Regression Analysis on US Treasuary Yield to Maturity Dataset

Anupriya Thirumurthy

The name of the data file for this project is RegressionAssignment-Data2014.csv.

```
# Clean the environment
rm(list=ls())
```

Step 1

```
# Read the data
# Visualize and get familiar with variables.
datapath<-'/Users/anupriyathirumurthy/Documents/AnuBackUp/University/MScA_UoC/Courses/StatisticalAnalysi
AssignmentData<- read.csv(file=paste(datapath,"regressionassignmentdata2014.csv",sep="/"),row.names=1,he
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
```

```
AssignmentData1 <- AssignmentData
```

Plotting input variables

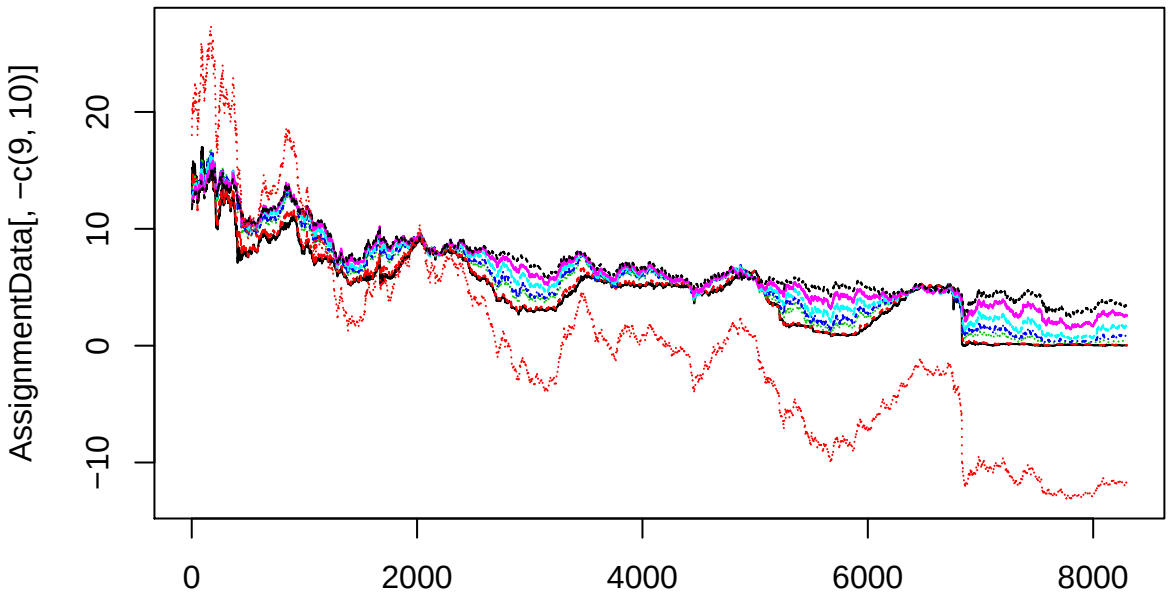
All the inputs seemed to look correlated with each other

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

Plotting input variables together with the output variable.

All the inputs have an effect on the output variable. For every change in the input variable, there is subsequent change in the output. When the inputs rise, the outputs also rise and when they fall, there is a similar fall in the output.

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



Step 2

#Estimating a simple regression model with each of the input and the output variable.

We could infer that all the predictors are significant in their respective individual models with one predictor and the intercept.

```
# Analyze the summary.
Input <- AssignmentData[,c(1:7)]
lm <- apply(Input,2, function(linearmodel) lm(AssignmentData$Output1~linearmodel))
cat("\nSimple Linear Model for input: USGG3M")
```

```
##
## Simple Linear Model for input: USGG3M
```

```
summary(lm$USGG3M)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ linearmodel)
##
## 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 ***
## linearmodel   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
cat("\n\nSimple Linear Model for input: USGG6M")
```

```
##
##
## Simple Linear Model for input: USGG6M
```

```r
summary(lm$USGG6M)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ linearmodel)
##
## 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 ***
## linearmodel   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
```

```r
cat("\n\nSimple Linear Model for input: USGG2YR")
```

```
##
##
## Simple Linear Model for input: USGG2YR
```

```r
summary(lm$USGG2YR)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ linearmodel)
##
## 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 ***
## linearmodel   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
```

```r
cat("\n\nSimple Linear Model for input: USGG3YR")
```

```
##
```

```
##
## Simple Linear Model for input: USGG3YR
```

```
summary(lm$USGG3YR)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ linearmodel)
##
## 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 ***
## linearmodel   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
```

```
cat("\n\nSimple Linear Model for input: USGG5YR")
```

```
##
##
## Simple Linear Model for input: USGG5YR
```

```
summary(lm$USGG5YR)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ linearmodel)
##
## 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 ***
## linearmodel   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
```

```
cat("\n\nSimple Linear Model for input: USGG10YR")
```

```
##
##
## Simple Linear Model for input: USGG10YR
```

```
summary(lm$USGG10YR)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ linearmodel)
##
## 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 ***
## linearmodel   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
```

```
cat("\n\nSimple Linear Model for input: USGG30YR")
```

```
##
##
## Simple Linear Model for input: USGG30YR
```

```
summary(lm$USGG30YR)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ linearmodel)
##
## 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 ***
## linearmodel   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
```

Analyze the summary.

1. Residuals : We can notice that the median is almost close to zero in all these individual models and also the distribution of the residuals appears to be nearly strongly symmetrical. That means that the model predicts almost all the points close to the actual observed points.
2. Intercepts are low, indicating the predictors have significance

3. We can observe a highly significant p-values in all the models indicating that all the predictors are being significant in the respective individual models.
4. R squared and Adj R squared : In multiple regression settings, the R2 will always increase as more variables are included in the model. That's why the adjusted R2 is the preferred measure as it adjusts for the number of variables considered. However as this is a simple intercept and one predictor model, we can see R squared and say that roughly 96% of the variance found in the response variable (output1) can be explained by the predictor variable (USGG3M) considering the summary of USGG3M model. Similarly we can infer for other models. The predictor model which explains the more variance of the output is USGG3YR and is 99.8% of the variance found in the response variable (output1). 1-R^2 would give us the variance unexplained.
5. Correlation explained: All the predictors and the outputs are perfectly correlated as they have a strong association between them.

```
#Checking relevance of the estimated parameters and the model as a whole, amount of correlation explain

#The following code gives the analysis for the input variables and plots the output variable together w

cat("\nVariance and Coefficients for input: USGG3M\n")
```

```
##
## Variance and Coefficients for input: USGG3M
```

