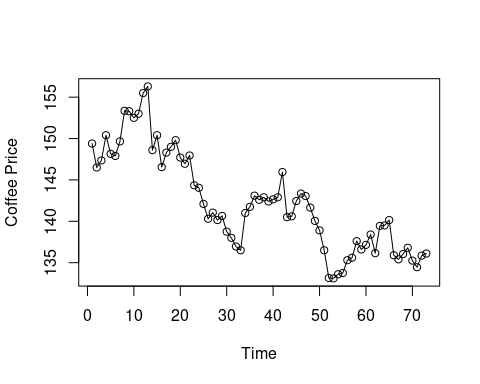
coffee <- read.table('Coffee\_prices\_2013.txt', sep = '\t', header = TRUE)

## Question 1.

The overall time series is decreasing linearly. The trend does repeat itself and has a seasonality of about 30 days, where in the every 10th of month we see a spike, though overall the trend does appear to be decreasing. As there is data for only about 80 days, we cannot extrapolate whether there is a cyclical component. There are some random fluctuation, but they appear to be minor.

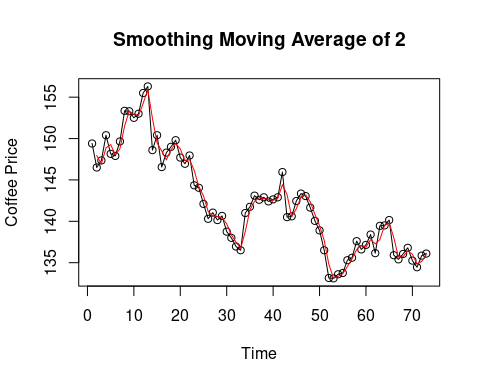
plot(coffee$time, coffee$price, xlab = "Time", ylab = "Coffee Price")  
lines(coffee$time, coffee$price)



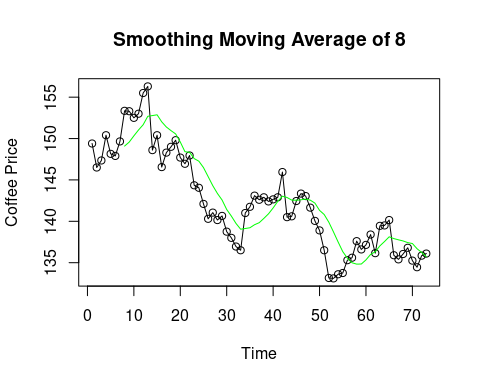
## Question 2.

As seen below, a moving average of a 8 is much more smoothed and captures the overall trend of price dropping of coffee in the graph. The moving average of 2 much more jagged, as it mostly accounts for the day to day changes in the graph, while still showing an overall decrease of coffee prices as a trend in graph. As such sudden change is described and captured better by moving average of 2 rather than the moving average of 8.

library(TTR)  
smooth1 = SMA(coffee$price, n = 2)  
smooth2 = SMA(coffee$price, n = 8)  
  
plot(coffee$time, coffee$price, xlab = "Time", ylab = "Coffee Price", main = "Smoothing Moving Average of 2")  
lines(coffee$time, coffee$price)  
lines(coffee$time, smooth1, col = 'red')



plot(coffee$time, coffee$price, xlab = "Time", ylab = "Coffee Price", main = "Smoothing Moving Average of 8")  
lines(coffee$time, coffee$price)  
lines(coffee$time, smooth2, col = 'green')



## Question 3

A smoothing exponential curve with a = 0.8, is less smooth than with a exponential curve with a = 0.2. The curve with a = 0.8 is more jagged and responsive to changes in prices. The curve with a = 0.2 on the other hand is not as responsive as the curve with a = 0.8, but shows the overall trend of coffee price declining much better.

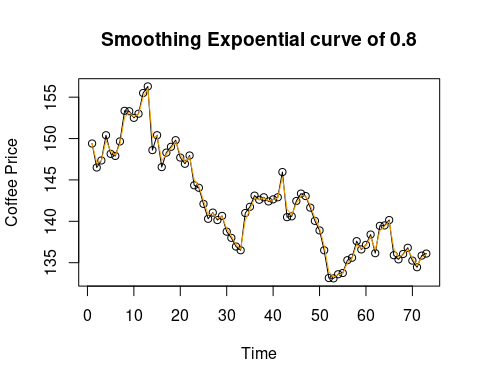
smoothexp1 =EMA(coffee$price, ratio = 0.8, n = 1)

