Course Project

John Navarro

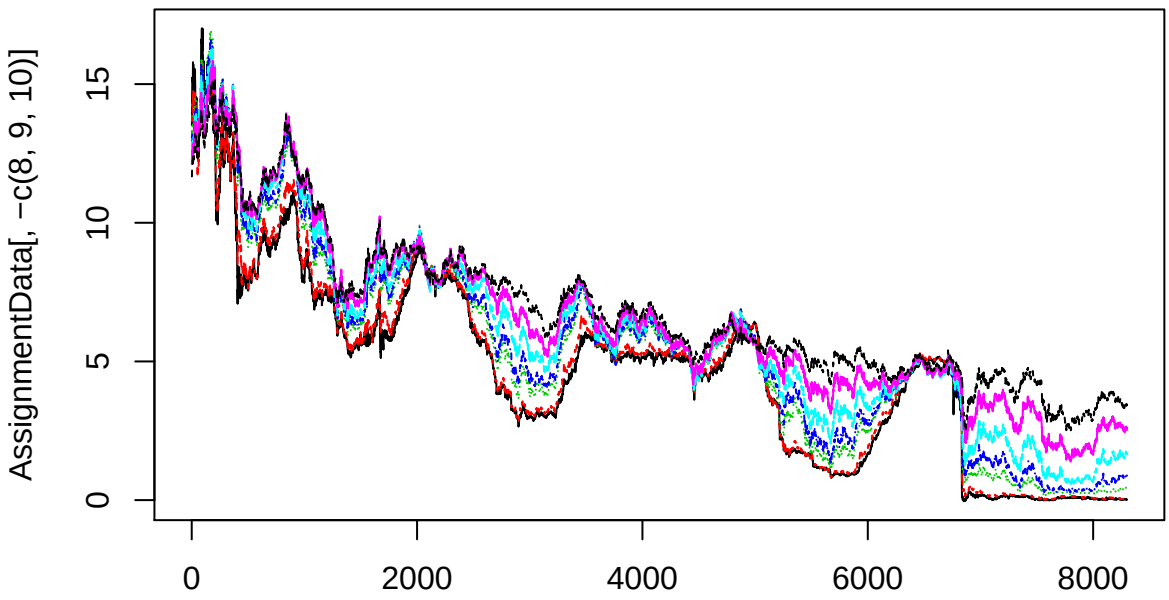
October 25, 2016

Step 1.

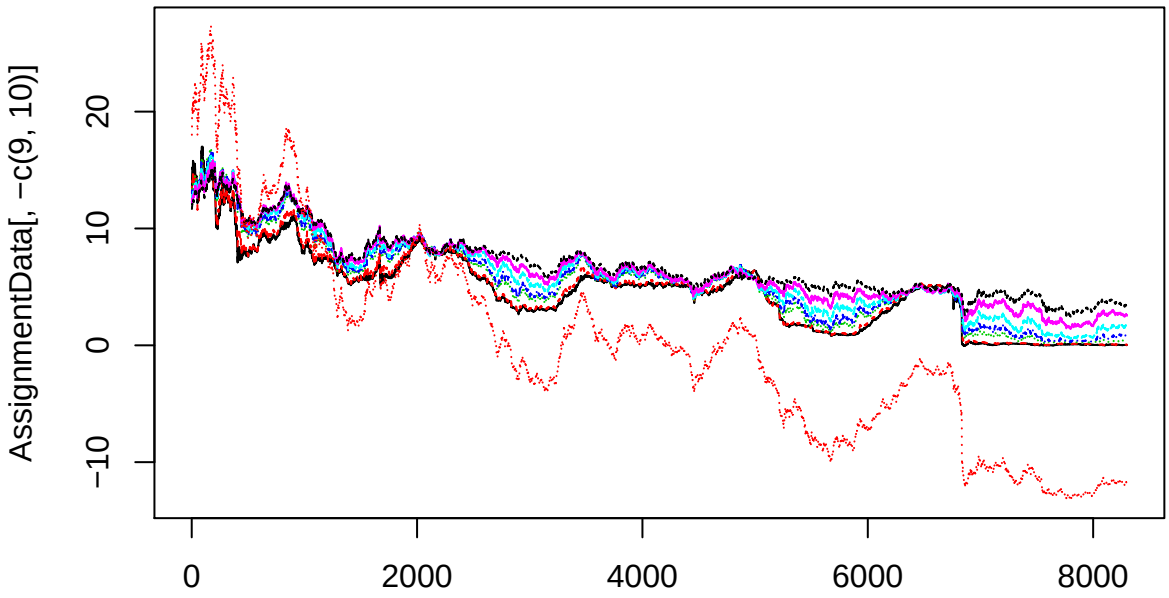
```
datapath <- "C:/Users/JohntheGreat/Documents/MSCA/StatisticalAnalysis/CourseProject"
AssignmentData<-
  read.csv(file=paste(datapath,"regressionassignmentdata2014.csv",sep="/"),
           row.names=1,header=TRUE,sep=",")
head(AssignmentData)
```

```
##           USGG3M USGG6M USGG2YR USGG3YR USGG5YR USGG10YR USGG30YR  Output1
## 1/5/1981   13.52  13.09  12.289   12.28  12.294   12.152   11.672 18.01553
## 1/6/1981   13.58  13.16  12.429   12.31  12.214   12.112   11.672 18.09140
## 1/7/1981   14.50  13.90  12.929   12.78  12.614   12.382   11.892 19.44731
## 1/8/1981   14.76  14.00  13.099   12.95  12.684   12.352   11.912 19.74851
## 1/9/1981   15.20  14.30  13.539   13.28  12.884   12.572   12.132 20.57204
## 1/12/1981  15.22  14.23  13.179   12.94  12.714   12.452   12.082 20.14218
##           Easing Tightening
## 1/5/1981      NA         NA
## 1/6/1981      NA         NA
## 1/7/1981      NA         NA
## 1/8/1981      NA         NA
## 1/9/1981      NA         NA
## 1/12/1981     NA         NA
```

```
matplot(AssignmentData[,-c(8,9,10)],type='l')
```




```
matplot(AssignmentData[,-c(9,10)],type='l')
```



Step 2.

```
##Estimate simple regression model with each of the input variables and the output variable given in Ass
Input1.linear.Model <- lm(Output1~USGG3M, data = AssignmentData)
summary(Input1.linear.Model)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG3M, data = AssignmentData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9374 -1.2115 -0.0528  1.2640  7.7048
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -11.72318    0.03137  -373.7   <2e-16 ***
## USGG3M        2.50756    0.00541   463.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.69 on 8298 degrees of freedom
## Multiple R-squared:  0.9628, Adjusted R-squared:  0.9628
## F-statistic: 2.148e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

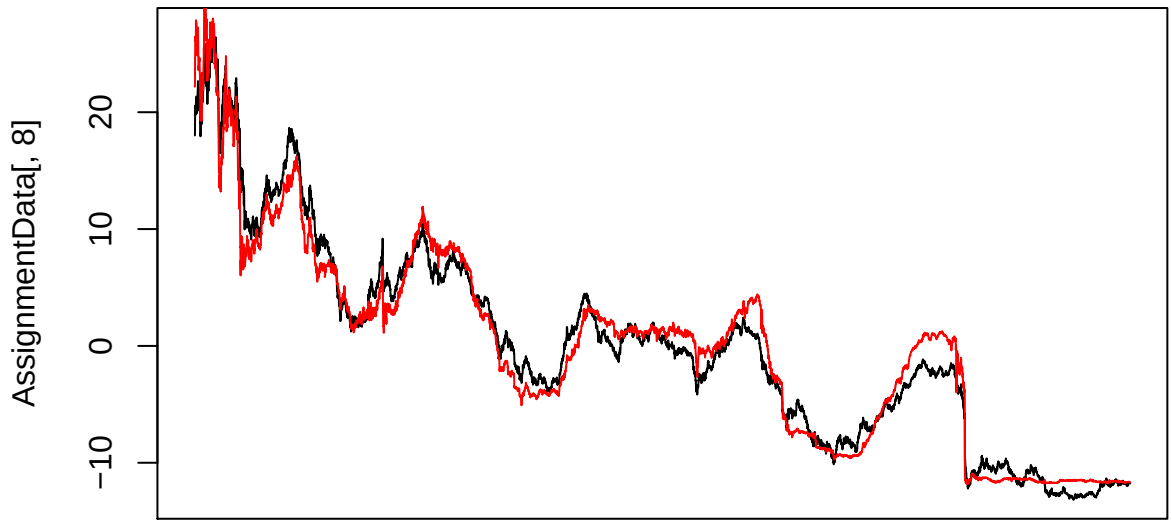
```r
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Input1.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.804438             2.857058
```

```r
#Coefficients of Input1.linear.Model
Input1.linear.Model$coefficients
```

```
## (Intercept)      USGG3M
##  -11.723184    2.507561
```

```r
#Plot of the Output with the fitted values from Input1.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Input1.linear.Model$fitted.values,col="red")
```




```r
#Input2 = USGG6M
Input2.linear.Model <- lm(Output1~USGG6M, data = AssignmentData)
summary(Input2.linear.Model)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG6M, data = AssignmentData)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.7529 -1.0385  0.0224  1.1443  4.1509
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -12.097528   0.026469  -457.0   <2e-16 ***
## USGG6M        2.497235   0.004445   561.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.403 on 8298 degrees of freedom
## Multiple R-squared:  0.9744, Adjusted R-squared:  0.9744
## F-statistic: 3.157e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

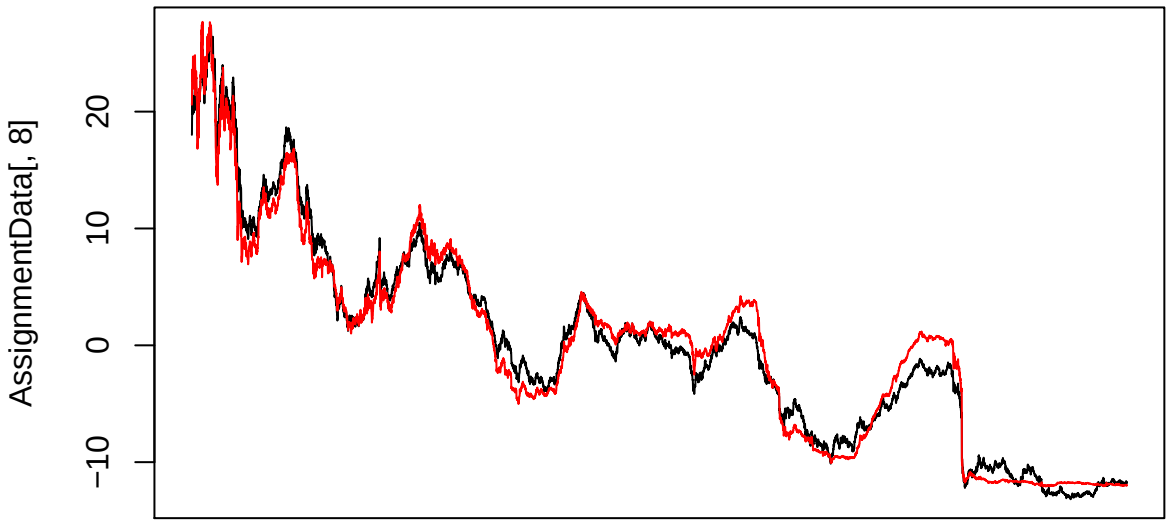
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Input2.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.804438             1.967321
```

```
#Coefficients of Input2.linear.Model
Input2.linear.Model$coefficients
```

```
## (Intercept)      USGG6M
##  -12.097528    2.497235
```

```
#Plot of the Output with the fitted values from Input2.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Input2.linear.Model$fitted.values,col="red")
```




```
#Input3 = USGG2YR
Input3.linear.Model <- lm(Output1~USGG2YR, data = AssignmentData)
summary(Input3.linear.Model)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG2YR, data = AssignmentData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.43277 -0.38356 -0.00578  0.43362  1.72564
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.055775   0.010031   -1302   <2e-16 ***
## USGG2YR       2.400449   0.001532    1567   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5087 on 8298 degrees of freedom
## Multiple R-squared:  0.9966, Adjusted R-squared:  0.9966
## F-statistic: 2.455e+06 on 1 and 8298 DF,  p-value: < 2.2e-16
```

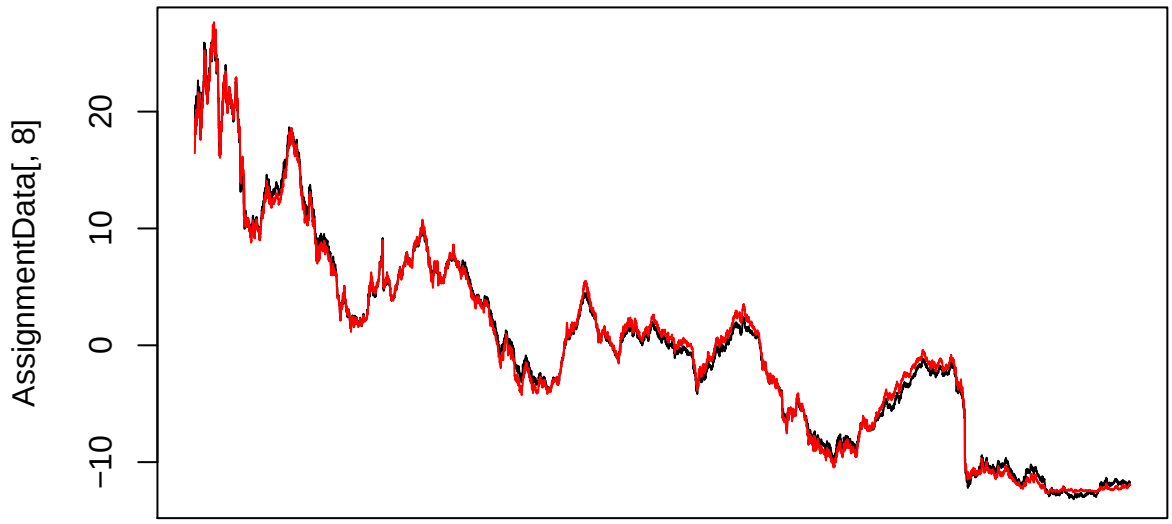
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Input3.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.8044379            0.2588092
```

```r
#Coefficients of Input3.linear.Model
Input3.linear.Model$coefficients
```

```
## (Intercept)     USGG2YR
##  -13.055775    2.400449
```

```r
#Plot of the Output with the fitted values from Input3.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Input3.linear.Model$fitted.values,col="red")
```




```r
#Input4 = USGG3YR
Input4.linear.Model <- lm(Output1~USGG3YR, data = AssignmentData)
summary(Input4.linear.Model)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG3YR, data = AssignmentData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0160 -0.2459  0.0325  0.2638  3.0666
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.861618   0.008214   -1688   <2e-16 ***
## USGG3YR       2.455793   0.001230    1996   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3996 on 8298 degrees of freedom
## Multiple R-squared:  0.9979, Adjusted R-squared:  0.9979
## F-statistic: 3.984e+06 on 1 and 8298 DF,  p-value: < 2.2e-16
```

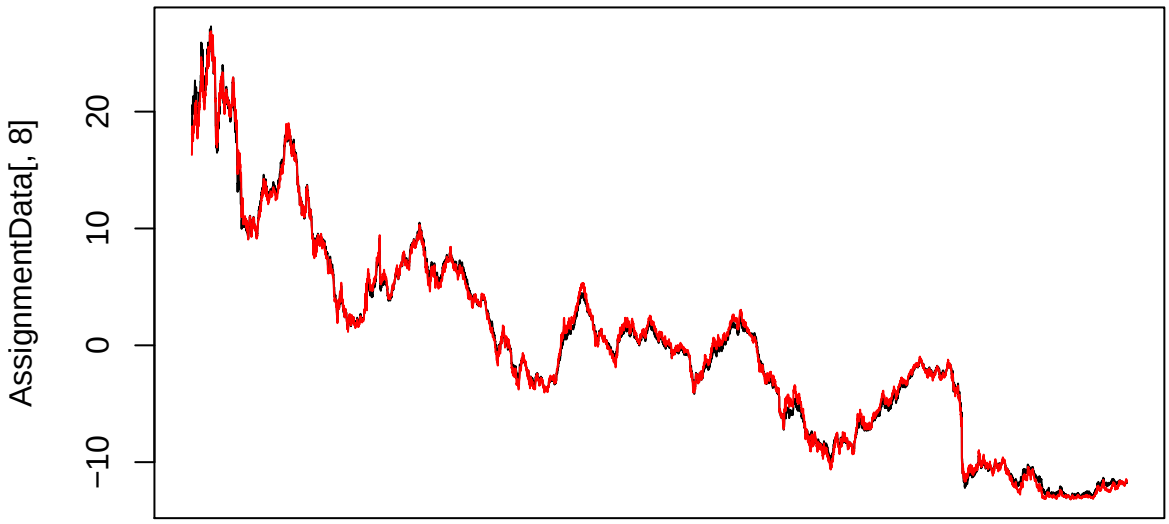
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Input4.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.804438             0.159657
```

```
#Coefficients of Input4.linear.Model
Input4.linear.Model$coefficients
```

```
## (Intercept)     USGG3YR
##  -13.861618    2.455793
```

```
#Plot of the Output with the fitted values from Input4.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Input4.linear.Model$fitted.values,col="red")
```




```
#Input5 = USGG5YR
Input5.linear.Model <- lm(Output1~USGG5YR, data = AssignmentData)
summary(Input5.linear.Model)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG5YR, data = AssignmentData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.6517 -0.6216 -0.0147  0.6523  3.1720
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -15.436649   0.017819  -866.3   <2e-16 ***
## USGG5YR       2.568742   0.002581   995.2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7989 on 8298 degrees of freedom
## Multiple R-squared:  0.9917, Adjusted R-squared:  0.9917
## F-statistic: 9.904e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

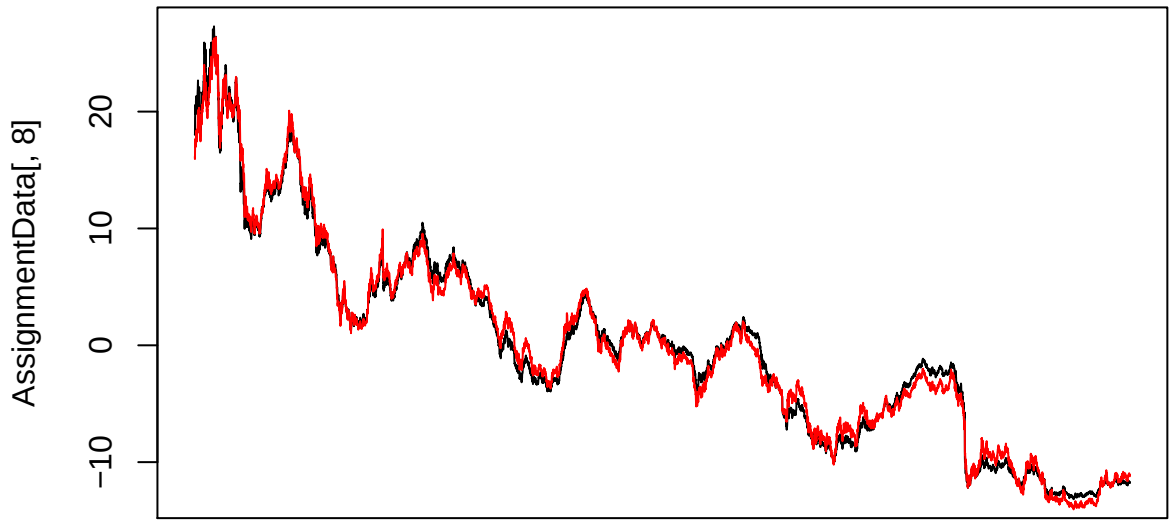
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Input5.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.8044379            0.6382572
```

```r
#Coefficients of Input5.linear.Model
Input5.linear.Model$coefficients
```

```
## (Intercept)     USGG5YR
##  -15.436649    2.568742
```

```r
#Plot of the Output with the fitted values from Input5.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Input5.linear.Model$fitted.values,col="red")
```




```r
#Input6 = USGG10YR
Input6.linear.Model <- lm(Output1~USGG10YR, data = AssignmentData)
summary(Input6.linear.Model)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG10YR, data = AssignmentData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0334 -1.2810 -0.1944  1.3561  4.2254
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -18.063370   0.039182  -461.0   <2e-16 ***
## USGG10YR      2.786991   0.005455   510.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.538 on 8298 degrees of freedom
## Multiple R-squared:  0.9692, Adjusted R-squared:  0.9692
## F-statistic: 2.61e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

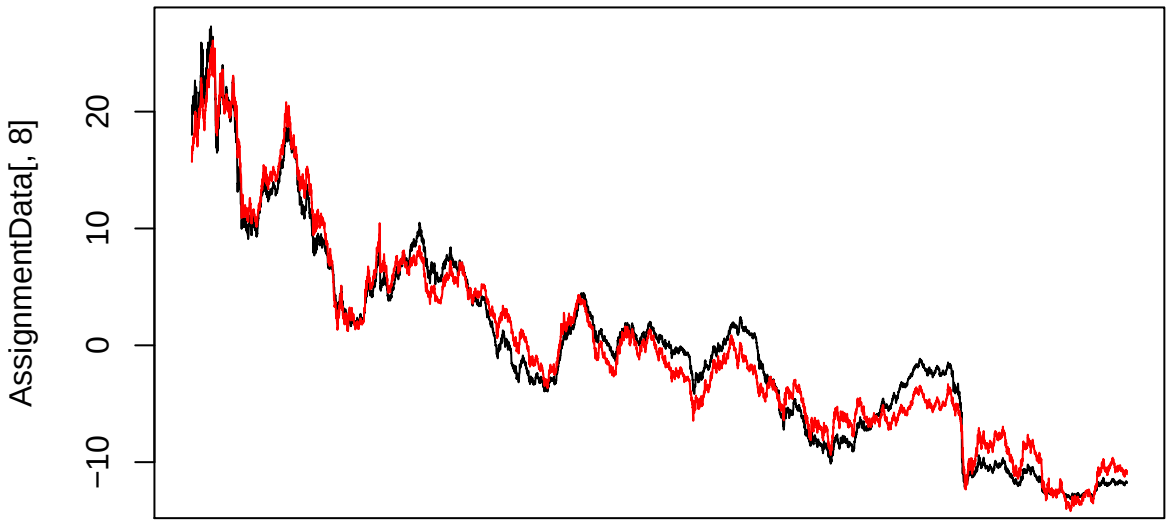
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Input6.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.804438             2.366752
```