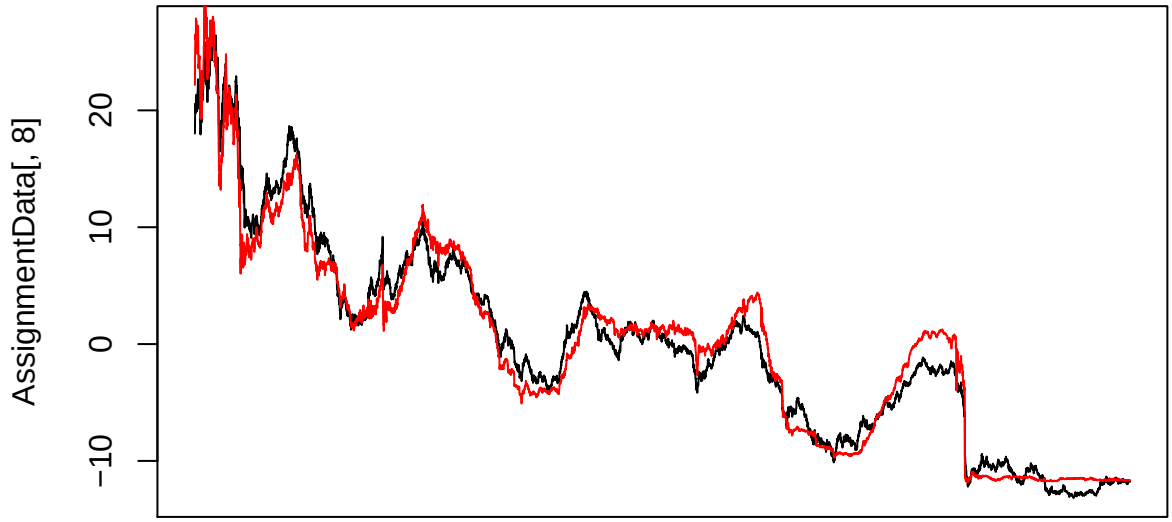
```
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(lm$USGG3M)$sigma^2)
```

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

```
lm$USGG3M$coefficients
```

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

```
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(lm$USGG3M$fitted.values,col="red")
```




```
cat("\n\nVariance and Coefficients for input: USGG6M\n")
```

```
##
##
## Variance and Coefficients for input: USGG6M
```

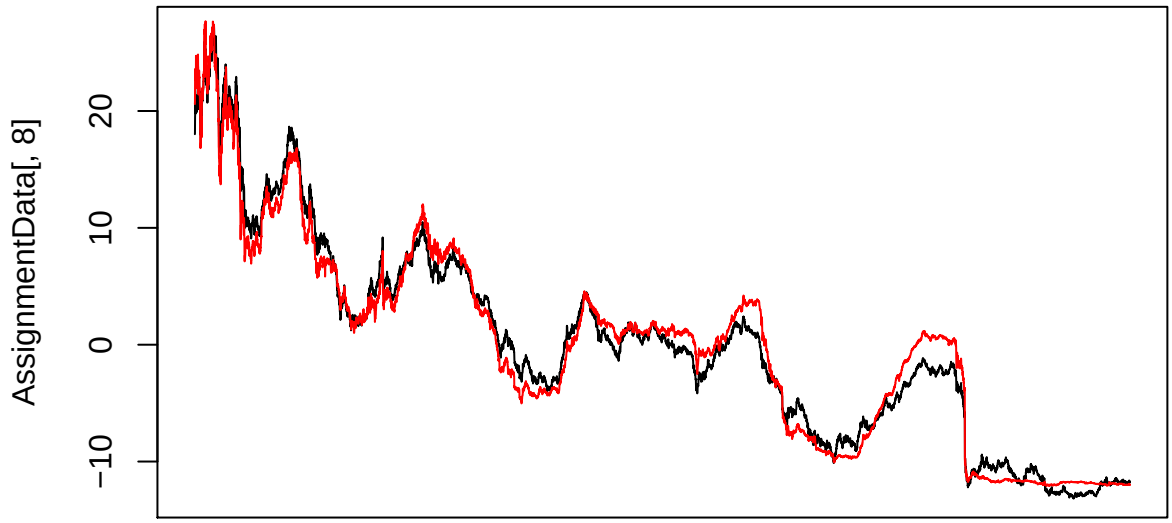
```
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(lm$USGG6M)$sigma^2)
```

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

```
lm$USGG6M$coefficients
```

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

```
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(lm$USGG6M$fitted.values,col="red")
```




```
cat("\n\nVariance and Coefficients for input: USGG2YR\n")
```

```
##
##
## Variance and Coefficients for input: USGG2YR
```

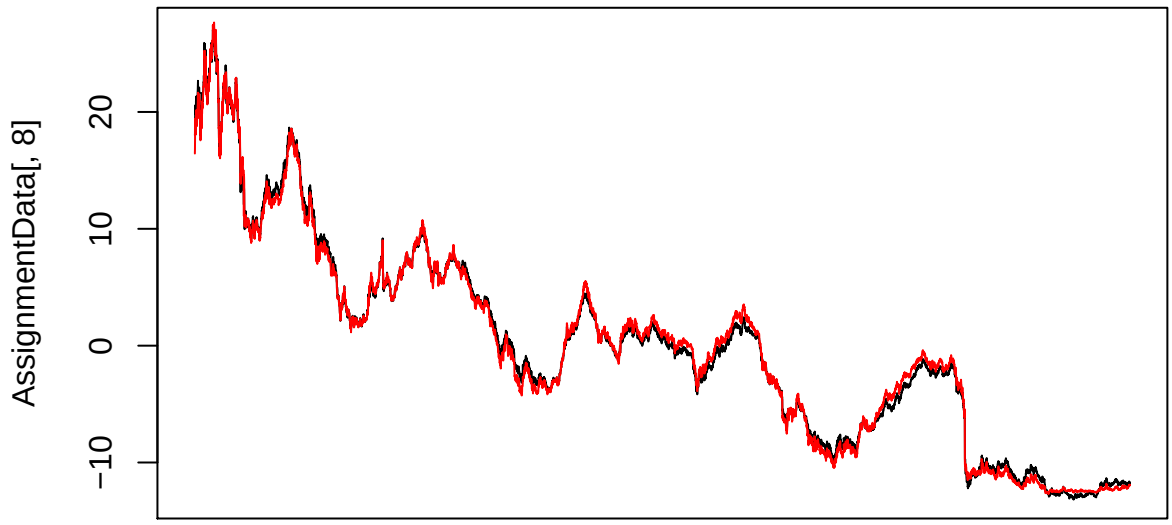
```
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(lm$USGG2YR)$sigma^2)
```

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

```
lm$USGG2YR$coefficients
```

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

```
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(lm$USGG2YR$fitted.values,col="red")
```




```
cat("\n\nVariance and Coefficients for input: USGG3YR\n")
```

```
##
##
## Variance and Coefficients for input: USGG3YR
```

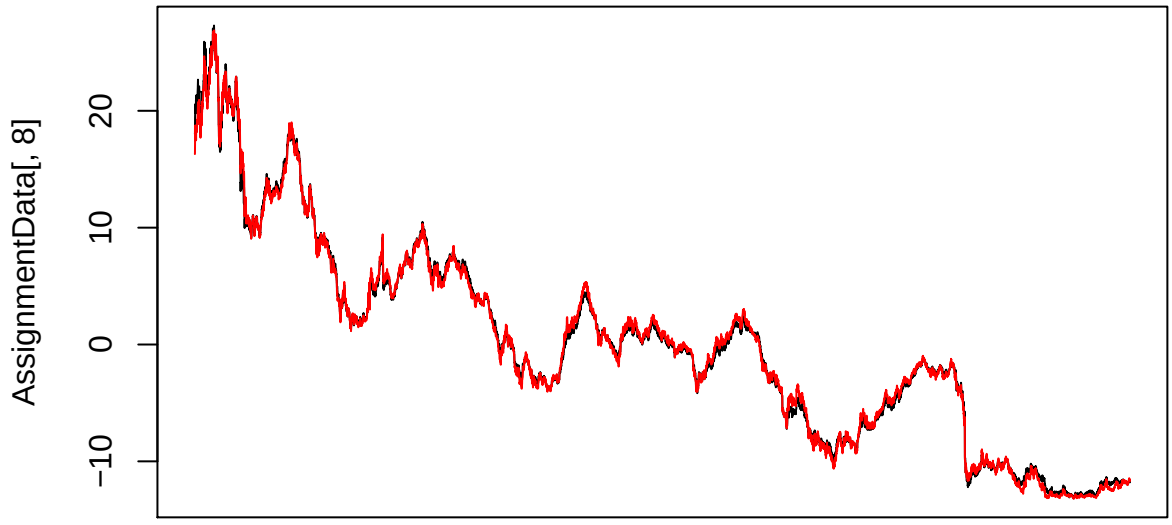
```
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(lm$USGG3YR)$sigma^2)
```