## Warning in EMA(coffee$price, ratio = 0.8, n = 1): both 'n' and 'ratio' are  
## specified; using 'n'

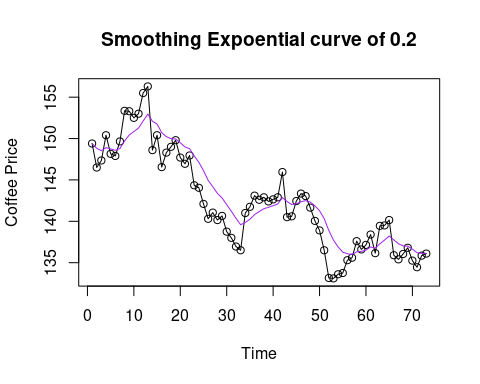
smoothexp2 = EMA(coffee$price, ratio = 0.2, n = 1)

## Warning in EMA(coffee$price, ratio = 0.2, n = 1): both 'n' and 'ratio' are  
## specified; using 'n'

plot(coffee$time, coffee$price, xlab = "Time", ylab = "Coffee Price", main = "Smoothing Expoential curve of 0.8")  
lines(coffee$time, coffee$price)  
lines(coffee$time, smoothexp1, col = 'orange')



plot(coffee$time, coffee$price, xlab = "Time", ylab = "Coffee Price", main = "Smoothing Expoential curve of 0.2")  
lines(coffee$time, coffee$price)  
lines(coffee$time, smoothexp2, col = 'purple')



## Question 4

The model depends on the first 1 lag. The model depends on the first lag the most because it has the highest coefficient. All other lags are considerably less than the first lag, and thus the model depends the most on the first lag.

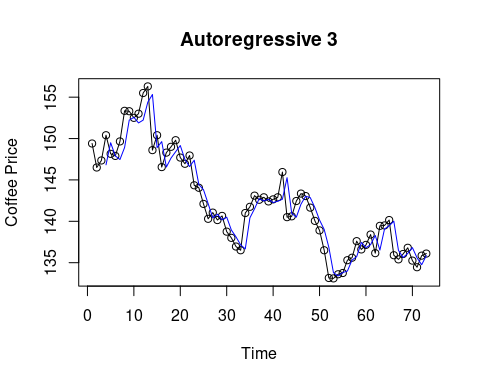
acf(coffee$price,lag.max = 5,plot = FALSE)

##   
## Autocorrelations of series 'coffee$price', by lag  
##   
## 0 1 2 3 4 5   
## 1.000 0.923 0.870 0.810 0.753 0.694

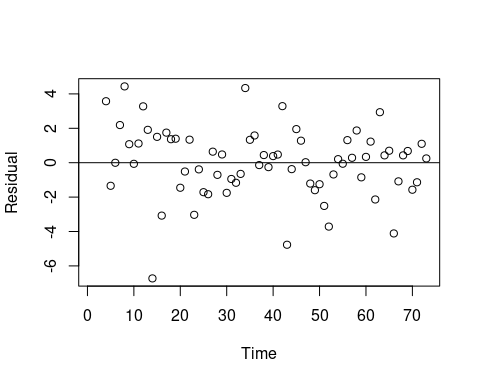
autoregressive = ar(coffee$price, aic = FALSE, order.max = 3, demean = FALSE,  
intercept = TRUE, method = 'ols')  
autoregressive

##   
## Call:  
## ar(x = coffee$price, aic = FALSE, order.max = 3, method = "ols", demean = FALSE, intercept = TRUE)  
##   
## Coefficients:  
## 1 2 3   
## 0.8456 0.1032 0.0004   
##   
## Intercept: 7.047 (5.952)   
##   
## Order selected 3 sigma^2 estimated as 4.009

fitted.autoregressive = coffee$price - autoregressive$resid  
  
plot(coffee$time, coffee$price, xlab = "Time", ylab = "Coffee Price", main = "Autoregressive 3")  
lines(coffee$time, coffee$price)  
lines(coffee$time, fitted.autoregressive, col = 'blue')



plot(coffee$time,autoregressive$resid, xlab = "Time", ylab = "Residual")  
abline(0,0)



## Question 5

The autoregressive model gives the best prediction, as it has the lowest APE of 1.94. All the other models have an APE higher than the autoregressive model.

absolute = 138.90  
  
prediction.smooth1 = smooth1[length(smooth1)]  
prediction.smooth2 = smooth2[length(smooth2)]  
prediction.smoothexp1 = smoothexp1[length(smoothexp1)]  
prediction.smoothexp2 = smoothexp2[length(smoothexp2)]  
prediction.autoregressive = predict(autoregressive,n.ahead = 1, se.fit = FALSE)  
  
abs(absolute-prediction.smooth1)/abs(absolute)\*100

## [1] 2.105838

abs(absolute-prediction.smooth2)/abs(absolute)\*100

## [1] 2.28582

abs(absolute-prediction.smoothexp1)/abs(absolute)\*100

## [1] 2.085957

abs(absolute-prediction.smoothexp2)/abs(absolute)\*100

## [1] 1.997467

abs(absolute-prediction.autoregressive)/abs(absolute)\*100

## Time Series:  
## Start = 74   
## End = 74   
## Frequency = 1   
## [1] 1.937486