```
#Coefficients of Input6.linear.Model
Input6.linear.Model$coefficients
```

```
## (Intercept)    USGG10YR
##  -18.063370    2.786991
```

```
#Plot of the Output with the fitted values from Input6.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Input6.linear.Model$fitted.values,col="red")
```




```
#Input7 = USGG30YR
Input7.linear.Model <- lm(Output1~USGG30YR, data = AssignmentData)
summary(Input7.linear.Model)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG30YR, data = AssignmentData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6447 -1.8426 -0.4731  1.9529  4.8734
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -21.08591    0.06560  -321.4   <2e-16 ***
## USGG30YR      3.06956    0.00886   346.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.229 on 8298 degrees of freedom
## Multiple R-squared:  0.9353, Adjusted R-squared:  0.9353
## F-statistic: 1.2e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

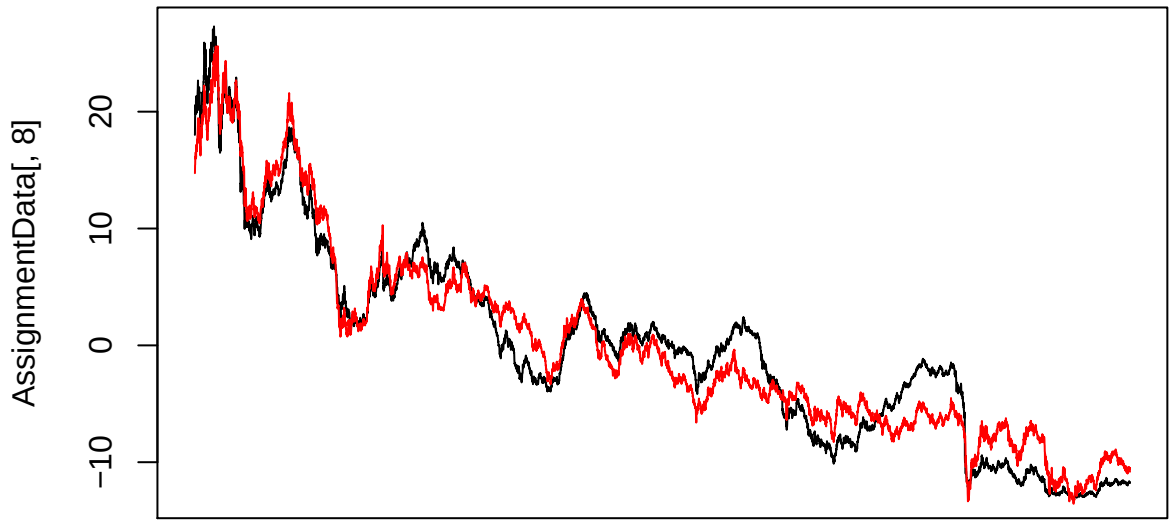
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Input7.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.804438             4.967286
```

```
#Coefficients of Input7.linear.Model
Input7.linear.Model$coefficients
```

```
## (Intercept)    USGG30YR
##  -21.085905    3.069561
```

```
#Plot of the Output with the fitted values from Input7.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Input7.linear.Model$fitted.values,col="red")
```



Collect all slopes and intercepts in one table and print this table. Try to do it in one line using apply() function.

```
#Create data frame with only inputs
AssignmentData.inputs.only <- AssignmentData[,1:7]
#Assign output column to y
y <- AssignmentData[,8]
#linmodel is a function, that runs lm() on x and pulls slope and intercept

linmodel <- function(x) {
  model <- lm(y~x)
  summary(model)$coefficients[1:2]
}
```

```
#use apply() to run linmodel on the input only dataframe, and input coefficients into table
apply(AssignmentData.inputs.only,2,linmodel)
```

```
##          USGG3M     USGG6M    USGG2YR    USGG3YR    USGG5YR   USGG10YR
## [1,] -11.723184 -12.097528 -13.055775 -13.861618 -15.436649 -18.063370
## [2,]   2.507561   2.497235   2.400449   2.455793   2.568742   2.786991
##        USGG30YR
## [1,] -21.085905
## [2,]   3.069561
```

Step.3

```
#Fit linear regression models using single output (column 8 Output1) as input and each of the original
#Input1 = USGG3M
Rev.Input1.linear.Model <- lm(USGG3M~Output1, data = AssignmentData)
summary(Rev.Input1.linear.Model)
```

```
##
## Call:
## lm(formula = USGG3M ~ Output1, data = AssignmentData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.86665 -0.51316 -0.04857  0.44260  3.07711
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.6751341  0.0072600   644.0   <2e-16 ***
## Output1     0.3839609  0.0008285   463.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6614 on 8298 degrees of freedom
## Multiple R-squared:  0.9628, Adjusted R-squared:  0.9628
## F-statistic: 2.148e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

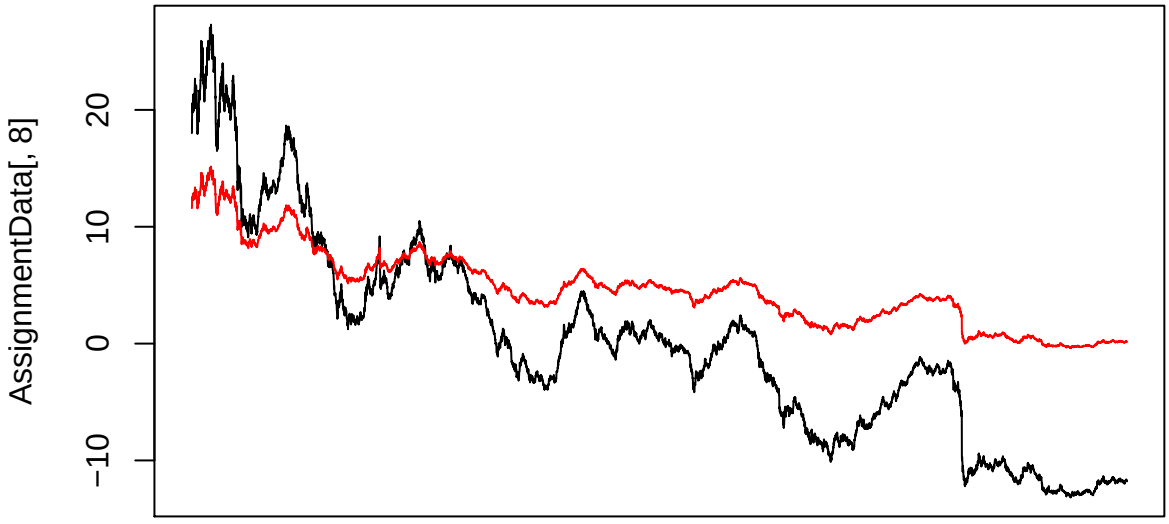
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Rev.Input1.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##           76.8044379            0.4374763
```

```
#Coefficients of Rev.Input1.linear.Model
Rev.Input1.linear.Model$coefficients
```

```
## (Intercept)     Output1
##   4.6751341   0.3839609
```

```r
#Plot of the Output with the fitted values from Rev.Input1.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Rev.Input1.linear.Model$fitted.values,col="red")
```




```r
#Input2 = USGG6M
Rev.Input2.linear.Model <- lm(USGG6M~Output1, data = AssignmentData)
summary(Rev.Input2.linear.Model)
```

```
##
## Call:
## lm(formula = USGG6M ~ Output1, data = AssignmentData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.51908 -0.44307 -0.04514  0.40876  1.71754
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.8443695  0.0060856   796.0   <2e-16 ***
## Output1     0.3901870  0.0006944   561.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5544 on 8298 degrees of freedom
## Multiple R-squared:  0.9744, Adjusted R-squared:  0.9744
```

```
## F-statistic: 3.157e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

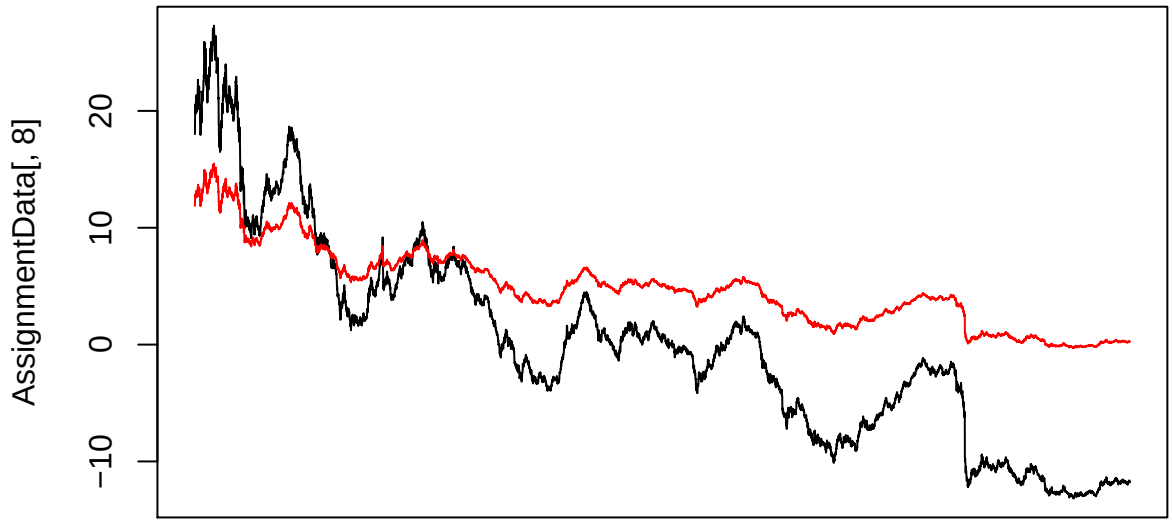
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Rev.Input2.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.8044379             0.3073892
```

```
#Coefficients of Rev.Input2.linear.Model
Rev.Input2.linear.Model$coefficients
```

```
## (Intercept)     Output1
##    4.844370    0.390187
```

```
#Plot of the Output with the fitted values from Rev.Input2.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Rev.Input2.linear.Model$fitted.values,col="red")
```




```
#Input3 = USGG2YR
Rev.Input3.linear.Model <- lm(USGG2YR~Output1, data = AssignmentData)
summary(Rev.Input3.linear.Model)
```

```
##
```

```
## Call:
## lm(formula = USGG2YR ~ Output1, data = AssignmentData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.69112 -0.18514  0.00422  0.15956  0.61596
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.438888   0.002322    2342   <2e-16 ***
## Output1     0.415185   0.000265    1567   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2116 on 8298 degrees of freedom
## Multiple R-squared:  0.9966, Adjusted R-squared:  0.9966
## F-statistic: 2.455e+06 on 1 and 8298 DF,  p-value: < 2.2e-16
```

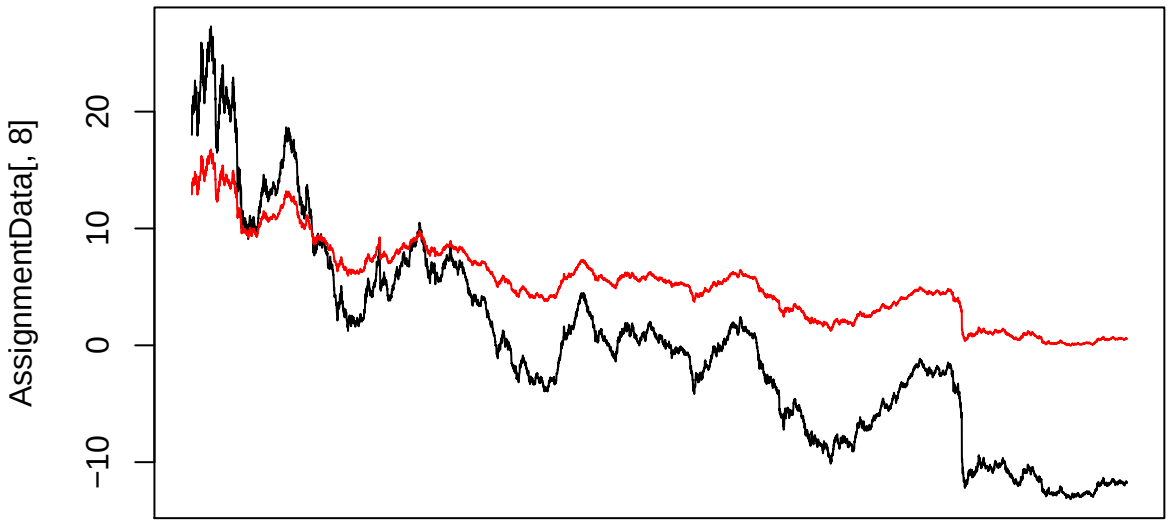
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Rev.Input3.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.804438             0.044764
```

```
#Coefficients of Rev.Input3.linear.Model
Rev.Input3.linear.Model$coefficients
```

```
## (Intercept)     Output1
##   5.4388879   0.4151851
```

```
#Plot of the Output with the fitted values from Rev.Input3.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Rev.Input3.linear.Model$fitted.values,col="red")
```




```
#Input4 = USGG3YR
Rev.Input4.linear.Model <- lm(USGG3YR~Output1, data = AssignmentData)
summary(Rev.Input4.linear.Model)
```

```
##
## Call:
## lm(formula = USGG3YR ~ Output1, data = AssignmentData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.23099 -0.10862 -0.01352  0.10095  0.83112
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.6444580  0.0017841    3164   <2e-16 ***
## Output1     0.4063541  0.0002036    1996   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1625 on 8298 degrees of freedom
## Multiple R-squared:  0.9979, Adjusted R-squared:  0.9979
## F-statistic: 3.984e+06 on 1 and 8298 DF,  p-value: < 2.2e-16
```

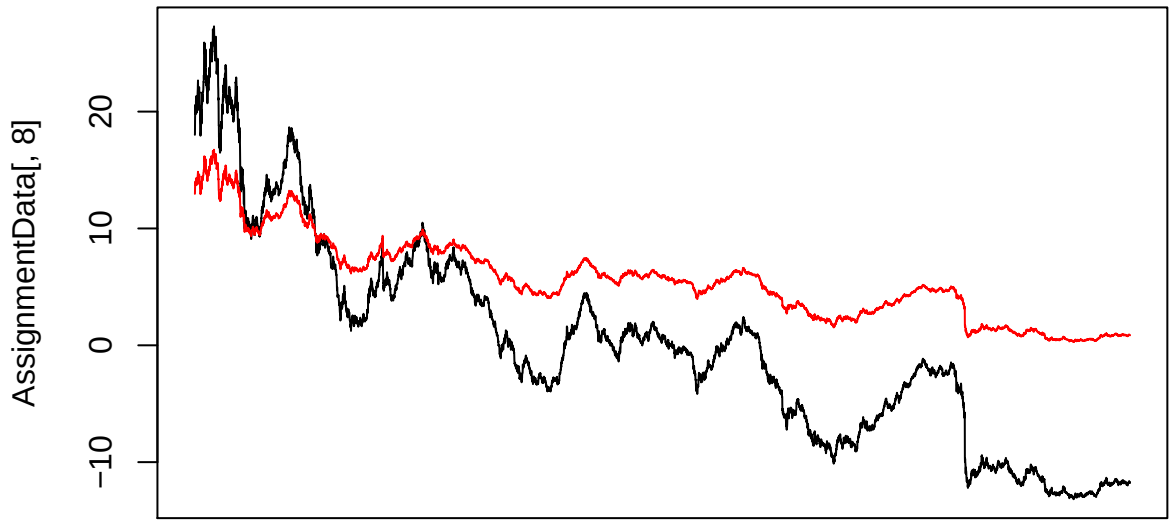
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Rev.Input4.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##          76.80443793           0.02641806
```

```
#Coefficients of Rev.Input4.linear.Model
Rev.Input4.linear.Model$coefficients
```

```
## (Intercept)     Output1
##   5.6444580   0.4063541
```

```
#Plot of the Output with the fitted values from Rev.Input4.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Rev.Input4.linear.Model$fitted.values,col="red")
```




```
#Input5 = USGG5YR
Rev.Input5.linear.Model <- lm(USGG5YR~Output1, data = AssignmentData)
summary(Rev.Input5.linear.Model)
```

```
##
## Call:
## lm(formula = USGG5YR ~ Output1, data = AssignmentData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.16714 -0.25482 -0.00012  0.25144  1.07477
```

```
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.0094207  0.0033996  1767.7   <2e-16 ***
## Output1     0.3860610  0.0003879   995.2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3097 on 8298 degrees of freedom
## Multiple R-squared:  0.9917, Adjusted R-squared:  0.9917
## F-statistic: 9.904e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

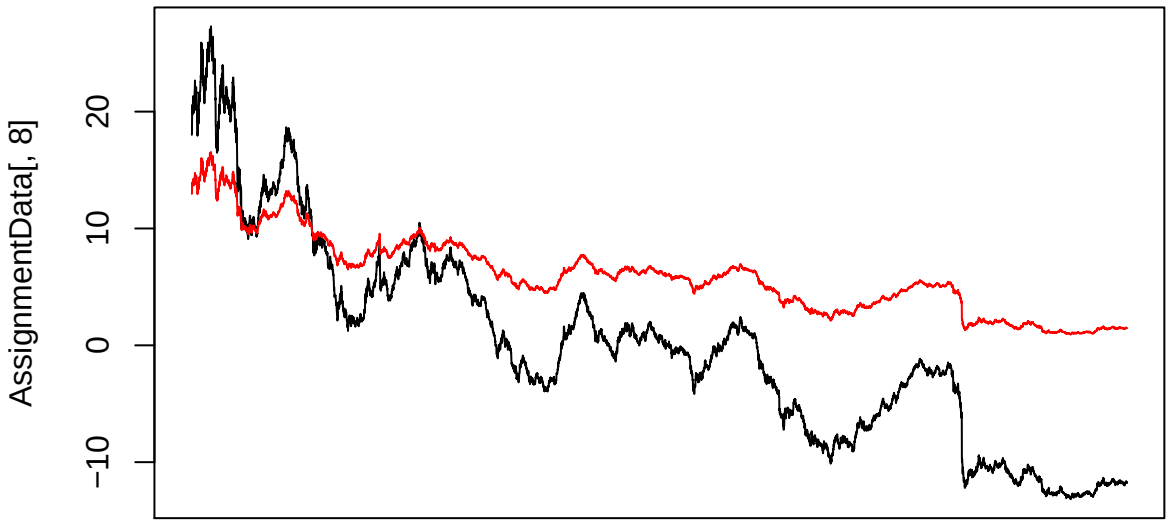
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Rev.Input5.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##          76.80443793           0.09592487
```

```
#Coefficients of Rev.Input5.linear.Model
Rev.Input5.linear.Model$coefficients
```

```
## (Intercept)     Output1
##    6.009421    0.386061
```

```
#Plot of the Output with the fitted values from Rev.Input5.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Rev.Input5.linear.Model$fitted.values,col="red")
```




```
#Input6 = USGG10YR
Rev.Input6.linear.Model <- lm(USGG10YR~Output1, data = AssignmentData)
summary(Rev.Input6.linear.Model)
```

```
##
## Call:
## lm(formula = USGG10YR ~ Output1, data = AssignmentData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.23614 -0.46180  0.07829  0.44858  1.24192
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.4813160  0.0059649  1086.6   <2e-16 ***
## Output1     0.3477544  0.0006807   510.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5434 on 8298 degrees of freedom
## Multiple R-squared:  0.9692, Adjusted R-squared:  0.9692
## F-statistic: 2.61e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

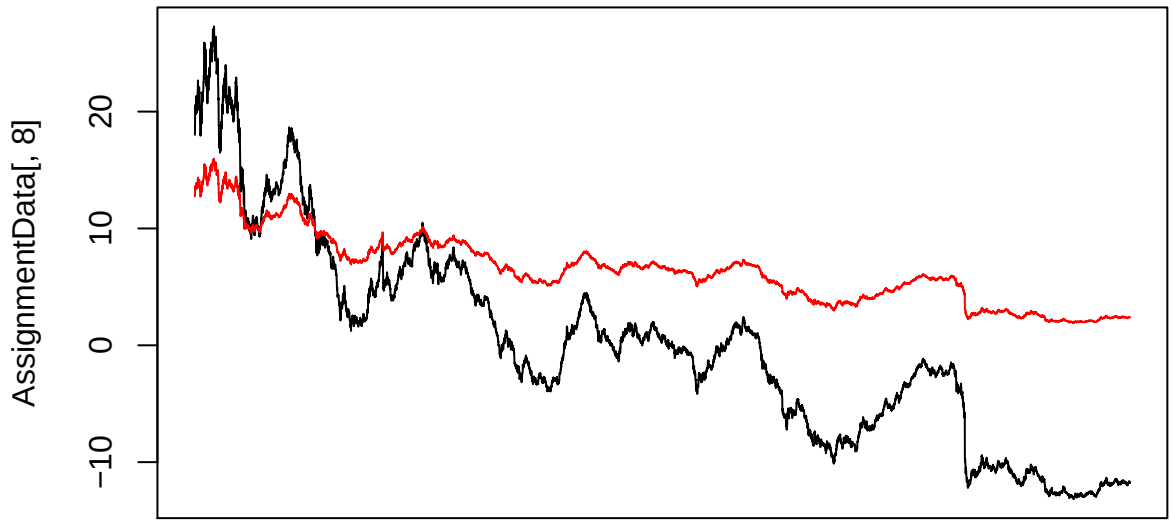
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Rev.Input6.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##            76.804438             0.295318
```