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

```
lm$USGG3YR$coefficients
```

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

```
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(lm$USGG3YR$fitted.values,col="red")
```




```
cat("\n\nVariance and Coefficients for input: USGG5YR\n")
```

```
##
##
```

```
## Variance and Coefficients for input: USGG5YR
```

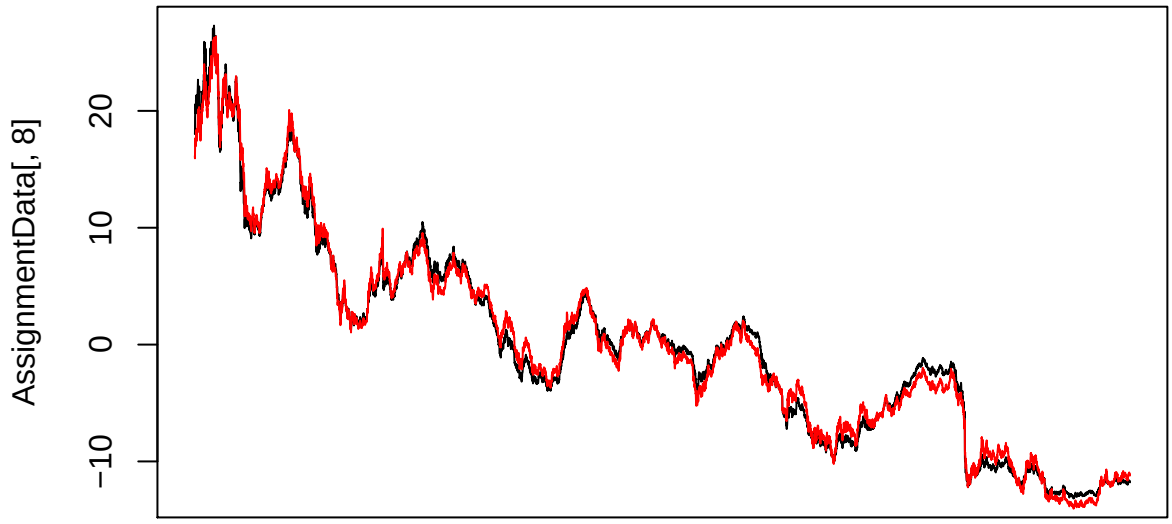
```
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(lm$USGG5YR)$sigma^2)
```

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

```
lm$USGG5YR$coefficients
```

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

```
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(lm$USGG5YR$fitted.values,col="red")
```




```
cat("\n\nVariance and Coefficients for input: USGG10YR\n")
```

```
##
##
## Variance and Coefficients for input: USGG10YR
```

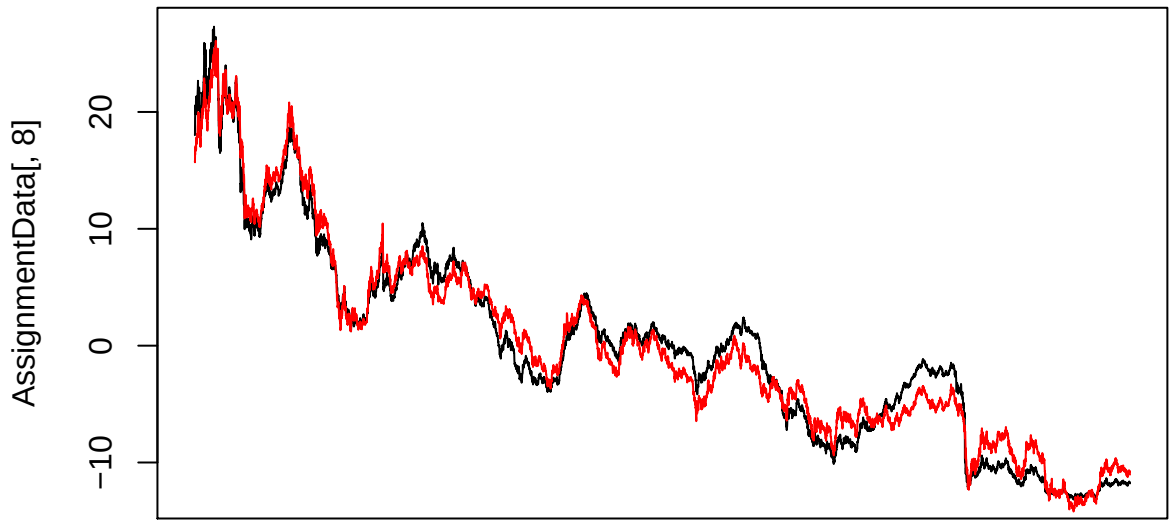
```
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(lm$USGG10YR)$sigma^2)
```

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

```
lm$USGG10YR$coefficients
```

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

```
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(lm$USGG10YR$fitted.values,col="red")
```




```
cat("\n\nVariance and Coefficients for input: USGG30YR\n")
```

```
##
##
## Variance and Coefficients for input: USGG30YR
```

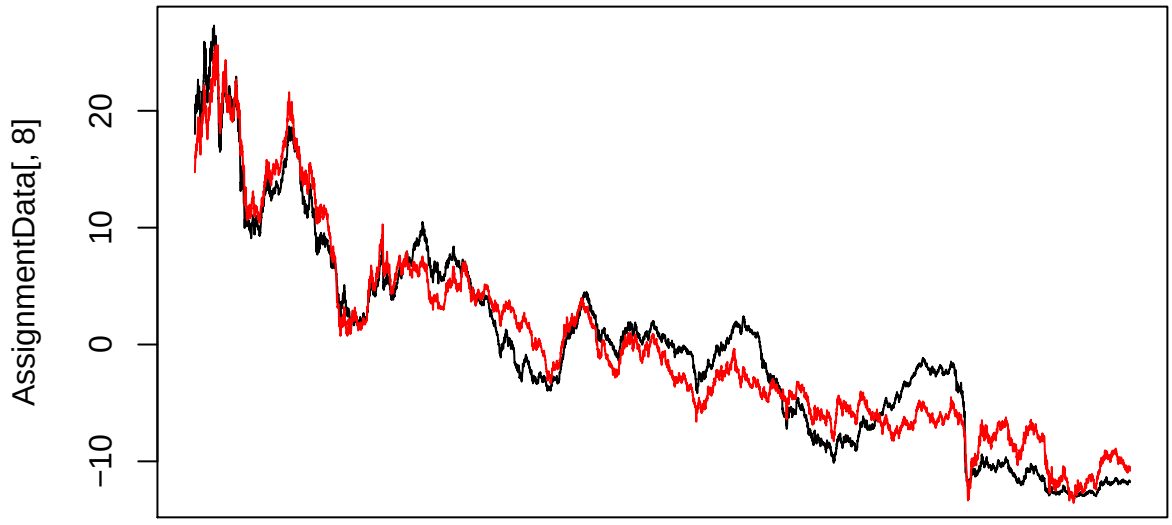
```
c(Total.Variance=var(AssignmentData[,8]),Unexplained.Variance=summary(lm$USGG30YR)$sigma^2)
```

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

```
lm$USGG30YR$coefficients
```

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

```
matplot(AssignmentData[,8],type="l",xaxt="n")
lines(lm$USGG30YR$fitted.values,col="red")
```



Plotting the output variable together with the fitted values. Comparisons and contrasts when we situate each model relative to the expectations or to the other models.

