(ma(elecsales,5),col="red")
legend("topleft", lty=1, cex=0.5, col=c("red"),
       legend=c("5-MA"))
plot(elecsales, main="Residential electricity sales", ylab="GWh", xlab="Year")
lines(ma(elecsales,7),col="red")
legend("topleft", lty=1, cex=0.5, col=c("red"),
       legend=c("7-MA"))
plot(elecsales, main="Residential electricity sales", ylab="GWh", xlab="Year")
lines(ma(elecsales,9),col="red")
legend("topleft", lty=1, cex=0.5, col=c("red"),
       legend=c("9-MA"))


#moving averages of moving averages
beer2 <- window(ausbeer,start=1992)
ma4 <- ma(beer2, order=4, centre=FALSE)
ma2x4 <- ma(beer2, order=4, centre=TRUE) #by having the centre=True we achieve a 2x4 average

#for the electricity data, since this data are annual we can use a 2x12-MA find the trend and eliminate the seasonality
plot(elecequip, ylab="New orders index", col="gray", main="Electrical equipment manufacturing (Euro area)")
lines(ma(elecequip, order=12, centre=TRUE), col="red")

# and what happens if centre is false?
plot(elecequip, ylab="New orders index", col="gray", main="Electrical equipment manufacturing (Euro area)")
lines(ma(elecequip, order=12, centre=FALSE), col="red")

################################################
#STL decomposition

#let's use the stl function to decompose the electricity data with the STL method
#t.window is the trend window
#s.window is the seasonal window
#how do this affect the fitting of the data?
#play with the parameters
fit <- stl(elecequip, t.window=15, s.window="periodic", robust=TRUE)
plot(fit)


################################################
#Forecasting with decomposition

#we take the decomposed data
fit <- stl(elecequip, t.window=15, s.window="periodic", robust=TRUE)
eeadj <- seasadj(fit)  #and we feed them to the seasonal adjestment function to remove the seasonality from the data

# plot(fit)
# plot(eeadj)

#then we use a naive prediction model to forecast from this seasonally adjested data
plot(naive(eeadj), xlab="New orders index", main="Naive forecasts of seasonally adjusted data")

#we put back the seasonality into the forcasting data though the forecast method
fcast <- forecast(fit, method="naive")
plot(fcast, ylab="New orders index")
```