```r
#Coefficients of Rev.Input6.linear.Model
Rev.Input6.linear.Model$coefficients
```

```
## (Intercept)     Output1
##   6.4813160   0.3477544
```

```r
#Plot of the Output with the fitted values from Rev.Input6.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Rev.Input6.linear.Model$fitted.values,col="red")
```




```r
#Input7 = USGG30YR
Rev.Input7.linear.Model <- lm(USGG30YR~Output1, data = AssignmentData)
summary(Rev.Input7.linear.Model)
```

```
##
## Call:
## lm(formula = USGG30YR ~ Output1, data = AssignmentData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5436 -0.5683  0.1237  0.5599  1.4505
```

```
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.8693554  0.0077077   891.2   <2e-16 ***
## Output1     0.3047124  0.0008795   346.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7022 on 8298 degrees of freedom
## Multiple R-squared:  0.9353, Adjusted R-squared:  0.9353
## F-statistic: 1.2e+05 on 1 and 8298 DF,  p-value: < 2.2e-16
```

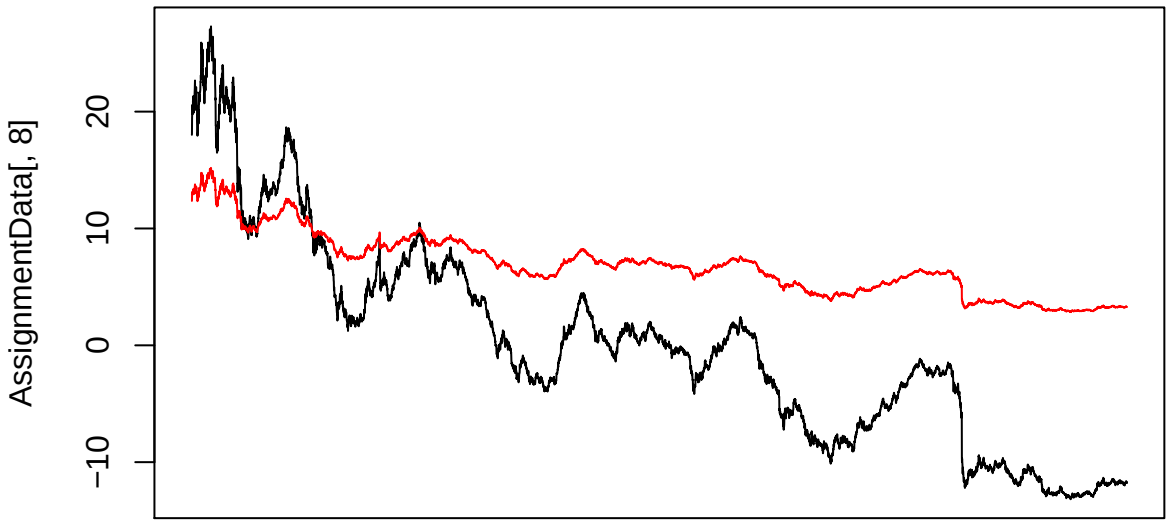
```
# total and unexplained variances
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(Rev.Input7.linear.Model)$sigma^2)
```

```
##       Total.Variance Unexplained.Variance
##           76.8044379            0.4930978
```

```
#Coefficients of Rev.Input7.linear.Model
Rev.Input7.linear.Model$coefficients
```

```
## (Intercept)     Output1
##   6.8693554   0.3047124
```

```
#Plot of the Output with the fitted values from Rev.Input7.linear.Model
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(Rev.Input7.linear.Model$fitted.values,col="red")
```



Collect all slopes and intercepts in one table and print this table.

```
#Create data frame with only outputs
AssignmentData.outputs.only <- AssignmentData[,1:7]
#Assign input column to input
input <- AssignmentData[,8]
#linmodel is a function, that runs lm() on x and pulls slope and intercept

linmodel <- function(x) {
  model <- lm(x~input)
  summary(model)$coefficients[1:2]
}
#use apply() to run linmodel on the input only dataframe, and input coefficients into table
apply(AssignmentData.inputs.only,2,linmodel)
```

```
##         USGG3M   USGG6M   USGG2YR   USGG3YR  USGG5YR  USGG10YR  USGG30YR
## [1,] 4.6751341 4.844370 5.4388879 5.6444580 6.009421 6.4813160 6.8693554
## [2,] 0.3839609 0.390187 0.4151851 0.4063541 0.386061 0.3477544 0.3047124
```

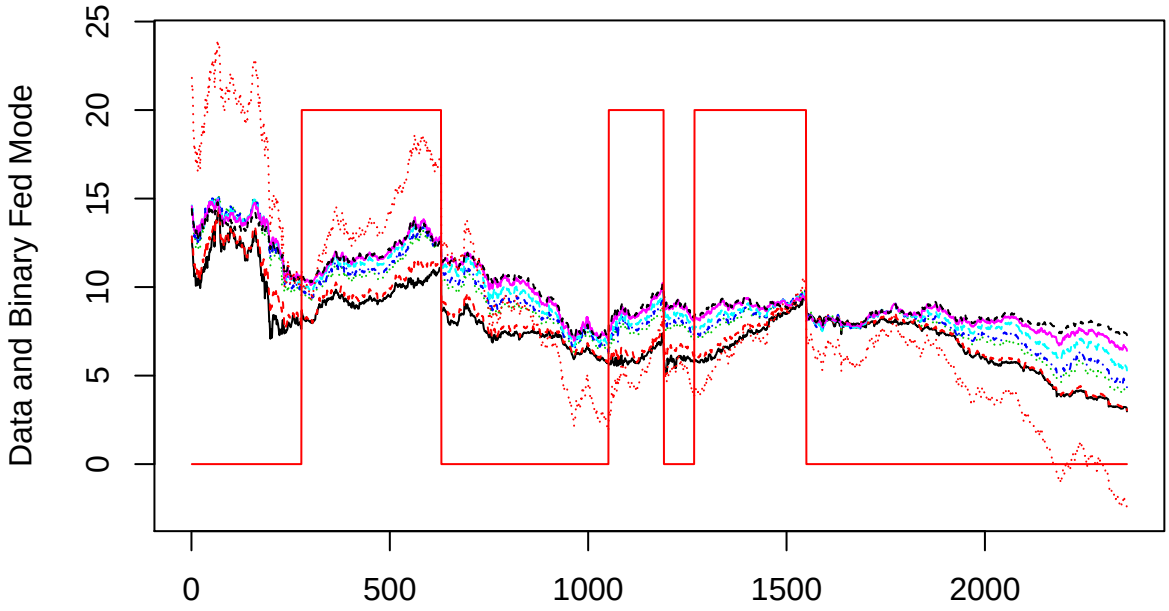
Step 4.

Estimate logistic regression using all inputs and the data on FED tightening and easing cycles.

```
AssignmentDataLogistic<-data.matrix(AssignmentData,rownames.force="automatic")
# Create columns of easing periods (as 0s) and tightening periods (as 1s)
EasingPeriods<-AssignmentDataLogistic[,9]
EasingPeriods[AssignmentDataLogistic[,9]==1]<-0
TighteningPeriods<-AssignmentDataLogistic[,10]
# Check easing and tightening periods
cbind(EasingPeriods,TighteningPeriods)[c(550:560,900:910,970:980),]
```

```
##            EasingPeriods TighteningPeriods
## 3/29/1983             NA                NA
## 3/30/1983             NA                NA
## 3/31/1983             NA                NA
## 4/4/1983              NA                 1
## 4/5/1983              NA                 1
## 4/6/1983              NA                 1
## 4/7/1983              NA                 1
## 4/8/1983              NA                 1
## 4/11/1983             NA                 1
## 4/12/1983             NA                 1
## 4/13/1983             NA                 1
## 8/27/1984             NA                 1
## 8/28/1984             NA                 1
## 8/29/1984             NA                 1
## 8/30/1984             NA                 1
## 8/31/1984             NA                 1
## 9/4/1984              NA                NA
## 9/5/1984              NA                NA
## 9/6/1984              NA                NA
## 9/7/1984              NA                NA
## 9/10/1984             NA                NA
## 9/11/1984             NA                NA
## 12/10/1984             0                NA
## 12/11/1984             0                NA
## 12/12/1984             0                NA
## 12/13/1984             0                NA
## 12/14/1984             0                NA
## 12/17/1984             0                NA
## 12/18/1984             0                NA
## 12/19/1984             0                NA
## 12/20/1984             0                NA
## 12/21/1984             0                NA
## 12/24/1984             0                NA
```

```
#Remove the periods of neither easing nor tightening.
All.NAs<-is.na(EasingPeriods)&is.na(TighteningPeriods)
AssignmentDataLogistic.EasingTighteningOnly<-AssignmentDataLogistic
AssignmentDataLogistic.EasingTighteningOnly[,9]<-EasingPeriods
AssignmentDataLogistic.EasingTighteningOnly<-AssignmentDataLogistic.EasingTighteningOnly[!All.NAs,]
AssignmentDataLogistic.EasingTighteningOnly[is.na(AssignmentDataLogistic.EasingTighteningOnly[,10]),10]<
# Binary output for logistic regression is now in column 10
# Plot the data and the binary output variable representing easing (0) and tightening (1) periods.
matplot(AssignmentDataLogistic.EasingTighteningOnly[,-c(9,10)],type="l",ylab="Data and Binary Fed Mode"
lines(AssignmentDataLogistic.EasingTighteningOnly[,10]*20,col="red")
```

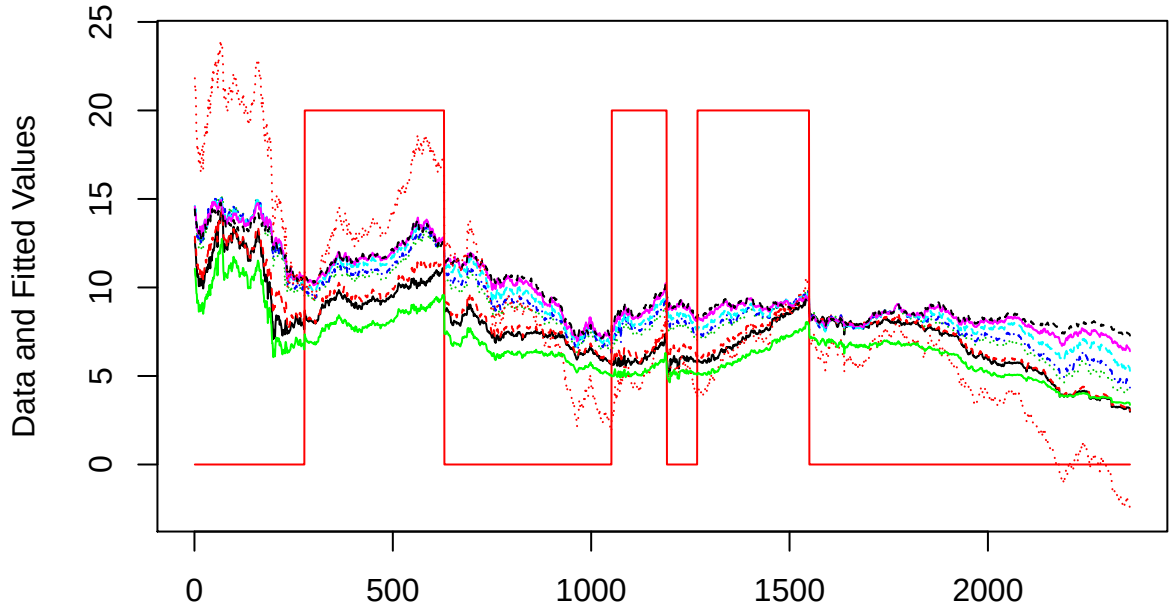



```
#Estimate logistic regression with 3M yields as predictors for easing/tightening output.
LogisticModel.TighteningEasing_3M<-glm(AssignmentDataLogistic.EasingTighteningOnly[,10]~
                                       AssignmentDataLogistic.EasingTighteningOnly[,1],family=binomial(li
summary(LogisticModel.TighteningEasing_3M)
```

```
##
## Call:
## glm(formula = AssignmentDataLogistic.EasingTighteningOnly[, 10] ~
##     AssignmentDataLogistic.EasingTighteningOnly[, 1], family = binomial(link = logit))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4239  -0.9014  -0.7737   1.3548   1.6743
##
## Coefficients:
##                                                 Estimate Std. Error
## (Intercept)                                     -2.15256    0.17328
## AssignmentDataLogistic.EasingTighteningOnly[, 1]  0.18638    0.02144
##                                                 z value Pr(>|z|)
## (Intercept)                                     -12.422   <2e-16 ***
## AssignmentDataLogistic.EasingTighteningOnly[, 1]   8.694   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 2983.5  on 2357  degrees of freedom
## Residual deviance: 2904.8  on 2356  degrees of freedom
## AIC: 2908.8
##
## Number of Fisher Scoring iterations: 4
```

```
matplot(AssignmentDataLogistic.EasingTighteningOnly[,-c(9,10)],type="l",ylab="Data and Fitted Values")
lines(AssignmentDataLogistic.EasingTighteningOnly[,10]*20,col="red")
lines(LogisticModel.TighteningEasing_3M$fitted.values*20,col="green")
```




```
#Now use all inputs as predictors for logistic regression.
LogisticModel.TighteningEasing_All <- glm(AssignmentDataLogistic.EasingTighteningOnly[,10]~
                                    AssignmentDataLogistic.EasingTighteningOnly[,1:7],family=binomial

summary(LogisticModel.TighteningEasing_All)$aic
```

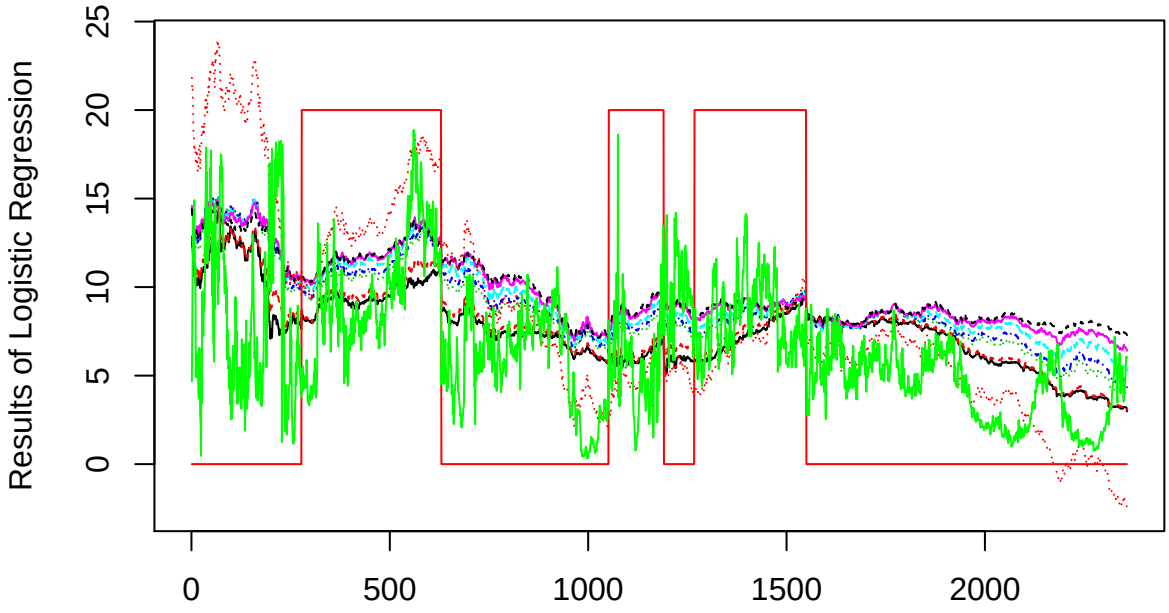
```
## [1] 2645.579
```

```
summary(LogisticModel.TighteningEasing_All)$coefficients[,c(1,4)]
```

```
##                                                    Estimate
## (Intercept)                                      -4.7551928
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG3M   -3.3456116
```

```
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG6M    4.1558535
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG2YR   3.9460296
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG3YR  -3.4642455
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG5YR  -3.2115319
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG10YR -0.9705444
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG30YR  3.3253517
##                                                                Pr(>|z|)
## (Intercept)                                                2.784283e-28
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG3M   4.073045e-36
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG6M   1.422964e-28
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG2YR  1.751687e-07
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG3YR  2.080617e-04
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG5YR  3.786229e-05
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG10YR 3.202140e-01
## AssignmentDataLogistic.EasingTighteningOnly[, 1:7]USGG30YR 6.036041e-08
```

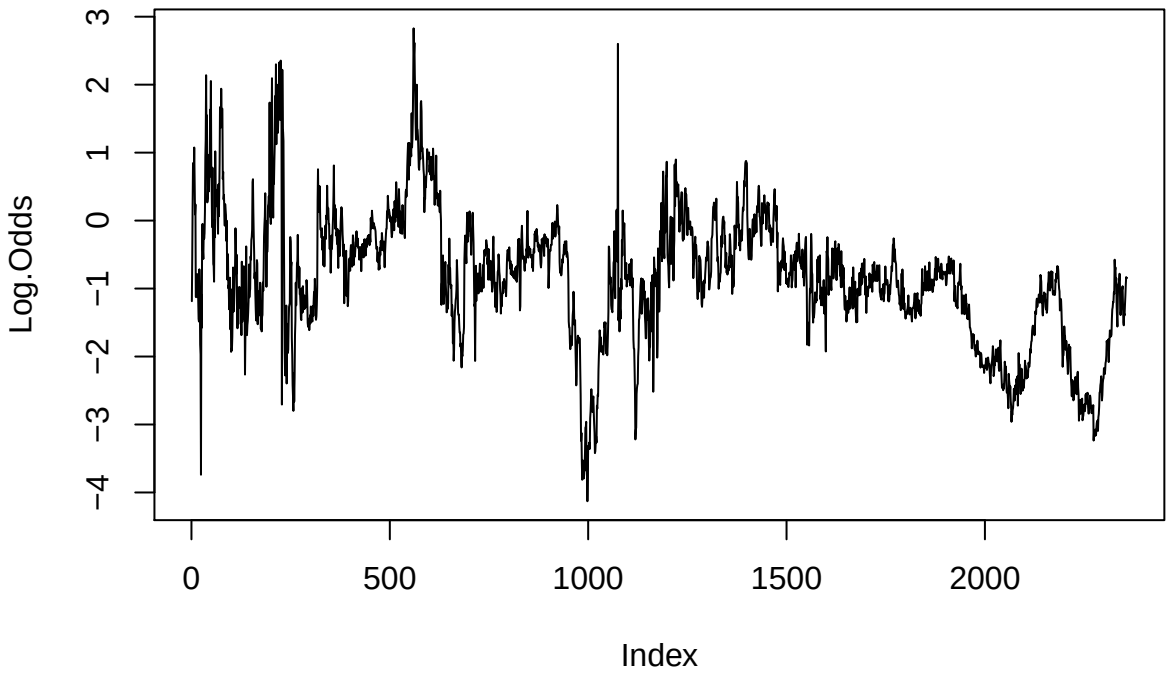
```
matplot(AssignmentDataLogistic.EasingTighteningOnly[,-c(9,10)],type="l",ylab="Results of Logistic Regres
lines(AssignmentDataLogistic.EasingTighteningOnly[,10]*20,col="red")
lines(LogisticModel.TighteningEasing_All$fitted.values*20,col="green")
```



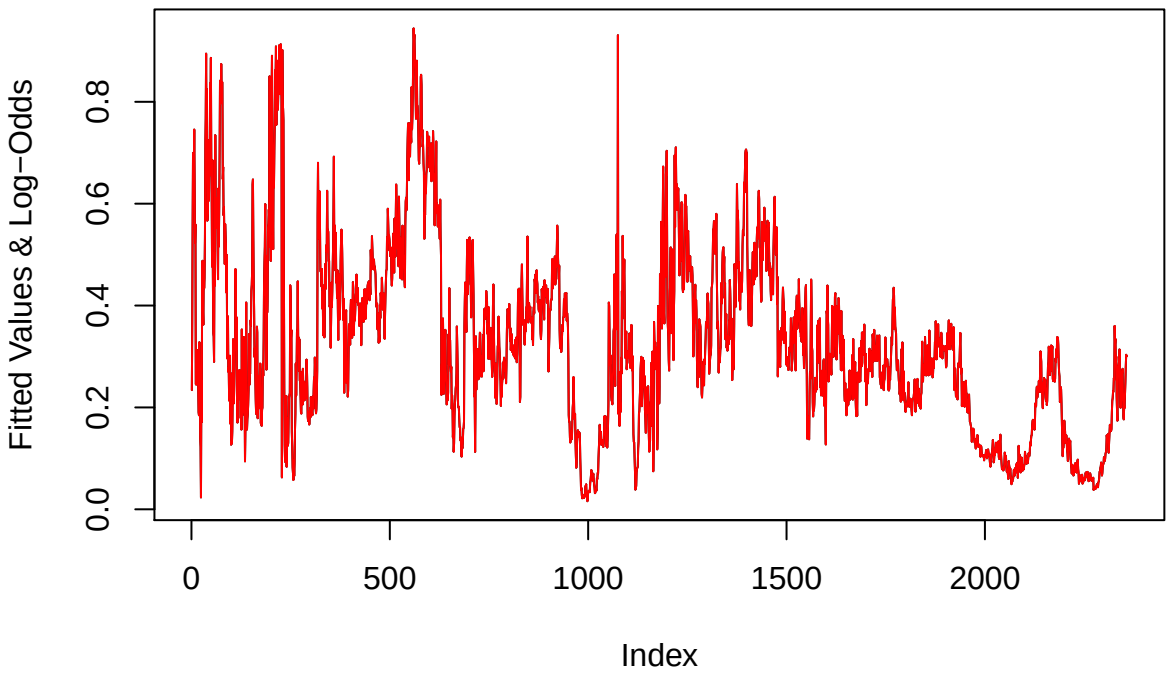
Interpret the coefficients of the model and the fitted values. The logistic regression coefficients of each variable represent the change in log odds of the outcome for a one unit increase in the predictor variable. For example, for a 1 unit increase in 3month yield, shows us the multiplicitive effect on log odds , e^-3.354 = 0.035, this would decrease the odds ratio by 3.5%. However, it does seem odd that the coefficients all have different positive and negative signs, despite the fact that they are all so correlated.

The fitted values are the probabilities of tightening or easing given the data in the yields on a given day.