Inference when we fit the models and plot them are:

1. Among all the single predictor and intercept model, The predictor model which explains the more variance of the output is USGG3YR and is 99.8% of the variance found in the response variable (output1). Hence in the plot the red line and the black lines are so close to each other or they collide.
2. Among all the single predictor and intercept model, The predictor model which explains the variance of the output relatively less is USGG30YR and is 93.5% of the variance found in the response variable (output1). Hence in the plot the red line and the black lines are a bit varied when compared with the other models.

Collectting all slopes and intercepts in one table

```
lm_table <- apply(Input,2, function(coeff) (lm(AssignmentData$Output1~coeff))$coefficients)
lm_table
```

```
##                  USGG3M     USGG6M    USGG2YR    USGG3YR    USGG5YR
## (Intercept) -11.723184 -12.097528 -13.055775 -13.861618 -15.436649
## coeff         2.507561   2.497235   2.400449   2.455793   2.568742
##                USGG10YR   USGG30YR
## (Intercept) -18.063370 -21.085905
## coeff         2.786991   3.069561
```

Step 3.

```
# Fitting linear regression models using single output (column 8 Output1) as input and each of the orig
lm1 <- apply(Input,2, function(coeff) (lm(coeff~AssignmentData$Output1)))
summary(lm1$USGG3M)
```

```
##
## Call:
## lm(formula = coeff ~ AssignmentData$Output1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.86665 -0.51316 -0.04857  0.44260  3.07711
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             4.6751341  0.0072600   644.0   <2e-16 ***
## AssignmentData$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
```

```
summary(lm1$USGG6M)
```

```
##
## Call:
## lm(formula = coeff ~ AssignmentData$Output1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.51908 -0.44307 -0.04514  0.40876  1.71754
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            4.8443695  0.0060856   796.0   <2e-16 ***
## AssignmentData$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
```

```
summary(lm1$USGG2YR)
```

```
##
## Call:
## lm(formula = coeff ~ AssignmentData$Output1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.69112 -0.18514  0.00422  0.15956  0.61596
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            5.438888   0.002322    2342   <2e-16 ***
## AssignmentData$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
```

```
summary(lm1$USGG3YR)
```

```
##
## Call:
## lm(formula = coeff ~ AssignmentData$Output1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.23099 -0.10862 -0.01352  0.10095  0.83112
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            5.6444580  0.0017841    3164   <2e-16 ***
## AssignmentData$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
```

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

```
lm1_table <- apply(Input,2, function(coeff) (lm(coeff~AssignmentData$Output1))$coefficients)
lm1_table
```

```
##                            USGG3M   USGG6M   USGG2YR   USGG3YR  USGG5YR
## (Intercept)            4.6751341 4.844370 5.4388879 5.6444580 6.009421
## AssignmentData$Output1 0.3839609 0.390187 0.4151851 0.4063541 0.386061
##                          USGG10YR  USGG30YR
## (Intercept)            6.4813160 6.8693554
## AssignmentData$Output1 0.3477544 0.3047124
```

Summary of this table and interesting fact when compared with step 2

The intercept is the expected mean value of Y when all X=0. In a regression equation with one predictor, X. If X sometimes = 0, the intercept is simply the expected mean value of Y at that value.

Notice that the intercepts of the models in step 2 are all negative and it doesn't really makes any sense. However, the intercepts in step 3 are positive indicating that the when the output is zero, the input increases by the intercept value. But actually i feel, the regression constant is generally not worth interpreting. Despite this, it is almost always a good idea to include the constant in your regression analysis. In the end, the real value of a regression model is the ability to understand how the response variable changes when you change the values of the predictor variables.

Step 4.

```
# Estimating logistic regression using all inputs and the data on FED tightening and easing cycles.
AssignmentDataLogistic<-data.matrix(AssignmentData,rownames.force="automatic")
# Preparing the easing-tightening data.
# Making the easing column equal to 0 during the easing periods and NA otherwise.
# Making the tightening column equal to 1 during the tightening periods and NA otherwise.
# Creating columns of easing periods (as 0s) and tightening periods (as 1s)
EasingPeriods<-AssignmentDataLogistic[,9]
EasingPeriods[AssignmentDataLogistic[,9]==1]<-0
TighteningPeriods<-AssignmentDataLogistic[,10]
# Checking 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
```

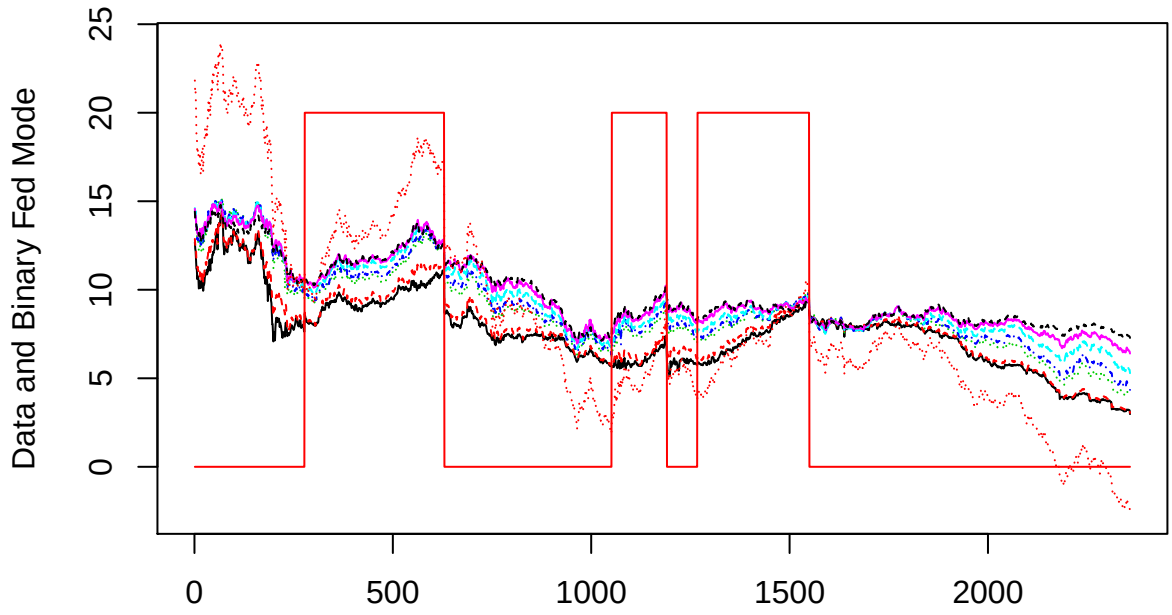
```
## 4/6/1983              NA                1
## 4/7/1983              NA                1
## 4/8/1983              NA                1
## 4/11/1983             NA                1
## 4/12/1983             NA                1
## 4/13/1983             NA                1
## 8/27/1984             NA                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
```

```
# Removing 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]
```

Plotting the data and the binary output variable representing easing (0) and tightening (1) periods.

Periods of Easing and low interest rates appear to occur in similar timeframes and seemed to be tightly coupled. Tightening periods seem to match the periods of rise in all interest rates.

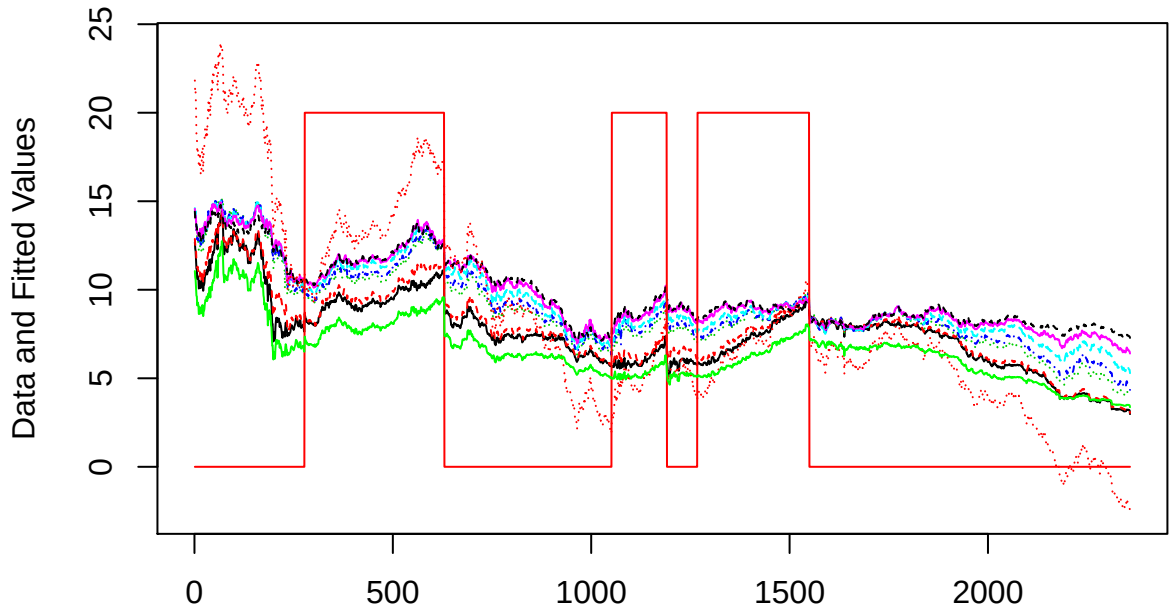
```
# Binary output for logistic regression is now in column 10
# Plotting the data and the binary output variable representing easing (0) and tightening (1) periods.
matplot(AssignmentDataLogistic.EasingTighteningOnly[,-c(9,10)],type="l",ylab="Data and Binary Fed Mode"
lines(AssignmentDataLogistic.EasingTighteningOnly[,10]*20,col="red")
```




```
# 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")
```



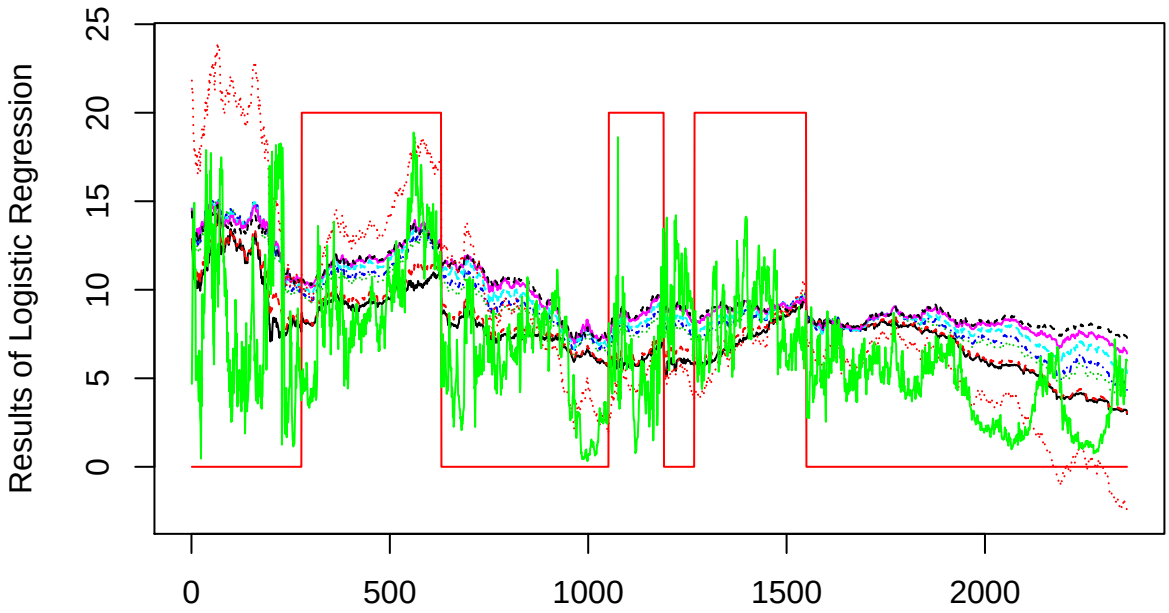

```
#Using all inputs as predictors for logistic regression.
LogisticModel.TightteningEasing_All<-glm(AssignmentDataLogistic.EasingTightteningOnly[,10]~
                                    AssignmentDataLogistic.EasingTightteningOnly[,c(1:7)],family=binom
summary(LogisticModel.TightteningEasing_All)$aic
```

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

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

```
##                                                                 Estimate
## (Intercept)                                                   -4.7551928
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG3M  -3.3456116
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG6M   4.1558535
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG2YR  3.9460296
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG3YR -3.4642455
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG5YR -3.2115319
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG10YR -0.9705444
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG30YR  3.3253517
##                                                                   Pr(>|z|)
## (Intercept)                                                   2.784283e-28
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG3M  4.073045e-36
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG6M  1.422964e-28
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG2YR 1.751687e-07
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG3YR 2.080617e-04
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG5YR 3.786229e-05
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG10YR 3.202140e-01
## AssignmentDataLogistic.EasingTightteningOnly[, c(1:7)]USGG30YR 6.036041e-08
```

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



Interpreting the coefficients of the model and the fitted values.

Logistic regression coefficients are reported as log odds. Log odds could be converted to normal odds using the exponential function.

For instance considering the logistic coefficient of USGG6M = 4.156, we could infer that there is a multiplicative effect of logodds equals to e^4 = 55 meaning, we are multiplying the odds of tightening being 1 by 55 times with a one unit shift in one year rate, which seems to be unbelievable. A one unit increase in the 6M rates meaning that the odds go down 55 times, one year increase in the 1 year rate is that the odds go up 55 times.

Another example, a logistic regression coefficient of -0.97 of the USGG10YR predictor corresponds to odds of e–0.97=2.63794445935, meaning that the easing was about 3 times more likely (about 75/25) than tightening response, for each unit of change in USGG10YR. Predictor effect on output tightening is possitive.

2. The Sign of coefficients would represents a positive or negative influence on dependent variable.

Fitted Values:

In Logistic regression, Probability of the event (Easing) occuring p is the actual value we are interested in. Fitted values are log (p/p-1). So, in order to figure out the likelihood of p, we calculate it by using $e^{y/(e}y + 1)$

We could infer in our case the fitted probabilities never change from a factor of 2 to 3. Therefore, the fact that the odds are going up by huge amounts does not make any sense.