```r
#Calculate and plot log-odds and probabilities. Compare probabilities with fitted values.
Log.Odds<-predict(LogisticModel.TighteningEasing_All)
plot(Log.Odds,type="l")
```




```r
Probabilities<-1/(exp(-Log.Odds)+1)
plot(LogisticModel.TighteningEasing_All$fitted.values,type="l",ylab="Fitted Values & Log-Odds")
lines(Probabilities,col="red")
```



Step 5. Compare linear regression models with different combinations of predictors. Select the best combination.

```
AssignmentDataRegressionComparison<-data.matrix(AssignmentData[,-c(9,10)],rownames.force="automatic")
AssignmentDataRegressionComparison<-AssignmentData[,-c(9,10)]
#
RegressionModelComparison.Full <- lm(Output1~., data = AssignmentDataRegressionComparison)
summary(RegressionModelComparison.Full)
```

```
##
## Call:
## lm(formula = Output1 ~ ., data = AssignmentDataRegressionComparison)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -5.108e-09 -3.430e-10 -1.000e-12  3.340e-10  5.085e-09
##
## Coefficients:
##               Estimate Std. Error    t value Pr(>|t|)
## (Intercept) -1.490e+01  1.057e-10 -1.410e+11   <2e-16 ***
## USGG3M       3.840e-01  9.860e-11  3.894e+09   <2e-16 ***
## USGG6M       3.902e-01  1.500e-10  2.601e+09   <2e-16 ***
## USGG2YR      4.152e-01  2.569e-10  1.616e+09   <2e-16 ***
## USGG3YR      4.064e-01  3.299e-10  1.232e+09   <2e-16 ***
## USGG5YR      3.861e-01  2.618e-10  1.474e+09   <2e-16 ***
## USGG10YR     3.478e-01  2.800e-10  1.242e+09   <2e-16 ***
```

```
## USGG30YR     3.047e-01  1.566e-10  1.945e+09   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.562e-09 on 8292 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 3.73e+22 on 7 and 8292 DF,  p-value: < 2.2e-16
```

```
summary(RegressionModelComparison.Full)$coefficients
```

```
##                Estimate   Std. Error       t value Pr(>|t|)
## (Intercept) -14.9041591 1.056850e-10 -141024294891        0
## USGG3M        0.3839609 9.860401e-11    3893968285        0
## USGG6M        0.3901870 1.500111e-10    2601053702        0
## USGG2YR       0.4151851 2.569451e-10    1615851177        0
## USGG3YR       0.4063541 3.299038e-10    1231735395        0
## USGG5YR       0.3860610 2.618339e-10    1474449865        0
## USGG10YR      0.3477544 2.800269e-10    1241860763        0
## USGG30YR      0.3047124 1.566487e-10    1945195584        0
```

```
c(summary.lm(RegressionModelComparison.Full)$r.squared,summary.lm(RegressionModelComparison.Full)$adj.r
```

```
## [1] 1 1
```

```
summary.lm(RegressionModelComparison.Full)$df
```

```
## [1]    8 8292    8
```

Intepret the fitted model. How good is the fit? How significant are the parameters? The model has a perfect R squared and adjusted R squared since they are both zero. The fit is perfectly overfitted. The parameters all have a Pr(t) equal to zero, so they are all statistically significant. We reject the null hypothesis that the slopes are equal to zero

```
#Estimate the Null model by including only intercept.
RegressionModelComparison.Null <- lm(Output1~1, data = AssignmentDataRegressionComparison)
summary(RegressionModelComparison.Null)$coefficients
```

```
##                 Estimate Std. Error      t value Pr(>|t|)
## (Intercept) 1.420082e-11 0.09619536 1.476248e-10        1
```

```
c(summary.lm(RegressionModelComparison.Null)$r.squared,summary.lm(RegressionModelComparison.Null)$adj.r
```

```
## [1] 0 0
```

```
summary.lm(RegressionModelComparison.Null)$df
```

```
## [1]    1 8299    1
```

Why summary(RegressionModelComparison.Null) does not show R2R2? The null model only has the intercept and does not have any predictors. Since it is intercept only, all the total sum of squares comes from the reisduals and the model sum of squares equals zero. Since R^2 = SSM/SST=0/637400=0

```r
#Compare models pairwise using anova()
anova(RegressionModelComparison.Full,RegressionModelComparison.Null)
```

```
## Analysis of Variance Table
##
## Model 1: Output1 ~ USGG3M + USGG6M + USGG2YR + USGG3YR + USGG5YR + USGG10YR +
##     USGG30YR
## Model 2: Output1 ~ 1
##   Res.Df    RSS Df Sum of Sq        F    Pr(>F)
## 1   8292      0
## 2   8299 637400 -7   -637400 3.73e+22 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interpret the results of anova(). Since the Pr(F) is statistically significant, we reject the null hypothesis that the two models are the same. The full model explains all the variation, the null model explains none.

```r
#Repeat the analysis for different combinations of input variables and select the one you think is the
drop1(RegressionModelComparison.Full)
```

```
## Warning: attempting model selection on an essentially perfect fit is
## nonsense
```

```
## Single term deletions
##
## Model:
## Output1 ~ USGG3M + USGG6M + USGG2YR + USGG3YR + USGG5YR + USGG10YR +
##     USGG30YR
##          Df Sum of Sq    RSS     AIC
## <none>                 0.000 -336591
## USGG3M    1    37.016 37.016  -44911
## USGG6M    1    16.516 16.516  -51609
## USGG2YR   1     6.374  6.374  -59512
## USGG3YR   1     3.704  3.704  -64018
## USGG5YR   1     5.307  5.307  -61032
## USGG10YR  1     3.765  3.765  -63882
## USGG30YR  1     9.237  9.237  -56433
```

```r
#Warning message:
#attempting model selection on an essentially perfect fit is nonsense #This is telling me that you cann

#Using add1()
#start with the null model
summary.lm(RegressionModelComparison.Null)
```

```
##
## Call:
## lm(formula = Output1 ~ 1, data = AssignmentDataRegressionComparison)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -13.173  -6.509  -0.415   4.860  27.298
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.42e-11   9.62e-02       0        1
##
## Residual standard error: 8.764 on 8299 degrees of freedom
```

```
myScope <- names(AssignmentData.inputs.only)
add1(RegressionModelComparison.Null, scope=myScope)
```

```
## Single term additions
##
## Model:
## Output1 ~ 1
##          Df Sum of Sq    RSS    AIC
## <none>                637400  36033
## USGG3M    1    613692  23708   8715
## USGG6M    1    621075  16325   5618
## USGG2YR   1    635252   2148 -11217
## USGG3YR   1    636075   1325 -15226
## USGG5YR   1    632104   5296  -3725
## USGG10YR  1    617761  19639   7153
## USGG30YR  1    596181  41219  13306
```

```
#adding USGG3YR
Null.3yr <- lm(Output1~USGG3YR, data = AssignmentDataRegressionComparison)
summary(Null.3yr)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG3YR, data = AssignmentDataRegressionComparison)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0160 -0.2459  0.0325  0.2638  3.0666
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.861618   0.008214   -1688   <2e-16 ***
## USGG3YR       2.455793   0.001230    1996   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3996 on 8298 degrees of freedom
## Multiple R-squared:  0.9979, Adjusted R-squared:  0.9979
## F-statistic: 3.984e+06 on 1 and 8298 DF,  p-value: < 2.2e-16
```

```
#Returns an R-squared of 0.9979, equal to Adjusted R-squared
#reset myScope without 3 yr
myScope <- myScope[-which(myScope=="USGG3YR")]
add1(Null.3yr, scope=myScope)
```

```
## Single term additions
##
## Model:
## Output1 ~ USGG3YR
##          Df Sum of Sq     RSS    AIC
## <none>                1324.83 -15226
## USGG3M    1    407.98  916.85 -18279
## USGG6M    1    270.17 1054.66 -17117
## USGG2YR   1     82.93 1241.91 -15761
## USGG5YR   1     20.94 1303.89 -15356
## USGG10YR  1     64.05 1260.78 -15636
## USGG30YR  1    100.79 1224.04 -15881
```

```
#add USGG3M
Null.3yr.3mo <- lm(Output1~USGG3YR+USGG3M, data = AssignmentDataRegressionComparison)
summary(Null.3yr.3mo)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG3YR + USGG3M, data = AssignmentDataRegressionComparison)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.19592 -0.25503  0.01678  0.24577  1.77420
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -13.671658   0.007515 -1819.36   <2e-16 ***
## USGG3YR       2.171934   0.004782   454.14   <2e-16 ***
## USGG3M        0.302081   0.004972    60.76   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3324 on 8297 degrees of freedom
## Multiple R-squared:  0.9986, Adjusted R-squared:  0.9986
## F-statistic: 2.88e+06 on 2 and 8297 DF,  p-value: < 2.2e-16
```

```
#Returns an R-squared of 0.9986, equal to Adjusted R-squared
#reset myScope without 3 mo
myScope <- myScope[-which(myScope=="USGG3M")]
add1(Null.3yr.3mo, scope=myScope)
```

```
## Single term additions
##
## Model:
## Output1 ~ USGG3YR + USGG3M
##          Df Sum of Sq    RSS    AIC
## <none>                916.85 -18279
## USGG6M    1    139.91 776.95 -19652
## USGG2YR   1    176.99 739.86 -20058
## USGG5YR   1    736.49 180.36 -31773
## USGG10YR  1    864.29  52.56 -42007
## USGG30YR  1    863.62  53.23 -41902
```

```r
#add USGG10YR
Null.3yr.3mo.10yr <- lm(Output1~USGG3YR+USGG3M+USGG10YR, data = AssignmentDataRegressionComparison)
summary(Null.3yr.3mo.10yr)
```

```
##
## Call:
## lm(formula = Output1 ~ USGG3YR + USGG3M + USGG10YR, data = AssignmentDataRegressionComparison)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.42997 -0.04207  0.00512  0.04825  0.69394
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -14.723583   0.003369 -4370.4   <2e-16 ***
## USGG3YR       1.120774   0.003068   365.3   <2e-16 ***
## USGG3M        0.705571   0.001616   436.7   <2e-16 ***
## USGG10YR      0.786689   0.002130   369.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0796 on 8296 degrees of freedom
## Multiple R-squared:  0.9999, Adjusted R-squared:  0.9999
## F-statistic: 3.353e+07 on 3 and 8296 DF,  p-value: < 2.2e-16
```

```r
#Returns an R-squared of 0.9999, equal to Adjusted R-squared
```

Since the R-squared is so high with just one predictor in the model (3YR), and it has the same value as the adjusted R-squared. It makes no sense to add more predictors, to go from an R-squared of 0.9979 to 0.9986.

Step 6.

Perform rolling window analysis of the yields data. Use package zoo for rolling window analysis.

```r
#Set the window width and window shift parameters for rolling window.
Window.width<-20; Window.shift<-5
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 3.3.2
```

```
##
## Attaching package: 'zoo'
```

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

```r
all.means<-rollapply(AssignmentDataRegressionComparison,width=Window.width,by=Window.shift,by.column=TRU
head(all.means,10)
```

```
##        USGG3M  USGG6M USGG2YR USGG3YR USGG5YR USGG10YR USGG30YR  Output1
##  [1,] 15.0405 14.0855 13.2795 12.9360 12.7825  12.5780  12.1515 20.14842
##  [2,] 15.1865 14.1440 13.4855 13.1085 12.9310  12.7370  12.3370 20.55208
##  [3,] 15.2480 14.2755 13.7395 13.3390 13.1500  12.9480  12.5500 21.04895
##  [4,] 14.9345 14.0780 13.7750 13.4765 13.2385  13.0515  12.6610 21.02611
##  [5,] 14.7545 14.0585 13.9625 13.6890 13.4600  13.2295  12.8335 21.31356
##  [6,] 14.6025 14.0115 14.0380 13.7790 13.5705  13.3050  12.8890 21.39061
##  [7,] 14.0820 13.5195 13.8685 13.6710 13.4815  13.1880  12.7660 20.77200
##  [8,] 13.6255 13.0635 13.6790 13.5735 13.4270  13.1260  12.6950 20.23626
##  [9,] 13.2490 12.6810 13.5080 13.4575 13.3680  13.0770  12.6470 19.76988
## [10,] 12.9545 12.4225 13.4140 13.4240 13.3850  13.1115  12.6755 19.53054
```

```
# Create points at which rolling means are calculated
Count<-1:length(AssignmentDataRegressionComparison[,1])
Rolling.window.matrix<-rollapply(Count,width=Window.width,by=Window.shift,by.column=FALSE,
          FUN=function(z) z)
Rolling.window.matrix[1:10,]
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
##  [1,]    1    2    3    4    5    6    7    8    9    10    11    12    13
##  [2,]    6    7    8    9   10   11   12   13   14    15    16    17    18
##  [3,]   11   12   13   14   15   16   17   18   19    20    21    22    23
##  [4,]   16   17   18   19   20   21   22   23   24    25    26    27    28
##  [5,]   21   22   23   24   25   26   27   28   29    30    31    32    33
##  [6,]   26   27   28   29   30   31   32   33   34    35    36    37    38
##  [7,]   31   32   33   34   35   36   37   38   39    40    41    42    43
##  [8,]   36   37   38   39   40   41   42   43   44    45    46    47    48
##  [9,]   41   42   43   44   45   46   47   48   49    50    51    52    53
## [10,]   46   47   48   49   50   51   52   53   54    55    56    57    58
##       [,14] [,15] [,16] [,17] [,18] [,19] [,20]
##  [1,]    14    15    16    17    18    19    20
##  [2,]    19    20    21    22    23    24    25
##  [3,]    24    25    26    27    28    29    30
##  [4,]    29    30    31    32    33    34    35
##  [5,]    34    35    36    37    38    39    40
##  [6,]    39    40    41    42    43    44    45
##  [7,]    44    45    46    47    48    49    50
##  [8,]    49    50    51    52    53    54    55
##  [9,]    54    55    56    57    58    59    60
## [10,]    59    60    61    62    63    64    65
```

```
# Take middle of each window
Points.of.calculation<-Rolling.window.matrix[,10]
Points.of.calculation[1:10]
```

```
##  [1] 10 15 20 25 30 35 40 45 50 55
```

```
length(Points.of.calculation)
```

```
## [1] 1657
```

```r
# Insert means into the total length vector to plot the rolling mean with the original data
Means.forPlot<-rep(NA,length(AssignmentDataRegressionComparison[,1]))
Means.forPlot[Points.of.calculation]<-all.means[,1]
Means.forPlot[1:50]
```

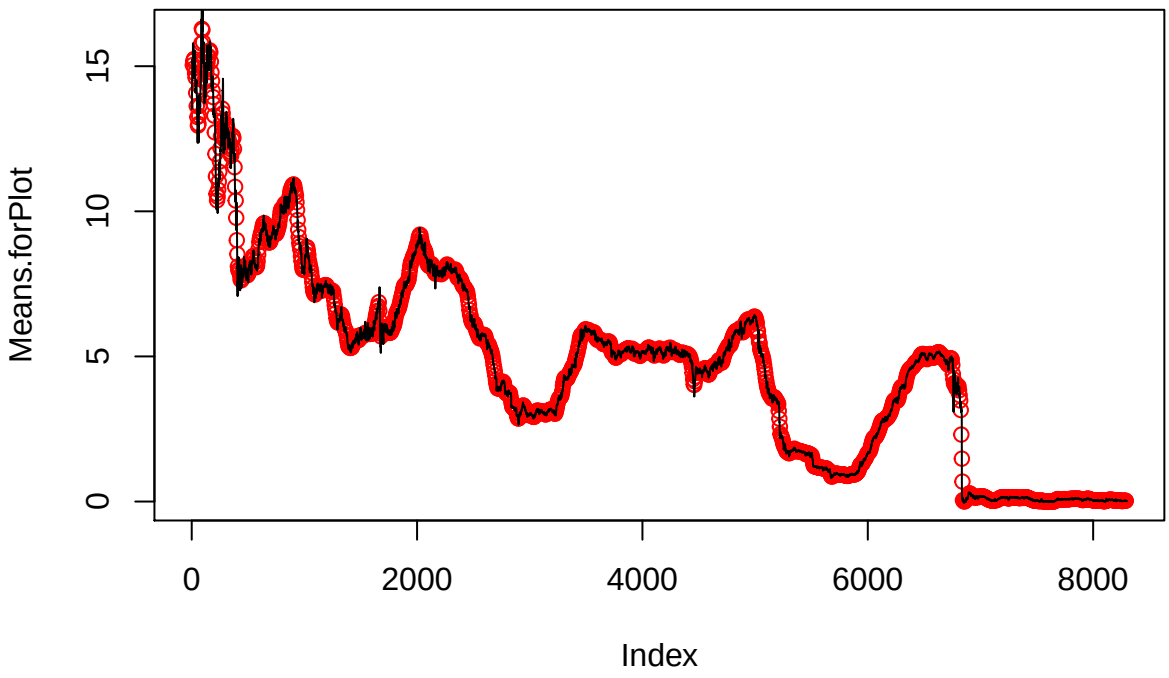
```
##  [1]      NA      NA      NA      NA      NA      NA      NA      NA
##  [9]      NA 15.0405      NA      NA      NA      NA 15.1865      NA
## [17]      NA      NA      NA 15.2480      NA      NA      NA      NA
## [25] 14.9345      NA      NA      NA      NA 14.7545      NA      NA
## [33]      NA      NA 14.6025      NA      NA      NA      NA 14.0820
## [41]      NA      NA      NA      NA 13.6255      NA      NA      NA
## [49]      NA 13.2490
```

```r
# Assemble the matrix to plot the rolling means
cbind(AssignmentDataRegressionComparison[,1],Means.forPlot)[1:50,]
```

```
##             Means.forPlot
##  [1,] 13.52            NA
##  [2,] 13.58            NA
##  [3,] 14.50            NA
##  [4,] 14.76            NA
##  [5,] 15.20            NA
##  [6,] 15.22            NA
##  [7,] 15.24            NA
##  [8,] 15.08            NA
##  [9,] 15.25            NA
## [10,] 15.15       15.0405
## [11,] 15.79            NA
## [12,] 15.32            NA
## [13,] 15.71            NA
## [14,] 15.72            NA
## [15,] 15.70       15.1865
## [16,] 15.35            NA
## [17,] 15.20            NA
## [18,] 15.06            NA
## [19,] 14.87            NA
## [20,] 14.59       15.2480
## [21,] 14.90            NA
## [22,] 14.85            NA
## [23,] 14.67            NA
## [24,] 14.74            NA
## [25,] 15.32       14.9345
## [26,] 15.52            NA
## [27,] 15.46            NA
## [28,] 15.54            NA
## [29,] 15.51            NA
## [30,] 15.14       14.7545
## [31,] 15.02            NA
## [32,] 14.48            NA
## [33,] 14.09            NA
## [34,] 14.23            NA
## [35,] 14.15       14.6025
## [36,] 14.20            NA
```

```
## [37,] 14.14            NA
## [38,] 14.22            NA
## [39,] 14.52            NA
## [40,] 14.39       14.0820
## [41,] 14.49            NA
## [42,] 14.51            NA
## [43,] 14.29            NA
## [44,] 14.16            NA
## [45,] 13.99       13.6255
## [46,] 13.92            NA
## [47,] 13.66            NA
## [48,] 13.21            NA
## [49,] 13.02            NA
## [50,] 12.95       13.2490
```

```
plot(Means.forPlot,col="red")
lines(AssignmentDataRegressionComparison[,1])
```




```
#Run rolling standard deviations daily for each variable
AssignmentDataRegressionComparison.Diff <- apply(AssignmentDataRegressionComparison, 2, diff, lag=1)
head(AssignmentDataRegressionComparison.Diff)
```

```
##           USGG3M USGG6M USGG2YR USGG3YR USGG5YR USGG10YR USGG30YR
## 1/6/1981    0.06   0.07    0.14    0.03   -0.08    -0.04     0.00
## 1/7/1981    0.92   0.74    0.50    0.47    0.40     0.27     0.22
```

```
## 1/8/1981    0.26   0.10    0.17    0.17    0.07    -0.03     0.02
## 1/9/1981    0.44   0.30    0.44    0.33    0.20     0.22     0.22
## 1/12/1981   0.02  -0.07   -0.36   -0.34   -0.17    -0.12    -0.05
## 1/13/1981   0.02  -0.13    0.13    0.03   -0.03     0.08     0.00
##               Output1
## 1/6/1981   0.07587222
## 1/7/1981   1.35591615
## 1/8/1981   0.30119608
## 1/9/1981   0.82353207
## 1/12/1981 -0.42985741
## 1/13/1981  0.03935812
```

```
rolling.sd<-rollapply(AssignmentDataRegressionComparison.Diff,width=Window.width,by=Window.shift,by.colu
head(rolling.sd)
```

```
##         USGG3M    USGG6M   USGG2YR   USGG3YR   USGG5YR  USGG10YR  USGG30YR
## [1,] 0.3433258 0.3262462 0.2748258 0.2030161 0.1713192 0.1299585 0.1202147
## [2,] 0.2933383 0.2907504 0.2261811 0.1499219 0.1450082 0.1146895 0.1192201
## [3,] 0.2613180 0.2437530 0.2006201 0.1632596 0.1654110 0.1459308 0.1351909
## [4,] 0.2551754 0.2469663 0.1989446 0.1692794 0.1717219 0.1551052 0.1422183
## [5,] 0.2480551 0.2481595 0.2102004 0.1786057 0.1744767 0.1643960 0.1516540
## [6,] 0.1963884 0.2363672 0.2095082 0.1809180 0.1822917 0.1664956 0.1537351
##        Output1
## [1,] 0.5639875
## [2,] 0.4707427
## [3,] 0.4681168
## [4,] 0.4786189
## [5,] 0.4888569
## [6,] 0.4788897
```

```
rolling.dates<-rollapply(AssignmentDataRegressionComparison[-1,],width=Window.width,by=Window.shift,
                         by.column=FALSE,FUN=function(z) rownames(z))
head(rolling.dates)
```

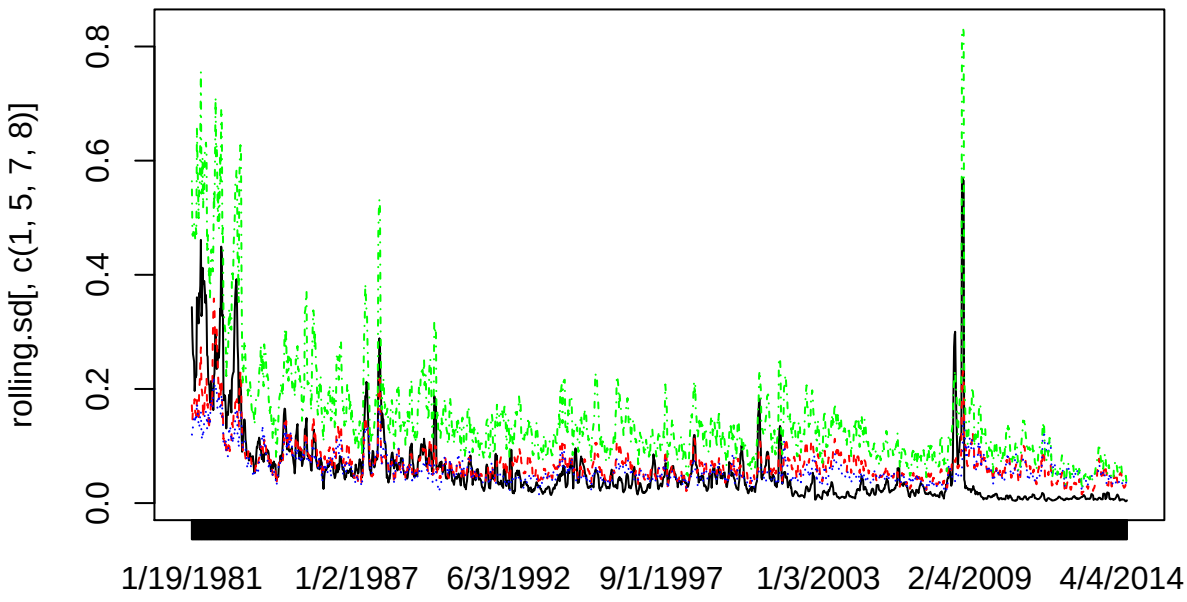
```
##      [,1]        [,2]        [,3]        [,4]        [,5]
## [1,] "1/6/1981"  "1/7/1981"  "1/8/1981"  "1/9/1981"  "1/12/1981"
## [2,] "1/13/1981" "1/14/1981" "1/15/1981" "1/16/1981" "1/19/1981"
## [3,] "1/20/1981" "1/21/1981" "1/22/1981" "1/23/1981" "1/26/1981"
## [4,] "1/27/1981" "1/28/1981" "1/29/1981" "1/30/1981" "2/2/1981"
## [5,] "2/3/1981"  "2/4/1981"  "2/5/1981"  "2/6/1981"  "2/9/1981"
## [6,] "2/10/1981" "2/11/1981" "2/13/1981" "2/17/1981" "2/18/1981"
##      [,6]        [,7]        [,8]        [,9]        [,10]
## [1,] "1/13/1981" "1/14/1981" "1/15/1981" "1/16/1981" "1/19/1981"
## [2,] "1/20/1981" "1/21/1981" "1/22/1981" "1/23/1981" "1/26/1981"
## [3,] "1/27/1981" "1/28/1981" "1/29/1981" "1/30/1981" "2/2/1981"
## [4,] "2/3/1981"  "2/4/1981"  "2/5/1981"  "2/6/1981"  "2/9/1981"
## [5,] "2/10/1981" "2/11/1981" "2/13/1981" "2/17/1981" "2/18/1981"
## [6,] "2/19/1981" "2/20/1981" "2/23/1981" "2/24/1981" "2/25/1981"
##      [,11]       [,12]       [,13]       [,14]       [,15]
## [1,] "1/20/1981" "1/21/1981" "1/22/1981" "1/23/1981" "1/26/1981"
## [2,] "1/27/1981" "1/28/1981" "1/29/1981" "1/30/1981" "2/2/1981"
## [3,] "2/3/1981"  "2/4/1981"  "2/5/1981"  "2/6/1981"  "2/9/1981"
## [4,] "2/10/1981" "2/11/1981" "2/13/1981" "2/17/1981" "2/18/1981"
```

```
## [5,] "2/19/1981" "2/20/1981" "2/23/1981" "2/24/1981" "2/25/1981"
## [6,] "2/26/1981" "2/27/1981" "3/2/1981"  "3/3/1981"  "3/4/1981"
##      [,16]       [,17]       [,18]       [,19]       [,20]
## [1,] "1/27/1981" "1/28/1981" "1/29/1981" "1/30/1981" "2/2/1981"
## [2,] "2/3/1981"  "2/4/1981"  "2/5/1981"  "2/6/1981"  "2/9/1981"
## [3,] "2/10/1981" "2/11/1981" "2/13/1981" "2/17/1981" "2/18/1981"
## [4,] "2/19/1981" "2/20/1981" "2/23/1981" "2/24/1981" "2/25/1981"
## [5,] "2/26/1981" "2/27/1981" "3/2/1981"  "3/3/1981"  "3/4/1981"
## [6,] "3/5/1981"  "3/6/1981"  "3/9/1981"  "3/10/1981" "3/11/1981"
```

```
rownames(rolling.sd)<-rolling.dates[,10]
head(rolling.sd)
```

```
##              USGG3M    USGG6M   USGG2YR   USGG3YR   USGG5YR  USGG10YR
## 1/19/1981 0.3433258 0.3262462 0.2748258 0.2030161 0.1713192 0.1299585
## 1/26/1981 0.2933383 0.2907504 0.2261811 0.1499219 0.1450082 0.1146895
## 2/2/1981  0.2613180 0.2437530 0.2006201 0.1632596 0.1654110 0.1459308
## 2/9/1981  0.2551754 0.2469663 0.1989446 0.1692794 0.1717219 0.1551052
## 2/18/1981 0.2480551 0.2481595 0.2102004 0.1786057 0.1744767 0.1643960
## 2/25/1981 0.1963884 0.2363672 0.2095082 0.1809180 0.1822917 0.1664956
##            USGG30YR   Output1
## 1/19/1981 0.1202147 0.5639875
## 1/26/1981 0.1192201 0.4707427
## 2/2/1981  0.1351909 0.4681168
## 2/9/1981  0.1422183 0.4786189
## 2/18/1981 0.1516540 0.4888569
## 2/25/1981 0.1537351 0.4788897
```

```
matplot(rolling.sd[,c(1,5,7,8)],xaxt="n",type="l",col=c("black","red","blue","green"))
axis(side=1,at=1:1656,rownames(rolling.sd))
```



Show periods of high volatility. How is volatility related to the level of rates? Higher volatility causes rates to be suppressed at a lower rate across the interest rate spectrum. We can see that rates usually decrease during the period, but not always. We don't see this in the high volatility window in the 3/81-7/81, but we do see decreasing in the periods starting 10/81, 7/82, 10/87, 11/07 and 11/08.

```
high.volatility.periods<-rownames(rolling.sd)[rolling.sd[,8]>.5]
high.volatility.periods
```

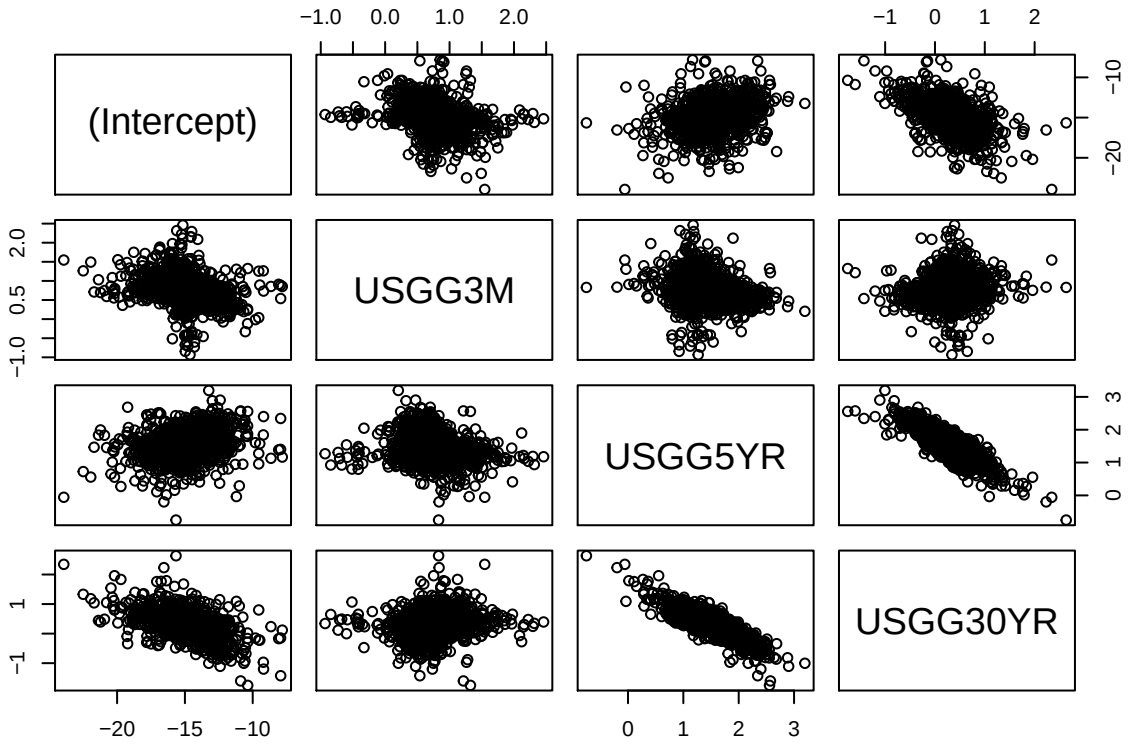
```
##  [1] "1/19/1981"  "3/25/1981"  "4/1/1981"   "4/8/1981"   "4/23/1981"
##  [6] "4/30/1981"  "5/7/1981"   "5/14/1981"  "5/21/1981"  "5/29/1981"
## [11] "6/5/1981"   "6/12/1981"  "6/19/1981"  "6/26/1981"  "7/6/1981"
## [16] "7/13/1981"  "7/20/1981"  "7/27/1981"  "10/28/1981" "11/5/1981"
## [21] "11/13/1981" "11/20/1981" "11/30/1981" "12/7/1981"  "12/14/1981"
## [26] "12/29/1981" "1/14/1982"  "1/21/1982"  "1/28/1982"  "2/4/1982"
## [31] "2/11/1982"  "7/29/1982"  "8/5/1982"   "8/12/1982"  "8/19/1982"
## [36] "8/26/1982"  "9/24/1982"  "10/1/1982"  "10/8/1982"  "10/18/1982"
## [41] "10/13/1987" "10/20/1987" "10/27/1987" "11/19/2007" "11/26/2007"
## [46] "11/12/2008" "11/19/2008"
```

```
# Rolling lm coefficients
Coefficients<-rollapply(AssignmentDataRegressionComparison,width=Window.width,by=Window.shift,by.column=
         FUN=function(z) coef(lm(Output1~USGG3M+USGG5YR+USGG30YR,data=as.data.frame(z))))
rolling.dates<-rollapply(AssignmentDataRegressionComparison[,1:8],width=Window.width,by=Window.shift,by
                         FUN=function(z) rownames(z))

rownames(Coefficients)<-rolling.dates[,10]
Coefficients[1:10,]
```

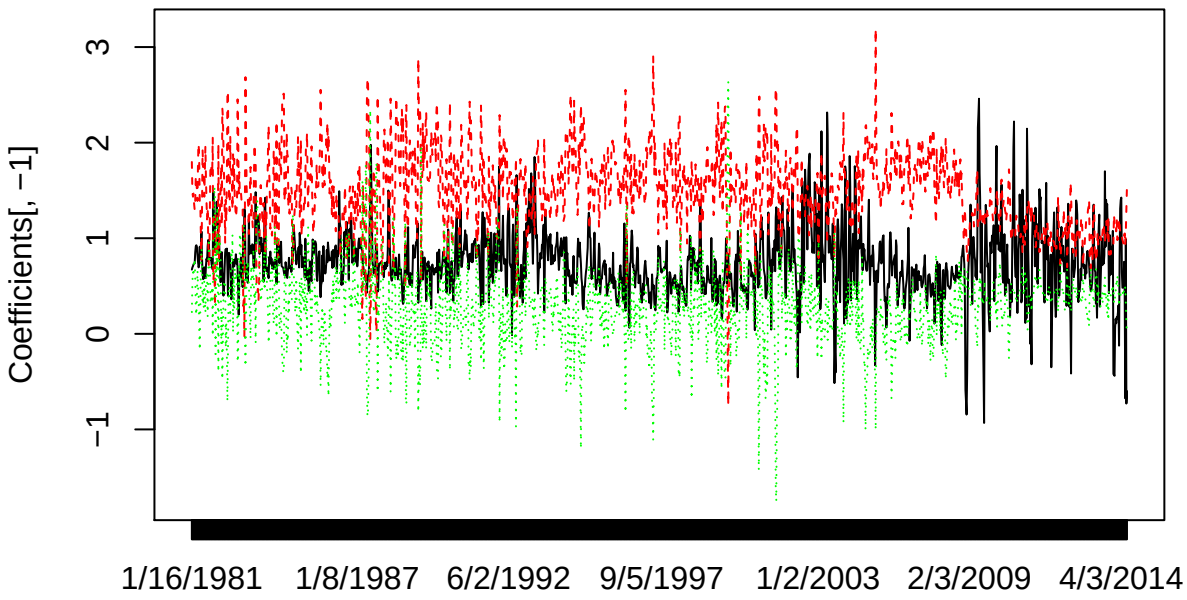
```
##            (Intercept)    USGG3M  USGG5YR  USGG30YR
## 1/16/1981   -15.70877 0.6723609 1.797680 0.2276011
## 1/23/1981   -15.96684 0.6948992 1.480514 0.5529139
## 1/30/1981   -16.77273 0.7078197 1.434388 0.6507280
## 2/6/1981    -16.90734 0.7279033 1.470083 0.6003377
## 2/17/1981   -17.46896 0.7343406 1.361289 0.7499705
## 2/24/1981   -17.04722 0.7357663 1.295641 0.7844908
## 3/3/1981    -17.67901 0.8544681 1.396653 0.5945022
## 3/10/1981   -17.72402 0.9162385 1.654274 0.2571200
## 3/17/1981   -17.00260 0.9265767 1.647852 0.1951273
## 3/24/1981   -16.84302 0.9102780 1.477727 0.3788401
```

```
# Pairs plot of Coefficients
pairs(Coefficients)
```



Interpret the pairs plot. The pairs plot shows a linear relationship between the US 5YR and US 30YR. There appears to be no linear relationship between the other variables.

```
# Plot of coefficients
matplot(Coefficients[,-1],xaxt="n",type="l",col=c("black","red","green"))
axis(side=1,at=1:1657,rownames(Coefficients))
```




```
high.slopespread.periods<-rownames(Coefficients)[Coefficients[,3]-Coefficients[,4]>3]
jump.slopes<-rownames(Coefficients)[Coefficients[,3]>3]
##black red gren, coefficients. intersted how coeff is changing in time depending on environemnt

high.slopespread.periods
```

```
##  [1] "4/26/1982"  "12/15/1982" "9/16/1985"  "5/12/1987"  "5/19/1987"
##  [6] "5/27/1987"  "9/25/1987"  "3/15/1988"  "9/27/1988"  "10/5/1988"
## [11] "3/10/1989"  "2/5/1992"   "8/3/1994"   "12/8/1994"  "6/14/1996"
## [16] "5/9/1997"   "5/16/1997"  "5/23/1997"  "5/30/1997"  "12/26/2000"
## [21] "1/2/2001"   "7/25/2001"  "8/1/2001"   "11/13/2003" "8/12/2004"
## [26] "12/16/2004"
```

```
jump.slopes
```

```
## [1] "12/16/2004"
```

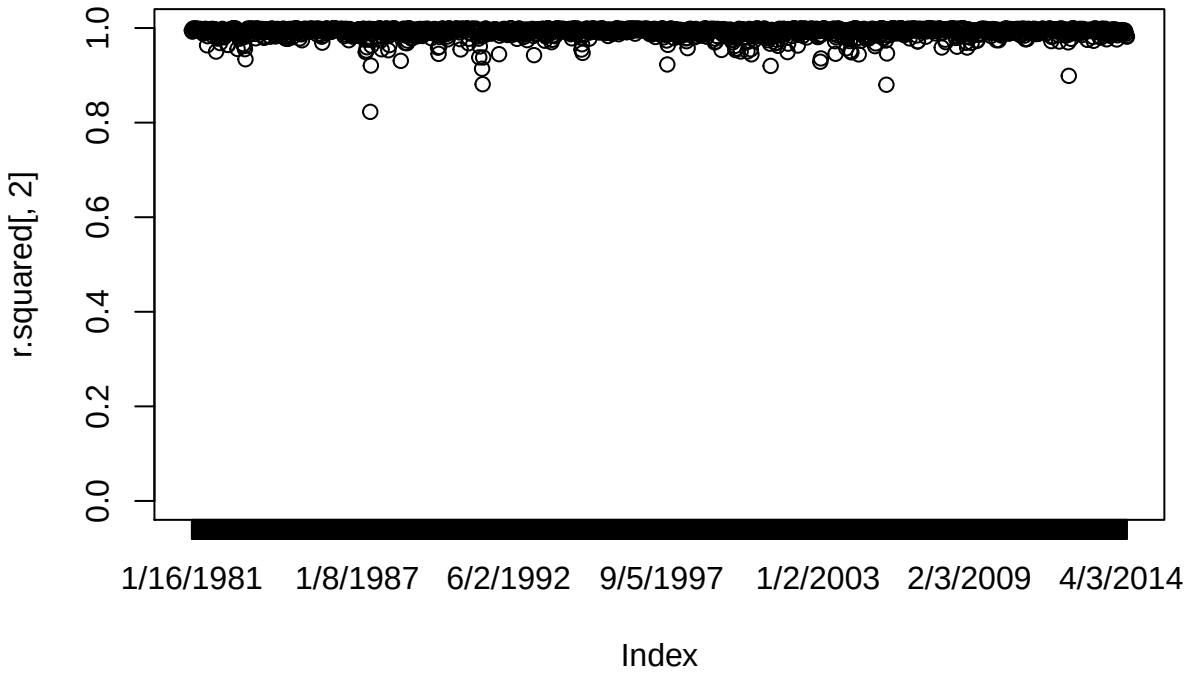
Is the picture of coefficients consistent with the picture of pairs? If yes, explain why. Yes it is consistent, the chart shows that the 3 Mo and 30 year coefficients are more in line for the majority of the time frame. While the 5 year coefficient is higher and more significant than the two ends of the yield curve. In the final period, all 3 seem to converge and overlap. As the 5YR and 30YR converge, that movement is reflected in the negative correlation in the pairs plot of the coefficients of the 5YR and 30YR.