Plots

Tightening periods and easing periods seem to match the period of rise and fall of interest rates correspondingly.

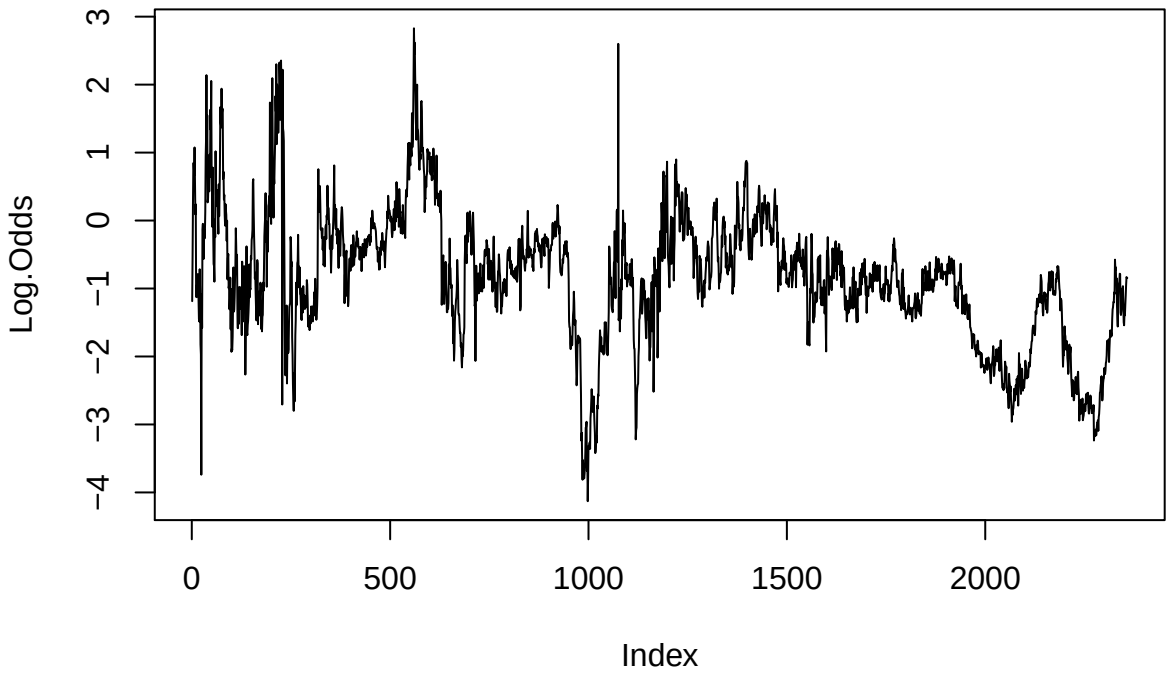
Fitting logistic model with all predictors.

Interpretting the summary/coefficients, contrast to previous model, and displaying the plot which overlays the predicted values.

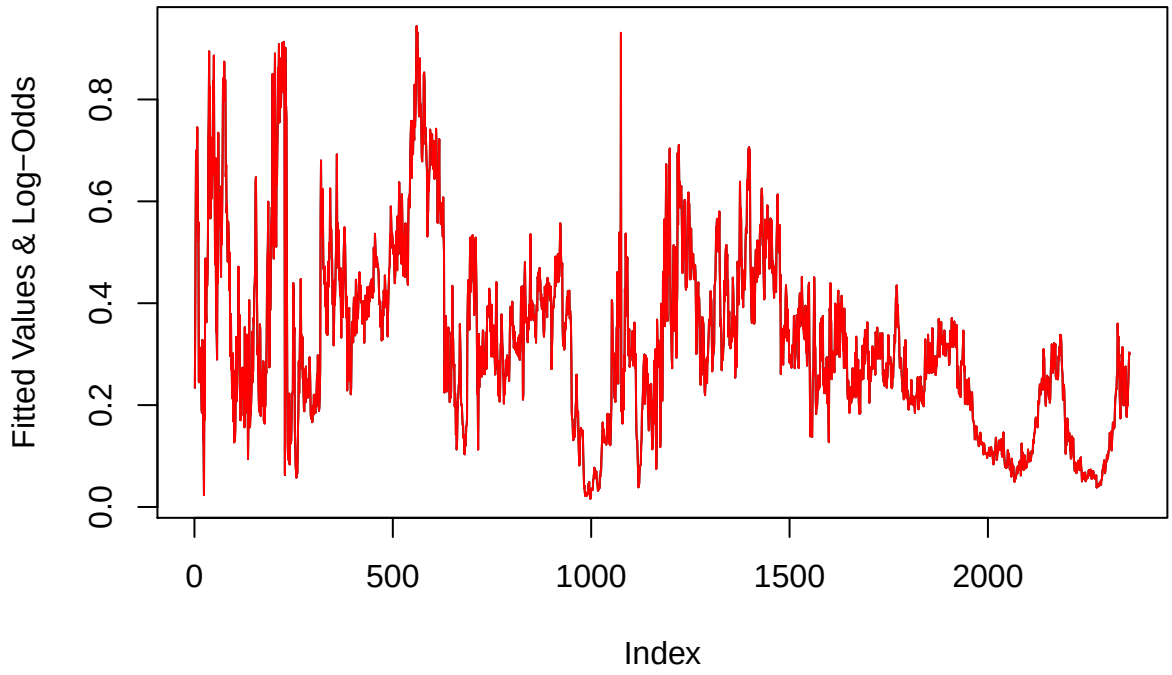
Some of coefficients have positive effect while others have negative effect on tightening, But overall the model is a better fit than the previous as we compare the AIC. Plot also suggests that this model better explains

the binary output than the previous model.

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




```
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.

```
# Comparing linear regression models with different combinations of predictors.
# Selecting the best combination.
AssignmentDataRegressionComparison<-data.matrix(AssignmentData[,-c(9,10)],rownames.force="automatic")
AssignmentDataRegressionComparison<-AssignmentData[,-c(9,10)]
# Estimating the full model by using all 7 predictors.
AssignmentDataRegressionComparison.fullModel<-lm(AssignmentData$Output1~.,data=AssignmentData[,1:7])

AssignmentDataRegressionComparison.fullModel$coefficients
```

```
## (Intercept)      USGG3M      USGG6M     USGG2YR     USGG3YR     USGG5YR
## -14.9041591   0.3839609   0.3901870   0.4151851   0.4063541   0.3860610
##    USGG10YR    USGG30YR
##   0.3477544   0.3047124
```

```
summary(AssignmentDataRegressionComparison.fullModel)$coefficients[,c(1,4)]
```

```
##                Estimate Pr(>|t|)
## (Intercept) -14.9041591        0
## USGG3M        0.3839609        0
## USGG6M        0.3901870        0
## USGG2YR       0.4151851        0
## USGG3YR       0.4063541        0
## USGG5YR       0.3860610        0
## USGG10YR      0.3477544        0
## USGG30YR      0.3047124        0
```

```
summary(AssignmentDataRegressionComparison.fullModel)$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(AssignmentDataRegressionComparison.fullModel)$r.squared,summary(AssignmentDataRegressionCompa
```

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

```
summary(AssignmentDataRegressionComparison.fullModel)$df
```

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

Looking at coefficients, R2, adjusted R2, degrees of freedom.

The coefficients are significant. Both R squared and adj R squared equals 1, and they say that 100% of the variance found in the response variable (output1) can be explained by the predictor variables used in the model. Thus it's a perfect fit.

Intepreting the fitted model.

We can infer that the p values are less than the level of significance, indicating that all predictors appear significant.

R-Squared and adjusted R squared suggests that the full model shows a perfect fit.