How often the R-squared is not considered high? For the most part the R-squared is high, above 0.90, there are only a handful of instances where it is closer to 0.8

```
# R-squared
r.squared<-rollapply(AssignmentDataRegressionComparison,width=Window.width,by=Window.shift,by.column=FA
         FUN=function(z) summary(lm(Output1~USGG3M+USGG5YR+USGG30YR,data=as.data.frame(z)))$r.squared)
r.squared<-cbind(rolling.dates[,10],r.squared)
r.squared[1:10,]
```

```
##                    r.squared
##  [1,] "1/16/1981" "0.995046300986446"
##  [2,] "1/23/1981" "0.992485868136646"
##  [3,] "1/30/1981" "0.998641209587999"
##  [4,] "2/6/1981"  "0.998849080081881"
##  [5,] "2/17/1981" "0.997958757207598"
##  [6,] "2/24/1981" "0.996489757136839"
##  [7,] "3/3/1981"  "0.99779753570421"
##  [8,] "3/10/1981" "0.998963395226792"
##  [9,] "3/17/1981" "0.998729445388789"
## [10,] "3/24/1981" "0.997073000898673"
```

```
plot(r.squared[,2],xaxt="n",ylim=c(0,1))
axis(side=1,at=1:1657,rownames(Coefficients))
```




```
(low.r.squared.periods<-r.squared[r.squared[,2]<.9,1])
```

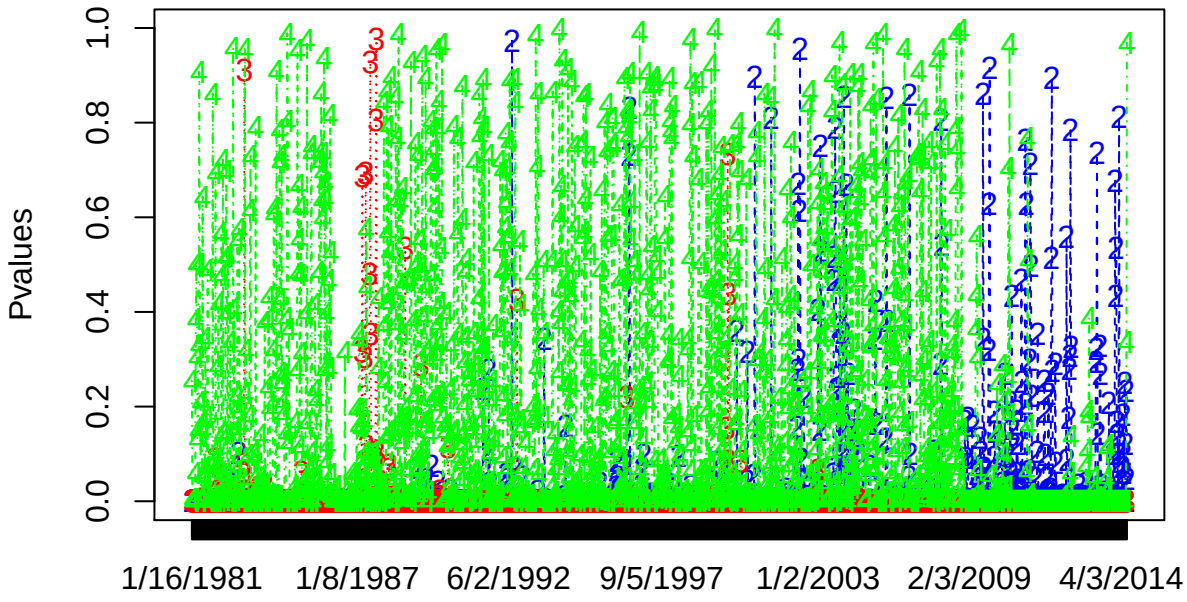
```
## [1] "6/24/1987" "6/27/1991" "4/28/2005" "6/20/2012"
```

What could cause decrease of R2R2? The occasional dip in R-squared could be caused by extreme market shocks, such as Fed policy changes or financial events(Black Tuesday, Lehman crisis, Flash crash). Most of the time these three variables capture all the variation in the model. On occasion, they are not enough and the other variables that we removed are responsible for the movements

```
# P-values
Pvalues<-rollapply(AssignmentDataRegressionComparison,width=Window.width,by=Window.shift,by.column=FALSE
                        FUN=function(z) summary(lm(Output1~USGG3M+USGG5YR+USGG30YR,data=as.data.frame(z)
#in recent times 2nd coeff less signfi
#3rd is insig, then becomes sign again, changing model/importance
rownames(Pvalues)<-rolling.dates[,10]
Pvalues[1:10,]
```

```
##            (Intercept)       USGG3M      USGG5YR     USGG30YR
## 1/16/1981 1.193499e-10 3.764585e-10 2.391260e-07 2.538852e-01
## 1/23/1981 3.751077e-12 1.008053e-11 2.447369e-07 5.300949e-03
## 1/30/1981 3.106359e-18 1.406387e-14 4.040035e-09 3.626961e-05
## 2/6/1981  2.591522e-19 3.360104e-19 3.828054e-11 2.221691e-05
## 2/17/1981 1.897239e-16 6.578118e-17 1.461743e-09 1.331767e-04
## 2/24/1981 2.341158e-13 1.000212e-13 9.008221e-07 4.733543e-03
## 3/3/1981  5.435581e-14 1.535503e-11 3.357199e-06 6.010473e-02
## 3/10/1981 6.227624e-16 1.178498e-16 1.679479e-05 3.851840e-01
## 3/17/1981 9.592582e-17 7.065226e-20 1.459692e-05 5.025726e-01
## 3/24/1981 8.248747e-16 6.689840e-16 6.413371e-04 3.052705e-01
```

```
matplot(Pvalues,xaxt="n",col=c("black","blue","red","green"),type="o")
axis(side=1,at=1:1657,rownames(Coefficients))
```




```
rownames(Pvalues)[Pvalues[,2]>.5]
```

```
##  [1] "7/15/1992"  "7/26/1996"  "8/2/1996"   "11/7/2000"  "5/30/2001"
##  [6] "5/2/2002"   "5/16/2002"  "5/23/2002"  "1/30/2003"  "2/6/2003"
## [11] "7/24/2003"  "7/31/2003"  "8/7/2003"   "11/20/2003" "12/18/2003"
## [16] "4/28/2005"  "2/10/2006"  "3/9/2007"   "3/16/2007"  "7/21/2009"
## [21] "10/6/2009"  "10/13/2009" "12/28/2010" "1/11/2011"  "3/1/2011"
## [26] "11/16/2011" "11/23/2011" "5/23/2012"  "7/11/2012"  "6/6/2013"
## [31] "1/16/2014"  "1/30/2014"  "3/6/2014"
```

```
rownames(Pvalues)[Pvalues[,3]>.5]
```

```
## [1] "12/1/1982" "3/16/1987" "4/28/1987" "6/24/1987" "9/3/1987"  "9/11/1987"
## [7] "9/20/1988" "12/3/1999"
```

```
rownames(Pvalues)[Pvalues[,4]>.5]
```

```
##   [1] "3/17/1981"  "4/22/1981"  "4/29/1981"  "6/4/1981"   "10/13/1981"
##   [6] "11/19/1981" "2/3/1982"   "2/26/1982"  "4/2/1982"   "4/12/1982"
##  [11] "5/3/1982"   "7/7/1982"   "9/1/1982"   "9/23/1982"  "12/8/1982"
##  [16] "2/18/1983"  "3/1/1983"   "5/4/1983"   "12/30/1983" "1/9/1984"
##  [21] "2/6/1984"   "3/14/1984"  "3/21/1984"  "4/11/1984"  "6/22/1984"
##  [26] "6/29/1984"  "10/31/1984" "11/16/1984" "11/26/1984" "12/17/1984"
##  [31] "3/25/1985"  "5/14/1985"  "5/21/1985"  "8/30/1985"  "9/9/1985"
```

```
##  [36] "9/23/1985"  "10/1/1985"  "10/8/1985"  "10/16/1985" "10/23/1985"
##  [41] "10/30/1985" "11/14/1985" "11/21/1985" "1/22/1986"  "1/29/1986"
##  [46] "5/5/1987"   "12/16/1987" "1/25/1988"  "2/1/1988"   "2/16/1988"
##  [51] "3/1/1988"   "3/22/1988"  "5/18/1988"  "6/3/1988"   "6/10/1988"
##  [56] "6/24/1988"  "7/25/1988"  "8/15/1988"  "12/5/1988"  "2/2/1989"
##  [61] "3/3/1989"   "4/10/1989"  "5/1/1989"   "6/13/1989"  "8/16/1989"
##  [66] "9/14/1989"  "9/21/1989"  "10/3/1989"  "10/11/1989" "10/18/1989"
##  [71] "11/1/1989"  "11/30/1989" "12/7/1989"  "1/8/1990"   "1/16/1990"
##  [76] "6/15/1990"  "7/30/1990"  "8/6/1990"   "10/2/1990"  "10/10/1990"
##  [81] "11/23/1990" "3/1/1991"   "5/13/1991"  "5/20/1991"  "6/13/1991"
##  [86] "7/5/1991"   "7/19/1991"  "9/16/1991"  "2/12/1992"  "2/20/1992"
##  [91] "3/12/1992"  "4/16/1992"  "4/24/1992"  "5/1/1992"   "5/8/1992"
##  [96] "6/2/1992"   "6/9/1992"   "6/16/1992"  "8/19/1992"  "8/26/1992"
## [101] "10/23/1992" "5/20/1993"  "6/11/1993"  "6/18/1993"  "8/30/1993"
## [106] "12/15/1993" "12/22/1993" "3/16/1994"  "3/30/1994"  "4/6/1994"
## [111] "4/20/1994"  "4/27/1994"  "6/29/1994"  "8/17/1994"  "9/21/1994"
## [116] "9/28/1994"  "12/22/1994" "12/29/1994" "1/5/1995"   "1/12/1995"
## [121] "1/26/1995"  "2/2/1995"   "2/9/1995"   "4/6/1995"   "4/13/1995"
## [126] "8/25/1995"  "9/29/1995"  "10/27/1995" "11/3/1995"  "11/10/1995"
## [131] "12/29/1995" "1/5/1996"   "1/12/1996"  "1/19/1996"  "3/29/1996"
## [136] "5/31/1996"  "6/21/1996"  "7/12/1996"  "7/19/1996"  "7/26/1996"
## [141] "8/2/1996"   "8/9/1996"   "8/16/1996"  "8/23/1996"  "9/27/1996"
## [146] "10/4/1996"  "12/6/1996"  "2/28/1997"  "3/7/1997"   "4/18/1997"
## [151] "4/25/1997"  "5/2/1997"   "6/13/1997"  "6/20/1997"  "6/27/1997"
## [156] "7/4/1997"   "10/10/1997" "10/17/1997" "12/12/1997" "12/19/1997"
## [161] "12/26/1997" "1/9/1998"   "1/16/1998"  "8/14/1998"  "8/21/1998"
## [166] "8/28/1998"  "9/18/1998"  "9/25/1998"  "12/4/1998"  "12/11/1998"
## [171] "1/8/1999"   "3/12/1999"  "4/2/1999"   "5/14/1999"  "6/25/1999"
## [176] "7/9/1999"   "7/30/1999"  "8/20/1999"  "9/10/1999"  "9/24/1999"
## [181] "10/15/1999" "12/31/1999" "3/3/2000"   "3/31/2000"  "4/7/2000"
## [186] "4/14/2000"  "4/21/2000"  "7/25/2000"  "11/21/2000" "11/28/2000"
## [191] "3/7/2001"   "5/30/2001"  "7/11/2001"  "10/11/2001" "12/13/2001"
## [196] "1/24/2002"  "1/31/2002"  "8/29/2002"  "9/26/2002"  "12/26/2002"
## [201] "1/16/2003"  "1/23/2003"  "3/6/2003"   "3/13/2003"  "3/20/2003"
## [206] "3/27/2003"  "5/1/2003"   "6/19/2003"  "6/26/2003"  "7/3/2003"
## [211] "9/18/2003"  "9/25/2003"  "10/16/2003" "10/23/2003" "10/30/2003"
## [216] "11/20/2003" "1/1/2004"   "2/5/2004"   "2/26/2004"  "3/4/2004"
## [221] "4/15/2004"  "4/22/2004"  "5/13/2004"  "5/27/2004"  "6/3/2004"
## [226] "6/17/2004"  "6/24/2004"  "7/8/2004"   "10/14/2004" "10/21/2004"
## [231] "10/28/2004" "11/18/2004" "12/23/2004" "3/17/2005"  "3/24/2005"
## [236] "3/31/2005"  "4/7/2005"   "7/21/2005"  "8/11/2005"  "9/22/2005"
## [241] "10/6/2005"  "11/3/2005"  "12/8/2005"  "12/15/2005" "1/20/2006"
## [246] "5/12/2006"  "5/19/2006"  "5/26/2006"  "6/2/2006"   "6/9/2006"
## [251] "6/16/2006"  "7/7/2006"   "7/21/2006"  "10/6/2006"  "11/3/2006"
## [256] "1/12/2007"  "2/23/2007"  "3/16/2007"  "4/27/2007"  "5/25/2007"
## [261] "9/7/2007"   "9/14/2007"  "9/21/2007"  "9/28/2007"  "11/16/2007"
## [266] "5/5/2009"   "6/1/2010"   "6/15/2010"  "1/25/2011"  "2/1/2011"
## [271] "6/12/2014"
```

Interpret the plot. For the majority of the time period, the 30YR was a poor predictor and had a higher P-value. Since roughly 2008, the 30YR has been a better predictor and the 3MO has had less value because of its higher P-value.

Step 7.

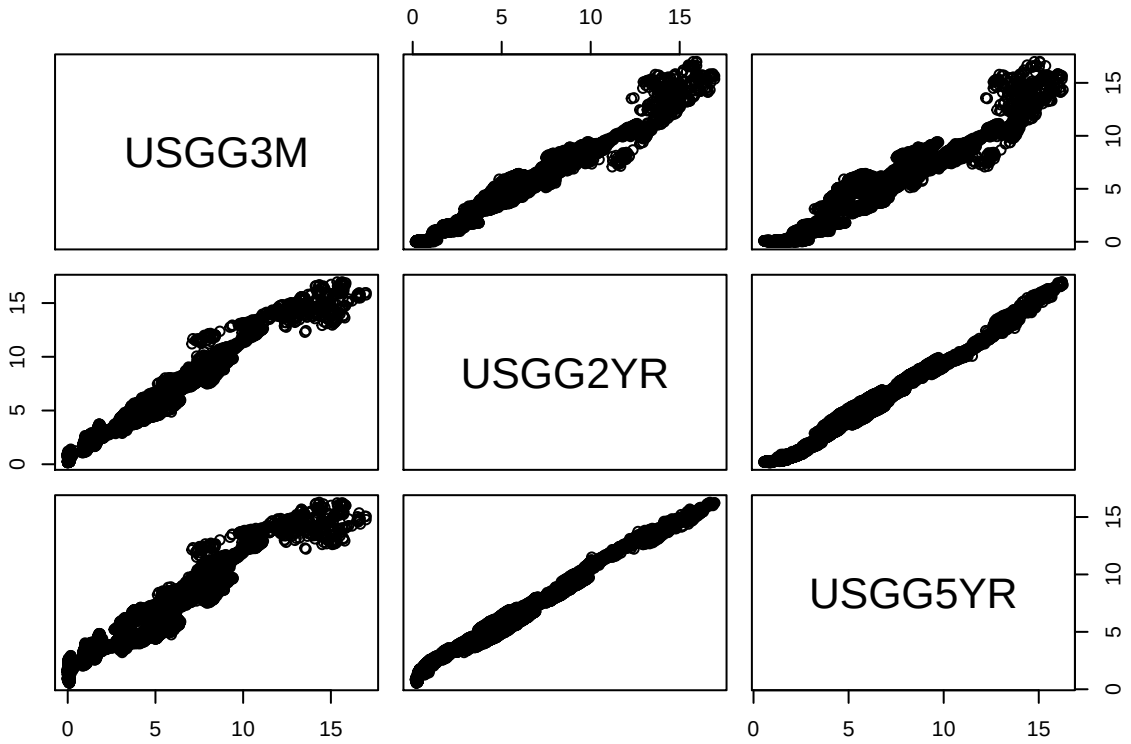
```
#Perform PCA with the inputs (columns 1-7)
AssignmentData.Output<-AssignmentData$Output1
AssignmentData<-data.matrix(AssignmentData[,1:7],rownames.force="automatic")
dim(AssignmentData)
```

```
## [1] 8300    7
```

```
head(AssignmentData)
```

```
##           USGG3M USGG6M USGG2YR USGG3YR USGG5YR USGG10YR USGG30YR
## 1/5/1981   13.52  13.09  12.289   12.28  12.294   12.152   11.672
## 1/6/1981   13.58  13.16  12.429   12.31  12.214   12.112   11.672
## 1/7/1981   14.50  13.90  12.929   12.78  12.614   12.382   11.892
## 1/8/1981   14.76  14.00  13.099   12.95  12.684   12.352   11.912
## 1/9/1981   15.20  14.30  13.539   13.28  12.884   12.572   12.132
## 1/12/1981  15.22  14.23  13.179   12.94  12.714   12.452   12.082
```

```
# Select 3 variables. Explore dimensionality and correlation
AssignmentData.3M_2Y_5Y<-AssignmentData[,c(1,3,5)]
pairs(AssignmentData.3M_2Y_5Y)
```



```r
#Observe the 3D plot of the set
#library("rgl")`;rgl.points(AssignmentData.3M_2Y_5Y)

#create means for each column
Means <- matrix(data=1, nrow=8300) %*% cbind(mean(AssignmentData[,1]),mean(AssignmentData[,2]),mean(Ass
head(Means)
```

```
##          [,1]    [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
## [1,] 4.675134 4.84437 5.438888 5.644458 6.009421 6.481316 6.869355
## [2,] 4.675134 4.84437 5.438888 5.644458 6.009421 6.481316 6.869355
## [3,] 4.675134 4.84437 5.438888 5.644458 6.009421 6.481316 6.869355
## [4,] 4.675134 4.84437 5.438888 5.644458 6.009421 6.481316 6.869355
## [5,] 4.675134 4.84437 5.438888 5.644458 6.009421 6.481316 6.869355
## [6,] 4.675134 4.84437 5.438888 5.644458 6.009421 6.481316 6.869355
```

```r
#creates a difference matrix
DifferencesMatrix <- AssignmentData - Means
head(DifferencesMatrix)
```

```
##              USGG3M  USGG6M  USGG2YR  USGG3YR  USGG5YR USGG10YR USGG30YR
## 1/5/1981   8.844866 8.24563 6.850112 6.635542 6.284579 5.670684 4.802645
## 1/6/1981   8.904866 8.31563 6.990112 6.665542 6.204579 5.630684 4.802645
## 1/7/1981   9.824866 9.05563 7.490112 7.135542 6.604579 5.900684 5.022645
## 1/8/1981  10.084866 9.15563 7.660112 7.305542 6.674579 5.870684 5.042645
## 1/9/1981  10.524866 9.45563 8.100112 7.635542 6.874579 6.090684 5.262645
## 1/12/1981 10.544866 9.38563 7.740112 7.295542 6.704579 5.970684 5.212645
```

```r
#creates the covariance matrix
Manual.Covariance.Matrix <- 1/(8300-1)* t(DifferencesMatrix) %*% DifferencesMatrix
Manual.Covariance.Matrix
```

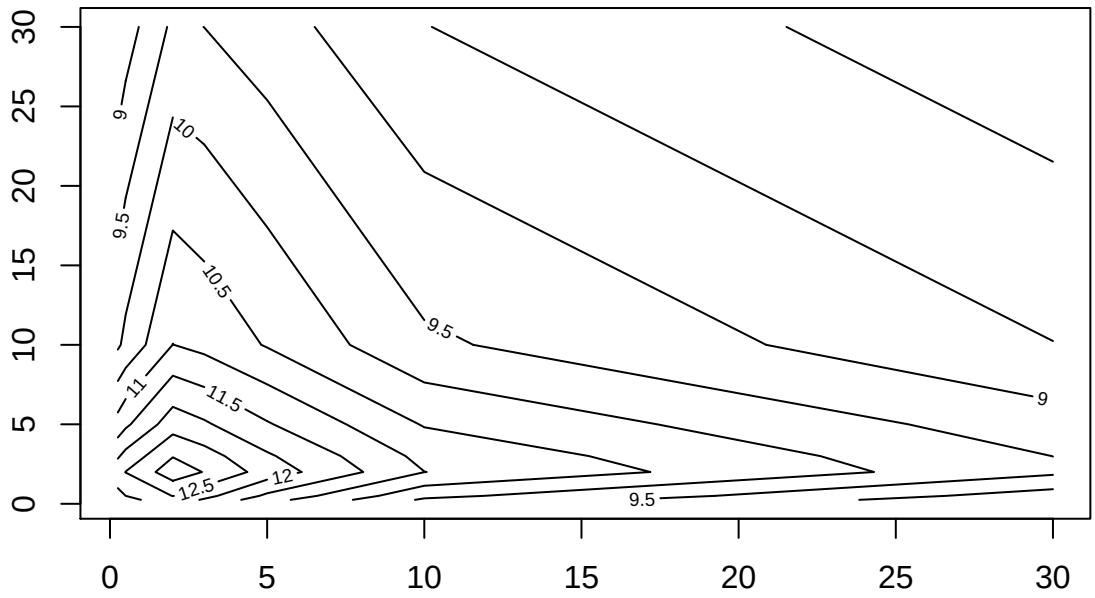
```
##             USGG3M    USGG6M   USGG2YR   USGG3YR   USGG5YR  USGG10YR
## USGG3M   11.760393 11.855287 12.303031 11.942035 11.188856  9.924865
## USGG6M   11.855287 12.000510 12.512434 12.158422 11.406959 10.128890
## USGG2YR  12.303031 12.512434 13.284203 12.977542 12.279514 11.005377
## USGG3YR  11.942035 12.158422 12.977542 12.708647 12.068078 10.856033
## USGG5YR  11.188856 11.406959 12.279514 12.068078 11.543082 10.463386
## USGG10YR  9.924865 10.128890 11.005377 10.856033 10.463386  9.583483
## USGG30YR  8.587987  8.768702  9.600181  9.497246  9.212159  8.510632
##          USGG30YR
## USGG3M   8.587987
## USGG6M   8.768702
## USGG2YR  9.600181
## USGG3YR  9.497246
## USGG5YR  9.212159
## USGG10YR 8.510632
## USGG30YR 7.624304
```

```r
Covariance.Matrix<-cov(AssignmentData)
Covariance.Matrix
```

```
##             USGG3M    USGG6M   USGG2YR   USGG3YR   USGG5YR  USGG10YR
```

```
## USGG3M   11.760393 11.855287 12.303031 11.942035 11.188856  9.924865
## USGG6M   11.855287 12.000510 12.512434 12.158422 11.406959 10.128890
## USGG2YR  12.303031 12.512434 13.284203 12.977542 12.279514 11.005377
## USGG3YR  11.942035 12.158422 12.977542 12.708647 12.068078 10.856033
## USGG5YR  11.188856 11.406959 12.279514 12.068078 11.543082 10.463386
## USGG10YR  9.924865 10.128890 11.005377 10.856033 10.463386  9.583483
## USGG30YR  8.587987  8.768702  9.600181  9.497246  9.212159  8.510632
##          USGG30YR
## USGG3M   8.587987
## USGG6M   8.768702
## USGG2YR  9.600181
## USGG3YR  9.497246
## USGG5YR  9.212159
## USGG10YR 8.510632
## USGG30YR 7.624304
```

```
#Plot the covariance matrix
Maturities<-c(.25,.5,2,3,5,10,30)
contour(Maturities,Maturities,Covariance.Matrix)
```



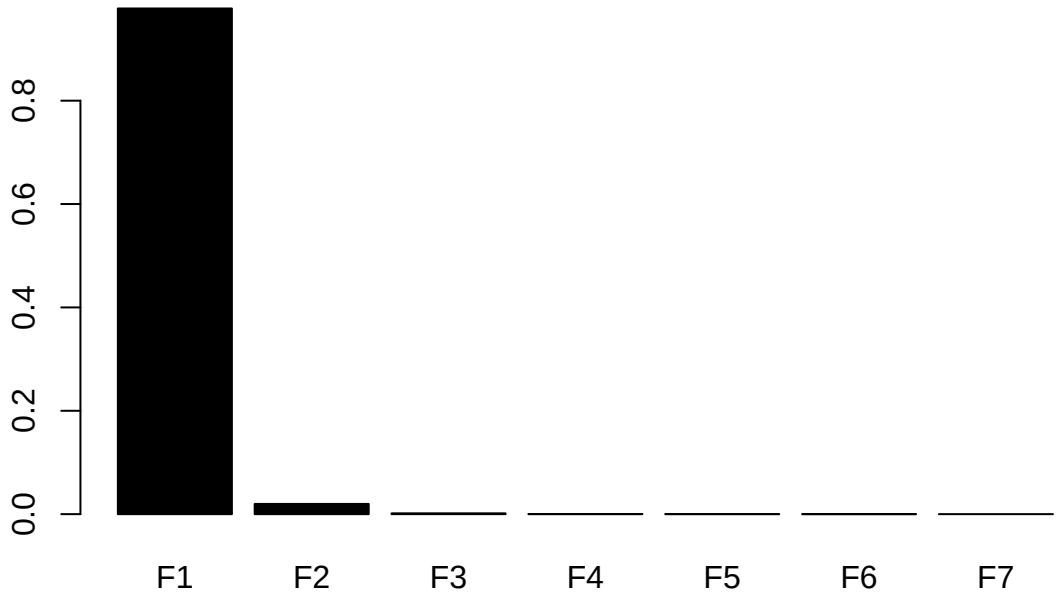
Perform the PCA by manually calculating factors, loadings and analyzing the importance of factors.

Find eigenvalues and eigenvectors. Calculate vector of means (zero loading), first 3 loadings and 3 factors.

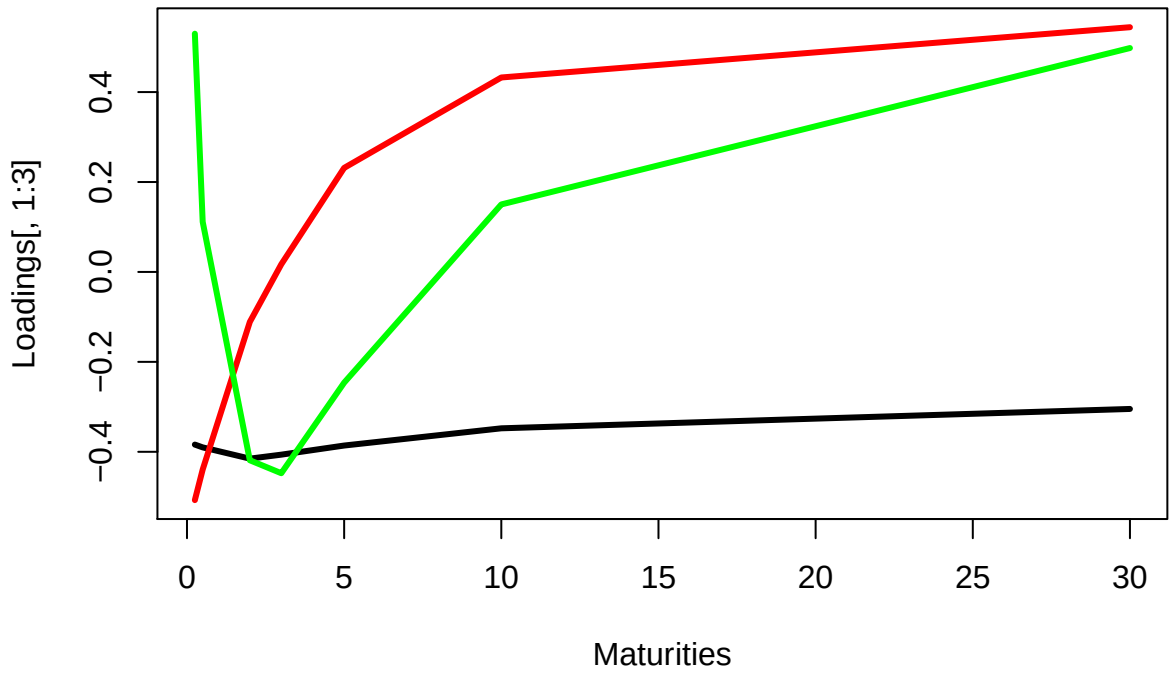
```
#Complete eigen decomposition on the covariance matrix
Eigen.Decomposition<-eigen(Covariance.Matrix)
Eigen.Decomposition
```

```
## $values
## [1] 76.804437933  1.551521258  0.122380498  0.014155405  0.008321381
## [6]  0.002248917  0.001555835
##
## $vectors
##            [,1]        [,2]       [,3]        [,4]       [,5]        [,6]
## [1,] -0.3839609 -0.50744508  0.5298222  0.40373501  0.3860878 -0.03976285
## [2,] -0.3901870 -0.43946144  0.1114737 -0.40526448 -0.6787624  0.09475452
## [3,] -0.4151851 -0.11112721 -0.4187873 -0.40896949  0.3787209 -0.29848638
## [4,] -0.4063541  0.01696988 -0.4476561  0.06433748  0.2362448  0.19760026
## [5,] -0.3860610  0.23140317 -0.2462364  0.53357656 -0.2868460  0.42125768
## [6,] -0.3477544  0.43245979  0.1500903  0.19856539 -0.2562426 -0.73561857
## [7,] -0.3047124  0.54421228  0.4979195 -0.42098839  0.2074508  0.37776687
##              [,7]
## [1,]  0.026742547
## [2,] -0.090913541
## [3,]  0.490009873
## [4,] -0.731570606
## [5,]  0.438559615
## [6,] -0.152627535
## [7,]  0.009199827
```

```r
#Plot the relative variances
barplot(Eigen.Decomposition$values/sum(Eigen.Decomposition$values),width=2,col = "black",
        names.arg=c("F1","F2","F3","F4","F5","F6","F7"))
```

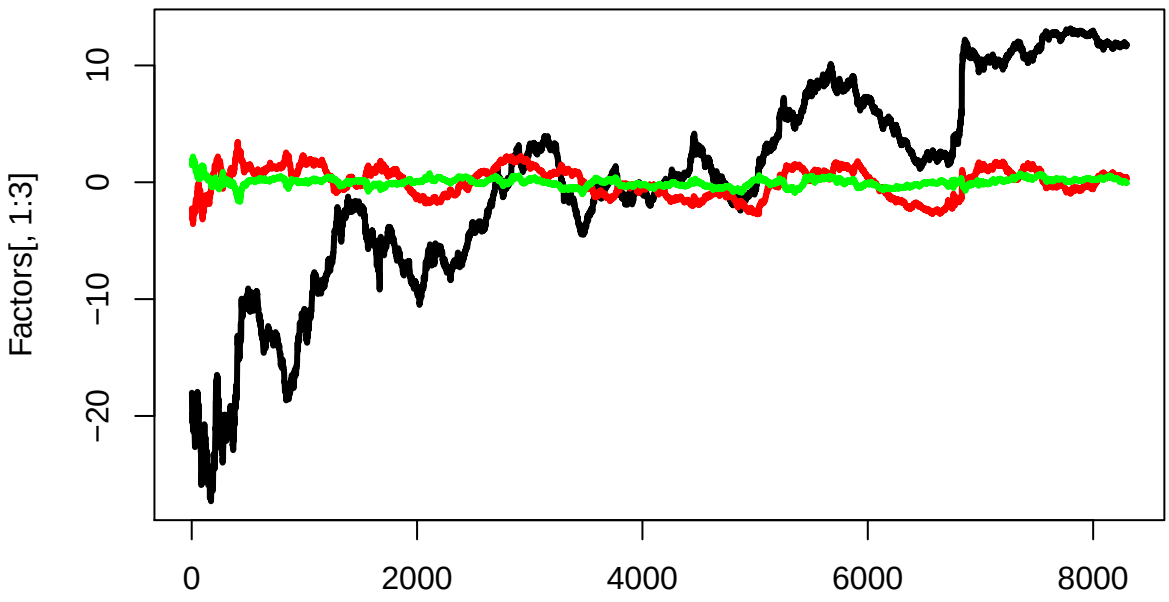


```
#Calculate Factors, plot the loadings
Loadings<-eigen(Covariance.Matrix)$vectors
Factors<-DifferencesMatrix%*%Loadings
matplot(Maturities,Loadings[,1:3],type="l",lty=1,col=c("black","red","green"),lwd=3)
```

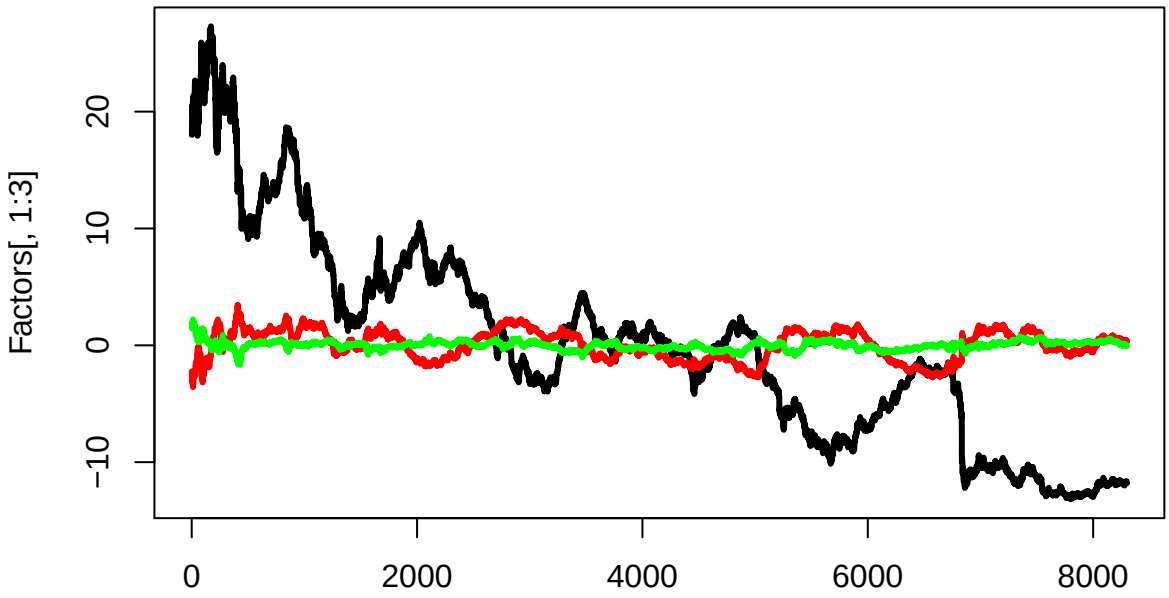


Interpret the factors by looking at the shapes of the loadings Factor 1 has a negative loading for all maturities. Factor 2 starts off negative, but becomes positive around the 3 yr point. Factor 3 starts off positive, turns negative around year 1 or 2, goes negative, and then begins to climb and turns postive again in year 8 or 9.

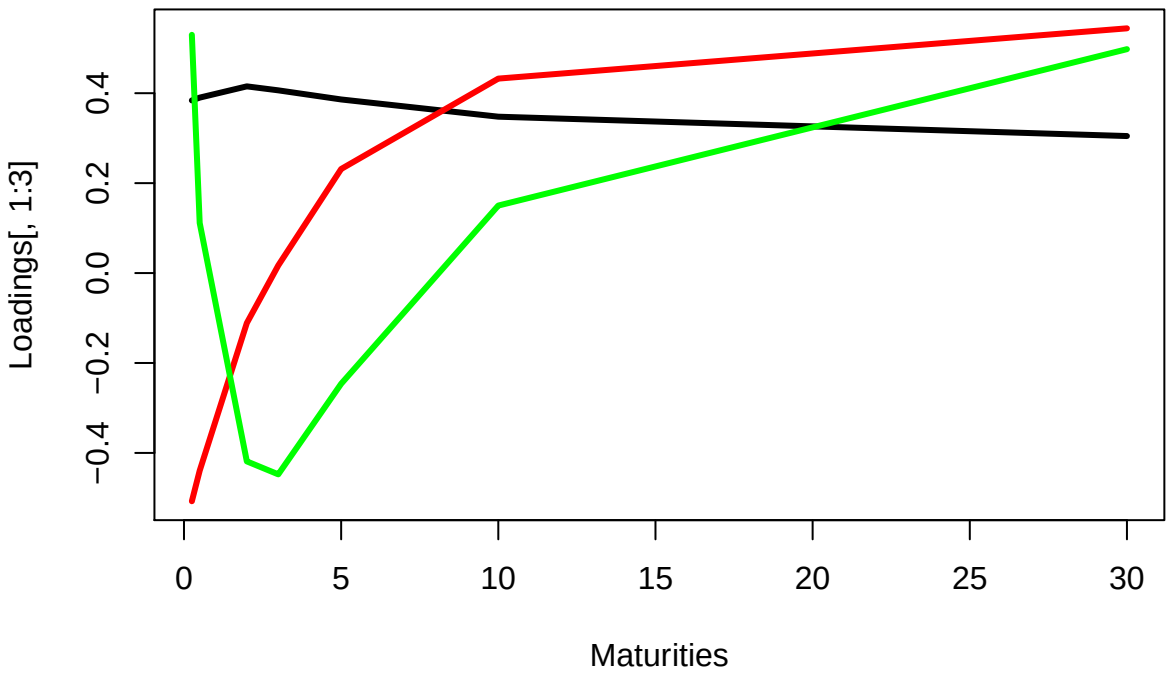
```
#Calculate and plot 3 selected factors
matplot(Factors[,1:3],type="l",col=c("black","red","green"),lty=1,lwd=3)
```



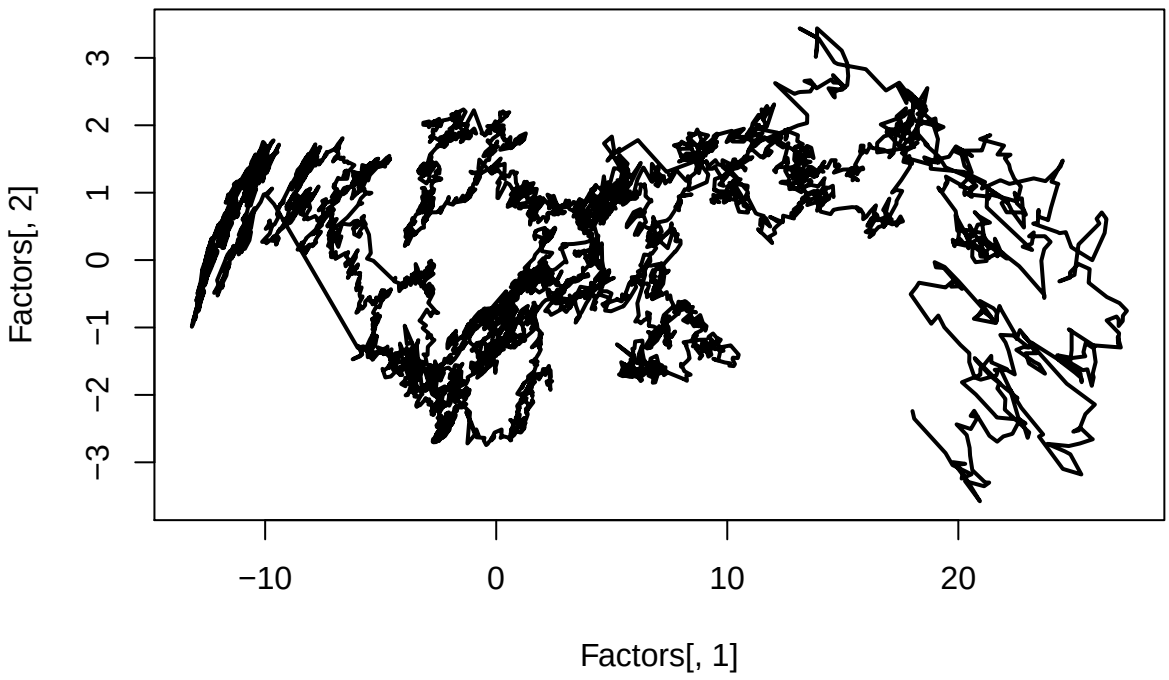

```
#Change the signs of the first factor and the corresponding loading
Loadings[,1]<--Loadings[,1]
Factors[,1]<--Factors[,1]
matplot(Factors[,1:3],type="l",col=c("black","red","green"),lty=1,lwd=3)
```




```
matplot(Maturities,Loadings[,1:3],type="l",lty=1,col=c("black","red","green"),lwd=3)
```

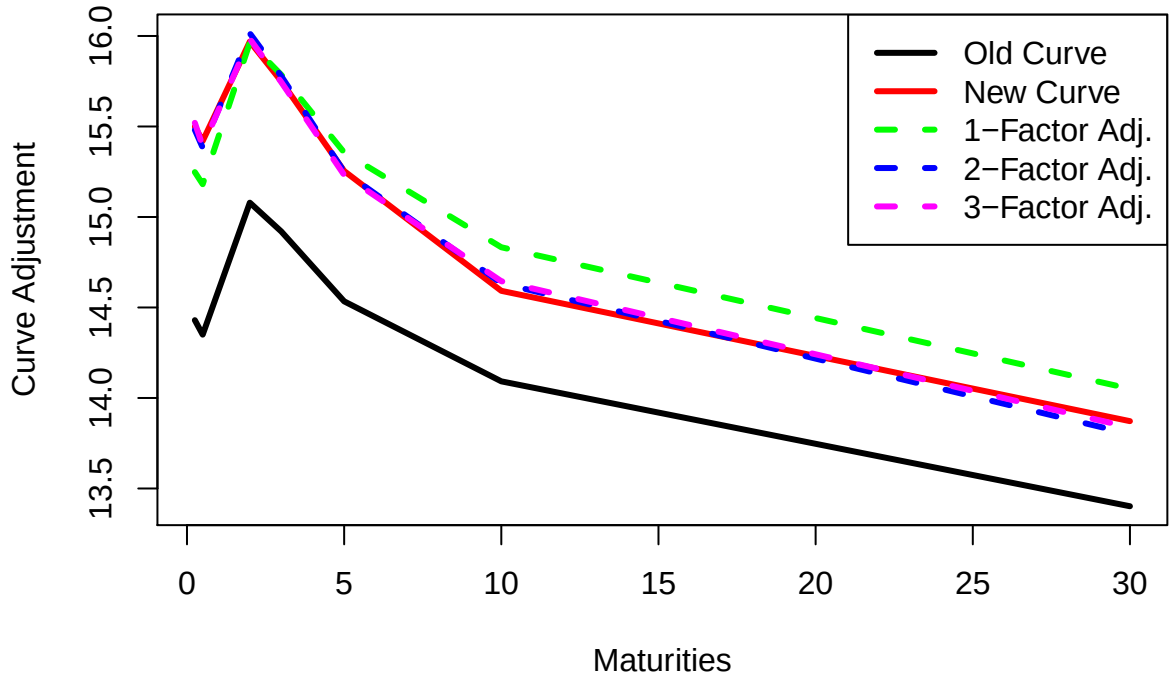



```
plot(Factors[,1],Factors[,2],type="l",lwd=2)
```



Draw at least three conclusions from the plot of the first two factors above. The data and relationships change over time. There is a period in the middle where both factors were increasing together and there was also a period where factor 1 stayed elevated as factor 2 fell lower. Additionally, there have been periods of both high and low volatility between the factors.

```
#Analyze the adjustments that each factor makes to the term curve.
OldCurve<-AssignmentData[135,]
NewCurve<-AssignmentData[136,]
CurveChange<-NewCurve-OldCurve
FactorsChange<-Factors[136,]-Factors[135,]
ModelCurveAdjustment.1Factor<-OldCurve+t(Loadings[,1])*FactorsChange[1]
ModelCurveAdjustment.2Factors<-OldCurve+t(Loadings[,1])*FactorsChange[1]+t(Loadings[,2])*FactorsChange[
ModelCurveAdjustment.3Factors<-OldCurve+t(Loadings[,1])*FactorsChange[1]+t(Loadings[,2])*FactorsChange[
  t(Loadings[,3])*FactorsChange[3]
matplot(Maturities,
        t(rbind(OldCurve,NewCurve,ModelCurveAdjustment.1Factor,ModelCurveAdjustment.2Factors,
                ModelCurveAdjustment.3Factors)),
        type="l",lty=c(1,1,2,2,2),col=c("black","red","green","blue","magenta"),lwd=3,ylab="Curve Adjus
legend(x="topright",c("Old Curve","New Curve","1-Factor Adj.","2-Factor Adj.",
                      "3-Factor Adj."),lty=c(1,1,2,2,2),lwd=3,col=c("black","red","green","blue","magen
```




```
#combine CurveChange, ModelCurve Adjustment.3Factors-OldCurve
rbind(CurveChange,ModelCurveAdjustment.3Factors-OldCurve)
```

```
##                USGG3M   USGG6M   USGG2YR   USGG3YR   USGG5YR  USGG10YR
## CurveChange 1.070000 1.070000 0.8900000 0.8300000 0.7200000 0.5000000
##             1.090063 1.041267 0.9046108 0.8248257 0.6979317 0.5531734
##               USGG30YR
## CurveChange 0.4700000
##             0.4357793
```