```
# Estimating the Null model by including only intercept.
AssignmentDataRegressionComparison.nullModel<-lm(AssignmentData$Output1~1,data=AssignmentData[,1:7])
summary(AssignmentDataRegressionComparison.nullModel)$coefficients
```

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

```
c(summary(AssignmentDataRegressionComparison.nullModel)$r.squared,summary(AssignmentDataRegressionCompa
```

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

```
summary(AssignmentDataRegressionComparison.nullModel)$df
```

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

Exploring the same parameters of the fitted null model as for the full model.

The predictor effects of null model are 0 and hence no R squared to explain the variance. We can also note that the p value is 1, which means insignificant, indicating that the only intercept model doesn't explain the output.

Why summary(RegressionModelComparison.Null) does not show R2 ?

R2 is nothing but the fraction of the total variance of Y that is "explained" by the variation in X.

The predictor effects of null model are 0. Hence none of the variance is explained by any of the predictors (or there are NO predictors at all to explain this). Hence the summary of null model does not show R2.

```
# Compare models pairwise using anova()
anova(AssignmentDataRegressionComparison.fullModel,AssignmentDataRegressionComparison.nullModel)
```

```
## Analysis of Variance Table
##
## Model 1: AssignmentData$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG3YR +
##     USGG5YR + USGG10YR + USGG30YR
## Model 2: AssignmentData$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
```

Interpreting the results of anova().

The hypotheses for the F-test of the overall significance in the ANOVA table are as follows:

Null hypothesis: The fit of the intercept-only model(null model) and full model are equal. Alternative hypothesis: The fit of the intercept-only model is significantly reduced compared to the full model.

As the P value for the F-test of overall significance test is less than the significance level, we can reject the null-hypothesis and conclude that the full model provides a better fit than the intercept-only model.

Repeating the analysis for different combinations of input variables and selecting the best one.

```
# Using drop to remove the predictors with low AIC
fwd.model <- step(AssignmentDataRegressionComparison.nullModel, direction='forward', scope=AssignmentDa
```

```
## Start:  AIC=36033.48
## AssignmentData$Output1 ~ 1
```

```
AssignmentData1 <- AssignmentData[,1:7]

drop1(AssignmentDataRegressionComparison.fullModel)
```

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

```
## Single term deletions
##
## Model:
## AssignmentData$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
```

```
AssignmentDataRegressionComparison.noUSGG3YR <- lm(AssignmentData$Output1~.,data=AssignmentData1[,-4])
summary(AssignmentDataRegressionComparison.noUSGG3YR)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ ., data = AssignmentData1[,
##     -4])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36972 -0.01132 -0.00141  0.00905  0.30202
##
## Coefficients:
##               Estimate Std. Error   t value Pr(>|t|)
## (Intercept) -14.863604   0.001358 -10942.56   <2e-16 ***
## USGG3M        0.389687   0.001332    292.51   <2e-16 ***
## USGG6M        0.357731   0.001997    179.09   <2e-16 ***
## USGG2YR       0.685589   0.001806    379.63   <2e-16 ***
```

```
## USGG5YR       0.577177   0.002853    202.33   <2e-16 ***
## USGG10YR      0.347515   0.003788     91.75   <2e-16 ***
## USGG30YR      0.270634   0.002085    129.77   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02113 on 8293 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 2.379e+08 on 6 and 8293 DF,  p-value: < 2.2e-16
```

```
anova(AssignmentDataRegressionComparison.fullModel,AssignmentDataRegressionComparison.noUSGG3YR)
```

```
## Analysis of Variance Table
##
## Model 1: AssignmentData$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG3YR +
##     USGG5YR + USGG10YR + USGG30YR
## Model 2: AssignmentData$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG5YR +
##     USGG10YR + USGG30YR
##   Res.Df    RSS Df Sum of Sq          F    Pr(>F)
## 1   8292 0.0000
## 2   8293 3.7037 -1   -3.7037 1.5172e+18 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# We should stop with the model with noUSGG3YR, as it has the least residual error while the full model

# Further trying to reduce, But i don't think its a best model if we reduce it any further.
# The below steps are only for analysis to drop and check

drop1(AssignmentDataRegressionComparison.noUSGG3YR)
```

```
## Single term deletions
##
## Model:
## AssignmentData$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG5YR +
##     USGG10YR + USGG30YR
##          Df Sum of Sq    RSS    AIC
## <none>                 3.704 -64018
## USGG3M    1    38.213 41.917 -43881
## USGG6M    1    14.325 18.028 -50884
## USGG2YR   1    64.366 68.070 -39857
## USGG5YR   1    18.282 21.986 -49237
## USGG10YR  1     3.760  7.463 -58204
## USGG30YR  1     7.521 11.225 -54817
```

```
AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR <- lm(AssignmentData$Output1~.,data=AssignmentDa
summary(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ ., data = AssignmentData1[,
##     c(-4, -6)])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.37829 -0.01722 -0.00095  0.01724  0.43850
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -14.908651   0.001798 -8293.0   <2e-16 ***
## USGG3M        0.383598   0.001889   203.1   <2e-16 ***
## USGG6M        0.366675   0.002832   129.5   <2e-16 ***
## USGG2YR       0.644176   0.002482   259.5   <2e-16 ***
## USGG5YR       0.791796   0.002318   341.6   <2e-16 ***
## USGG30YR      0.447951   0.001113   402.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03 on 8294 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.417e+08 on 5 and 8294 DF,  p-value: < 2.2e-16
```

```
anova(AssignmentDataRegressionComparison.noUSGG3YR,AssignmentDataRegressionComparison.noUSGG3YR.noUSGG1
```

```
## Analysis of Variance Table
##
## Model 1: AssignmentData$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG5YR +
##     USGG10YR + USGG30YR
## Model 2: AssignmentData$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG5YR +
##     USGG30YR
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1   8293 3.7037
## 2   8294 7.4635 -1   -3.7597 8418.3 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
drop1(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR)
```

```
## Single term deletions
##
## Model:
## AssignmentData$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG5YR +
##     USGG30YR
##          Df Sum of Sq     RSS    AIC
## <none>                   7.463 -58204
## USGG3M    1    37.121  44.584 -43371
## USGG6M    1    15.086  22.549 -49029
## USGG2YR   1    60.611  68.074 -39858
## USGG5YR   1   105.012 112.476 -35691
## USGG30YR  1   145.865 153.329 -33119
```

```
AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M <- lm(AssignmentData$Output1~.,data=As
summary(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ ., data = AssignmentData1[,
##     c(-2, -4, -6)])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.38697 -0.03049 -0.00084  0.03257  0.43021
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -14.812574   0.002846 -5204.6   <2e-16 ***
## USGG3M        0.607302   0.001326   458.0   <2e-16 ***
## USGG2YR       0.843754   0.003381   249.5   <2e-16 ***
## USGG5YR       0.744117   0.003977   187.1   <2e-16 ***
## USGG30YR      0.423994   0.001907   222.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05214 on 8295 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 5.862e+07 on 4 and 8295 DF,  p-value: < 2.2e-16
```

```
anova(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR,AssignmentDataRegressionComparison.noUSGG3
```