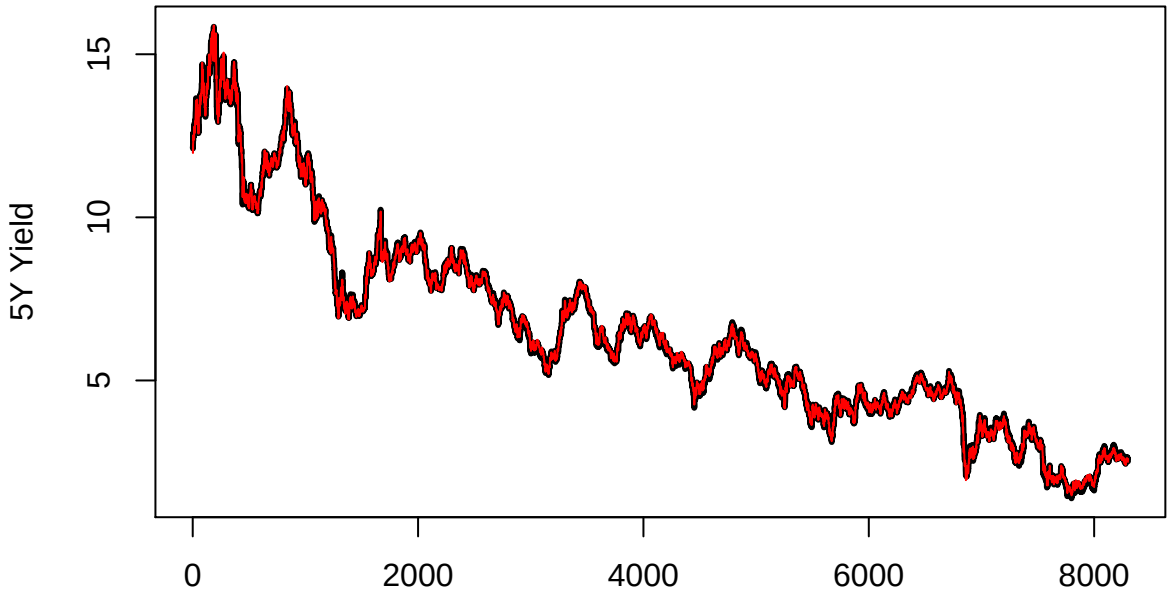
Explain how shapes of the loadings affect the adjustnents using only factor 1, factors 1 and 2, and all 3 factors. The graph shows that by adding only factor 1, that it doesn't completely match the new model. There should be 2 or 3 factors used to get the best fit to the New curve.

```
# How close is the approximation for each maturity?
# 5Y
cbind(Maturities,Loadings)
```

```
##      Maturities
## [1,]       0.25 0.3839609 -0.50744508  0.5298222  0.40373501  0.3860878
## [2,]       0.50 0.3901870 -0.43946144  0.1114737 -0.40526448 -0.6787624
## [3,]       2.00 0.4151851 -0.11112721 -0.4187873 -0.40896949  0.3787209
## [4,]       3.00 0.4063541  0.01696988 -0.4476561  0.06433748  0.2362448
## [5,]       5.00 0.3860610  0.23140317 -0.2462364  0.53357656 -0.2868460
## [6,]      10.00 0.3477544  0.43245979  0.1500903  0.19856539 -0.2562426
```

```
## [7,]      30.00 0.3047124  0.54421228  0.4979195 -0.42098839  0.2074508
##
## [1,] -0.03976285  0.026742547
## [2,]  0.09475452 -0.090913541
## [3,] -0.29848638  0.490009873
## [4,]  0.19760026 -0.731570606
## [5,]  0.42125768  0.438559615
## [6,] -0.73561857 -0.152627535
## [7,]  0.37776687  0.009199827
```

```
Model.10Y<-Means[6,6]+Loadings[6,1]*Factors[,1]+Loadings[6,2]*Factors[,2]+Loadings[6,3]*Factors[,3]
matplot(cbind(AssignmentData[,6],Model.10Y),type="l",lty=1,lwd=c(3,1),col=c("black","red"),ylab="5Y Yie
```



Repeat the PCA using princomp

```
# Do PCA analysis using princomp()
PCA.Yields<-princomp(AssignmentData)
names(PCA.Yields)
```

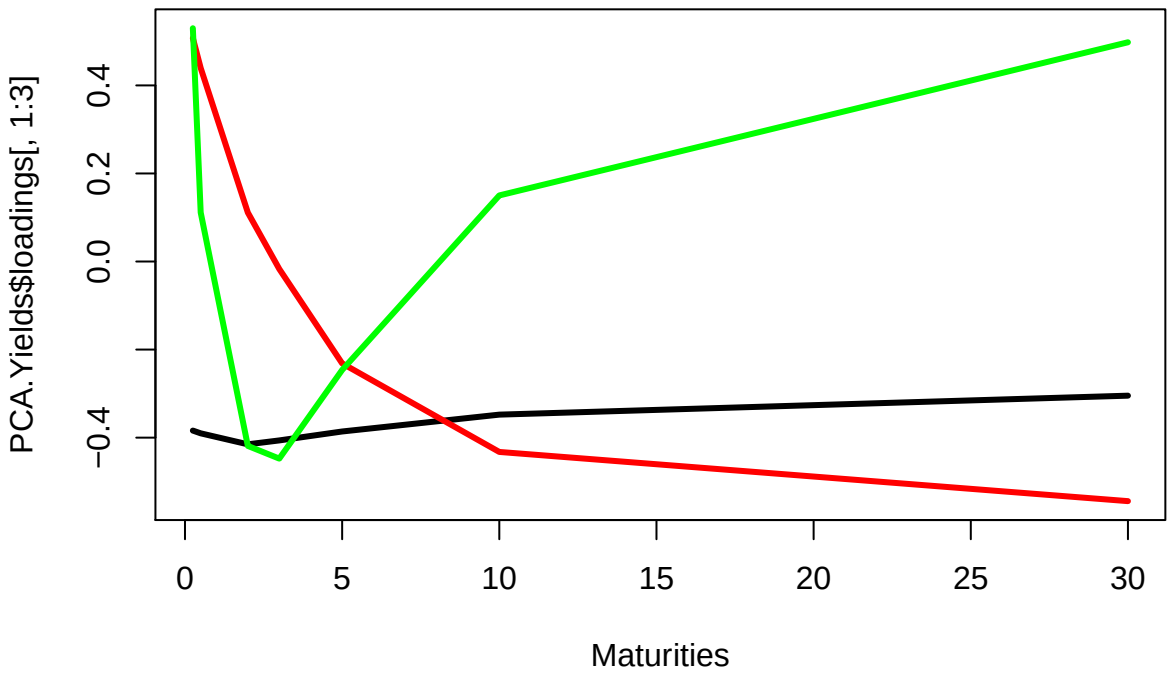
```
## [1] "sdev"     "loadings" "center"   "scale"    "n.obs"    "scores"
## [7] "call"
```

```
# Check that the loadings are the same
cbind(PCA.Yields$loadings[,1:3],Maturities,Eigen.Decomposition$vectors[,1:3])
```

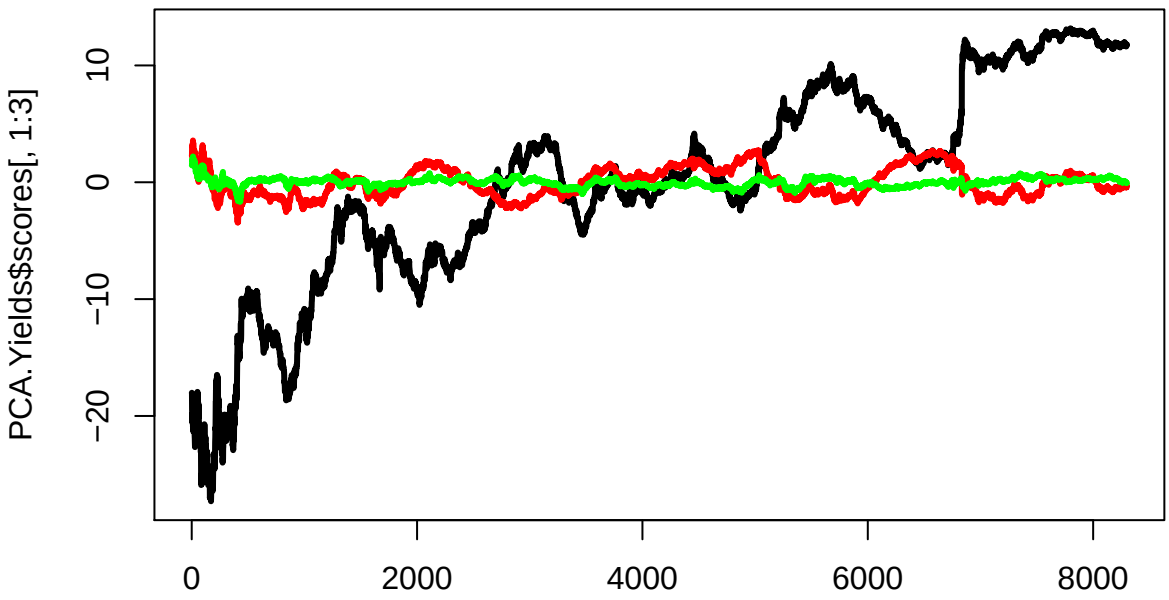
```
##              Comp.1      Comp.2     Comp.3 Maturities
```

```
## USGG3M   -0.3839609  0.50744508  0.5298222       0.25 -0.3839609
## USGG6M   -0.3901870  0.43946144  0.1114737       0.50 -0.3901870
## USGG2YR  -0.4151851  0.11112721 -0.4187873       2.00 -0.4151851
## USGG3YR  -0.4063541 -0.01696988 -0.4476561       3.00 -0.4063541
## USGG5YR  -0.3860610 -0.23140317 -0.2462364       5.00 -0.3860610
## USGG10YR -0.3477544 -0.43245979  0.1500903      10.00 -0.3477544
## USGG30YR -0.3047124 -0.54421228  0.4979195      30.00 -0.3047124
##
## USGG3M   -0.50744508  0.5298222
## USGG6M   -0.43946144  0.1114737
## USGG2YR  -0.11112721 -0.4187873
## USGG3YR   0.01696988 -0.4476561
## USGG5YR   0.23140317 -0.2462364
## USGG10YR  0.43245979  0.1500903
## USGG30YR  0.54421228  0.4979195
```

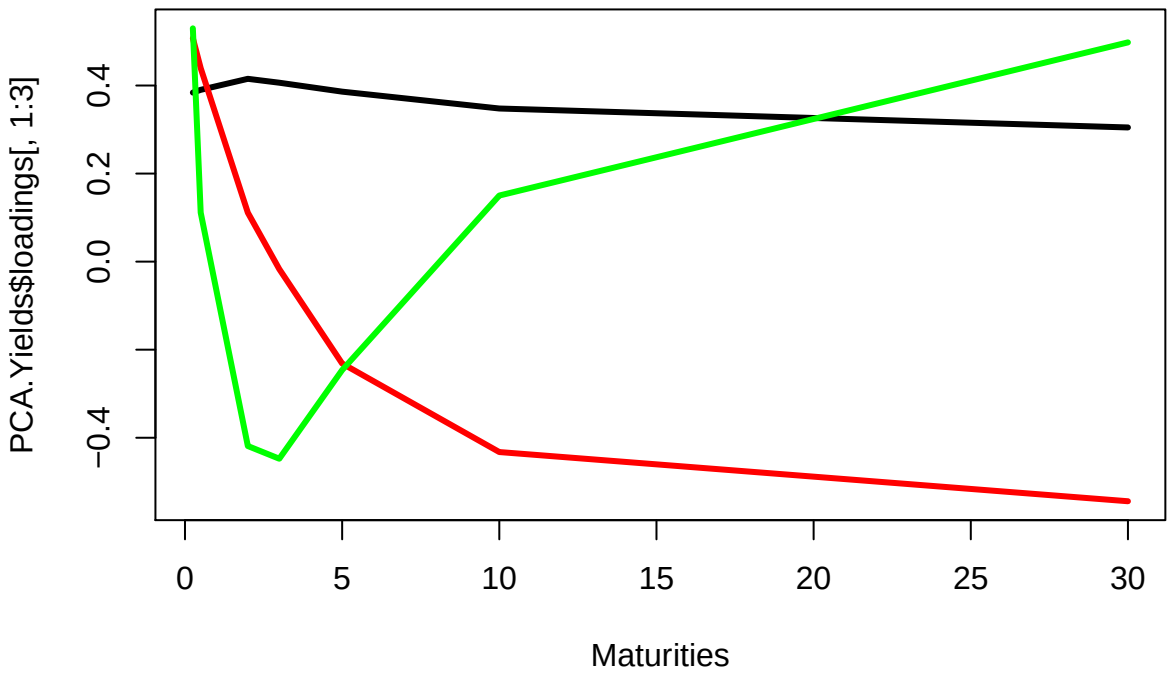
```
#Plot loadings from PCA
matplot(Maturities,PCA.Yields$loadings[,1:3],type="l",col=c("black","red","green"),lty=1,lwd=3)
```



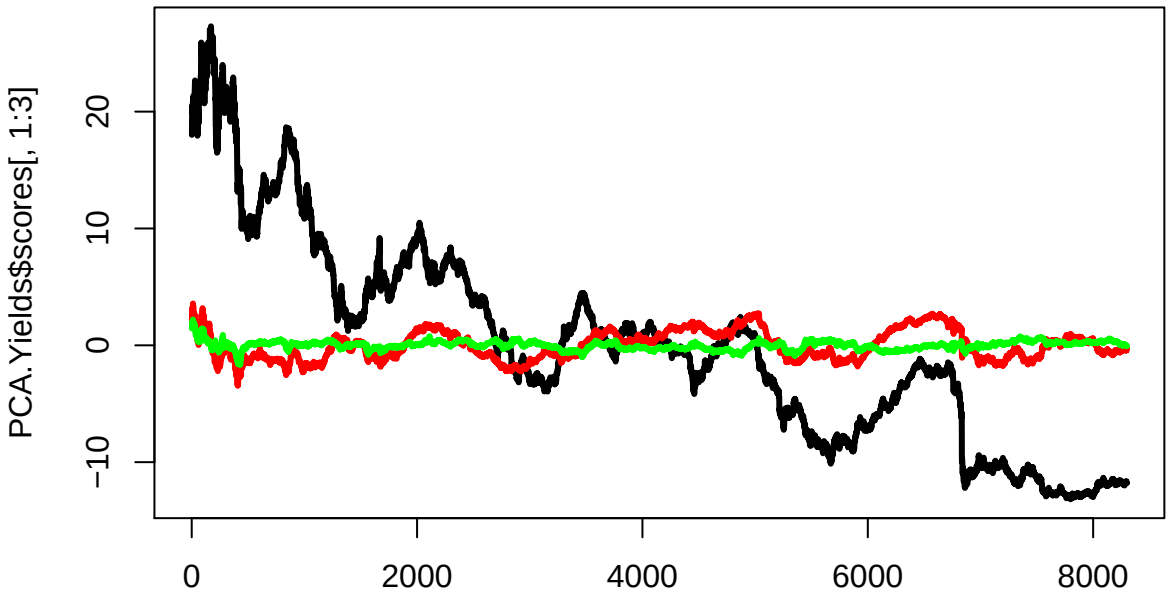

```
#Plot the factors calculated from PCA
matplot(PCA.Yields$scores[,1:3],type="l",col=c("black","red","green"),lwd=3,lty=1)
```



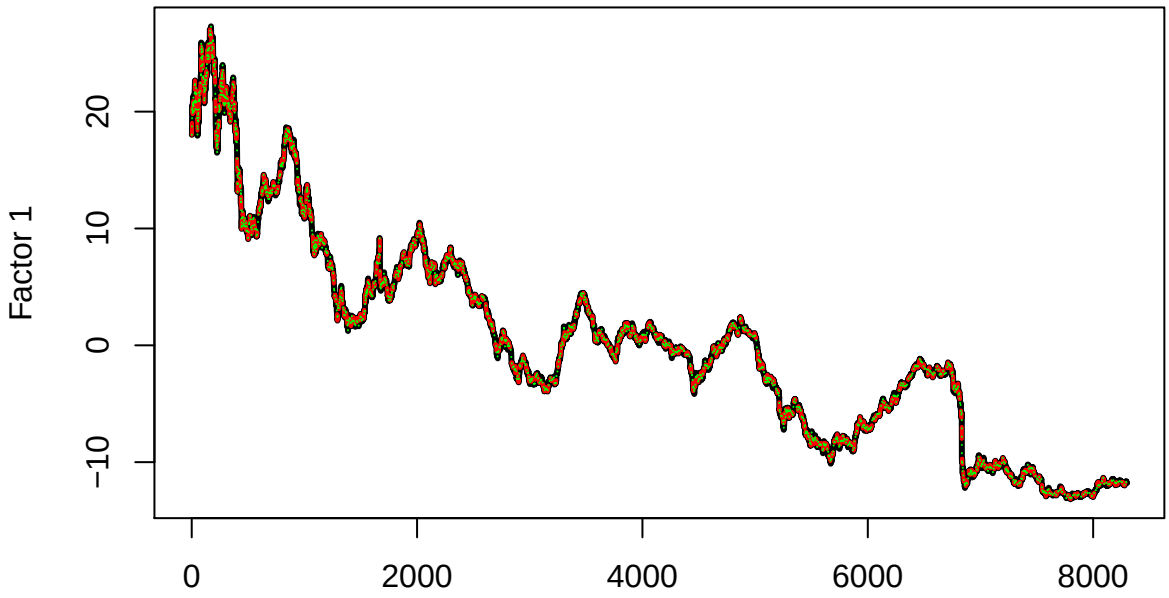

```
# Change the signs of the 1st factor and the first loading
PCA.Yields$loadings[,1]<--PCA.Yields$loadings[,1]
PCA.Yields$scores[,1]<--PCA.Yields$scores[,1]
matplot(Maturities,PCA.Yields$loadings[,1:3],type="l",col=c("black","red","green"),lty=1,lwd=3)
```




```
#Plot 3 factors from PCA
matplot(PCA.Yields$scores[,1:3],type="l",col=c("black","red","green"),lwd=3,lty=1)
```




```
# What variable we had as Output?
matplot(cbind(PCA.Yields$scores[,1],AssignmentData.Output,Factors[,1]),type="l",col=c("black","red","gr
```



Compare the regression coefficients from Step 2 and Step 3 with factor loadings.

First, look at the slopes for AssignmentData.Input~AssignmentData.Output

```
t(apply(AssignmentData, 2, function(AssignmentData.col) lm(AssignmentData.col~AssignmentData.Output)$co
```

```
##          (Intercept) AssignmentData.Output
## USGG3M      4.675134             0.3839609
## USGG6M      4.844370             0.3901870
## USGG2YR     5.438888             0.4151851
## USGG3YR     5.644458             0.4063541
## USGG5YR     6.009421             0.3860610
## USGG10YR    6.481316             0.3477544
## USGG30YR    6.869355             0.3047124
```

```
cbind(PCA.Yields$center,PCA.Yields$loadings[,1])
```

```
##              [,1]      [,2]
## USGG3M   4.675134 0.3839609
## USGG6M   4.844370 0.3901870
## USGG2YR  5.438888 0.4151851
## USGG3YR  5.644458 0.4063541
## USGG5YR  6.009421 0.3860610
## USGG10YR 6.481316 0.3477544
## USGG30YR 6.869355 0.3047124
```

This shows that the zero loading equals the vector of intercepts of models Y~Output1, where Y is one of the columns of yields in the data. Also, the slopes of the same models are equal to the first loading.

Check if the same is true in the opposite direction: is there a correspondence between the coefficients of models Output1~Yield and the first loading.

```
AssignmentData.Centered<-t(apply(AssignmentData,1,function(AssignmentData.row) AssignmentData.row-PCA.Y
dim(AssignmentData.Centered)
```

```
## [1] 8300    7
```

```
t(apply(AssignmentData.Centered, 2, function(AssignmentData.col) lm(AssignmentData.Output~AssignmentData
```

```
##           (Intercept) AssignmentData.col
## USGG3M   1.420077e-11           2.507561
## USGG6M   1.421187e-11           2.497235
## USGG2YR  1.419747e-11           2.400449
## USGG3YR  1.419989e-11           2.455793
## USGG5YR  1.419549e-11           2.568742
## USGG10YR 1.420297e-11           2.786991
## USGG30YR 1.420965e-11           3.069561
```

```
#Use all inputs together
t(lm(AssignmentData.Output~AssignmentData.Centered)$coef)[-1]
```

```
## [1] 0.3839609 0.3901870 0.4151851 0.4063541 0.3860610 0.3477544 0.3047124
```

```
PCA.Yields$loadings[,1]
```

```
##    USGG3M    USGG6M   USGG2YR   USGG3YR   USGG5YR  USGG10YR  USGG30YR
## 0.3839609 0.3901870 0.4151851 0.4063541 0.3860610 0.3477544 0.3047124
```

```
#This means that the factor is a portfolio of all input variables with weights.
PCA.Yields$loadings[,1]
```

```
##    USGG3M    USGG6M   USGG2YR   USGG3YR   USGG5YR  USGG10YR  USGG30YR
## 0.3839609 0.3901870 0.4151851 0.4063541 0.3860610 0.3477544 0.3047124
```