```
## Analysis of Variance Table
##
## Model 1: AssignmentData$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG5YR +
##     USGG30YR
## Model 2: AssignmentData$Output1 ~ USGG3M + USGG2YR + USGG5YR + USGG30YR
##   Res.Df     RSS Df Sum of Sq     F    Pr(>F)
## 1   8294  7.4635
## 2   8295 22.5492 -1   -15.086 16765 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
drop1(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M)
```

```
## Single term deletions
##
## Model:
## AssignmentData$Output1 ~ USGG3M + USGG2YR + USGG5YR + USGG30YR
##          Df Sum of Sq    RSS    AIC
## <none>                 22.55 -49029
## USGG3M    1    570.22 592.77 -21897
## USGG2YR   1    169.26 191.81 -31262
## USGG5YR   1     95.15 117.70 -35316
## USGG30YR  1    134.40 156.95 -32927
```

```
AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR  <- lm(AssignmentData$Output
summary(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ ., data = AssignmentData1[,
##     c(-2, -4, -5, -6)])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.55605 -0.05109  0.01731  0.08219  0.70527
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -15.055581   0.005785 -2602.5   <2e-16 ***
```

```
## USGG3M        0.496988   0.002713   183.2   <2e-16 ***
## USGG2YR       1.404153   0.003584   391.8   <2e-16 ***
## USGG30YR      0.741709   0.001981   374.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1191 on 8296 degrees of freedom
## Multiple R-squared:  0.9998, Adjusted R-squared:  0.9998
## F-statistic: 1.497e+07 on 3 and 8296 DF,  p-value: < 2.2e-16
```

```
anova(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M,AssignmentDataRegressionComparis
```

```
## Analysis of Variance Table
##
## Model 1: AssignmentData$Output1 ~ USGG3M + USGG2YR + USGG5YR + USGG30YR
## Model 2: AssignmentData$Output1 ~ USGG3M + USGG2YR + USGG30YR
##   Res.Df     RSS Df Sum of Sq     F    Pr(>F)
## 1   8295  22.549
## 2   8296 117.697 -1   -95.148 35001 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
drop1(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR)
```

```
## Single term deletions
##
## Model:
## AssignmentData$Output1 ~ USGG3M + USGG2YR + USGG30YR
##          Df Sum of Sq    RSS    AIC
## <none>                 117.7 -35316
## USGG3M    1     476.0  593.7 -21886
## USGG2YR   1    2177.7 2295.4 -10662
## USGG30YR  1    1988.2 2105.9 -11378
```

```
AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR.noUSGG3M  <- lm(AssignmentDa
summary(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR.noUSGG3M)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ ., data = AssignmentData1[,
##     c(-1, -2, -4, -5, -6)])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.65662 -0.17773  0.01775  0.17526  1.84678
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -14.590584   0.011675 -1249.8   <2e-16 ***
## USGG2YR       2.023026   0.002685   753.5   <2e-16 ***
## USGG30YR      0.522257   0.003544   147.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2675 on 8297 degrees of freedom
## Multiple R-squared:  0.9991, Adjusted R-squared:  0.9991
```

```
## F-statistic: 4.45e+06 on 2 and 8297 DF,  p-value: < 2.2e-16
```

```
anova(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR,AssignmentDataRegressi
```

```
## Analysis of Variance Table
##
## Model 1: AssignmentData$Output1 ~ USGG3M + USGG2YR + USGG30YR
## Model 2: AssignmentData$Output1 ~ USGG2YR + USGG30YR
##   Res.Df   RSS Df Sum of Sq     F    Pr(>F)
## 1   8296 117.7
## 2   8297 593.7 -1      -476 33552 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
drop1(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR.noUSGG3M)
```

```
## Single term deletions
##
## Model:
## AssignmentData$Output1 ~ USGG2YR + USGG30YR
##          Df Sum of Sq   RSS    AIC
## <none>                  594 -21886
## USGG2YR   1     40625 41219  13306
## USGG30YR  1      1554  2148 -11217
```

```
# Model with only USGG2YR
AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR.noUSGG3M.noUSGG30YR  <- lm(A
summary(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR.noUSGG3M.noUSGG30YR)
```

```
##
## Call:
## lm(formula = AssignmentData$Output1 ~ USGG2YR, data = AssignmentData1)
##
## 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
```

```
anova(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR.noUSGG3M,AssignmentData
```

```
## Analysis of Variance Table
##
## Model 1: AssignmentData$Output1 ~ USGG2YR + USGG30YR
## Model 2: AssignmentData$Output1 ~ USGG2YR
##   Res.Df    RSS Df Sum of Sq     F    Pr(>F)
## 1   8297  593.7
## 2   8298 2147.6 -1   -1553.9 21716 < 2.2e-16 ***
```

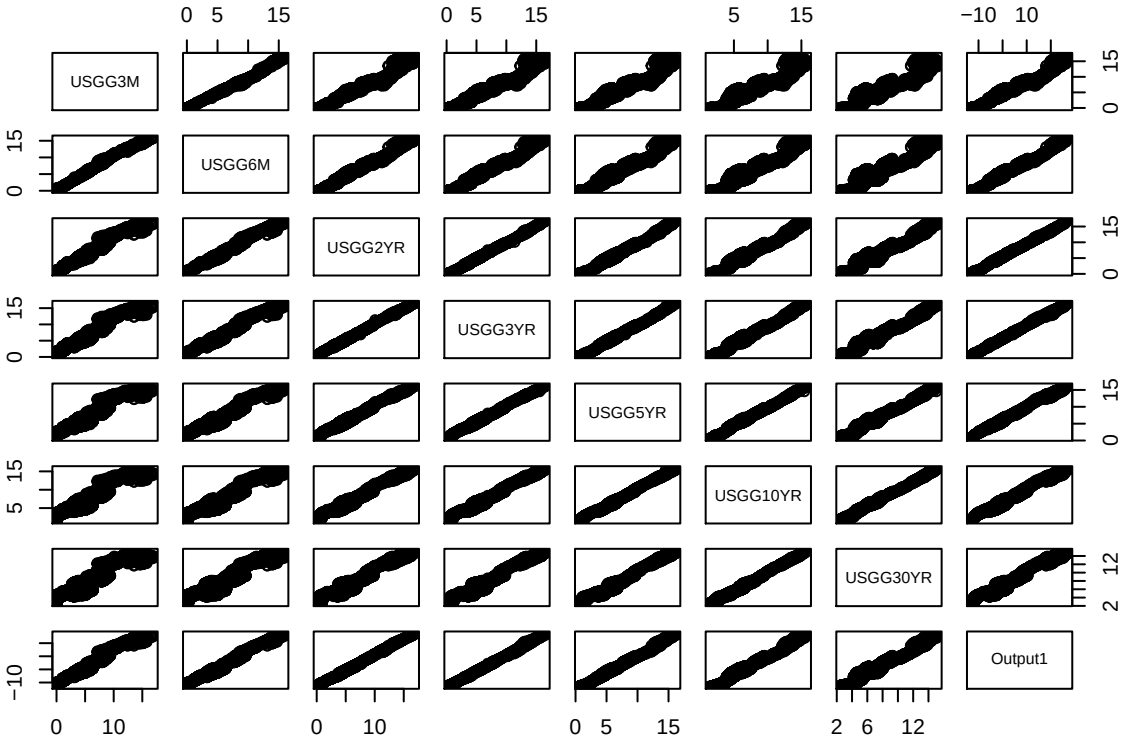
```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
drop1(AssignmentDataRegressionComparison.noUSGG3YR.noUSGG10YR.noUSGG6M.noUSGG5YR.noUSGG3M.noUSGG30YR)
```

```
## Single term deletions
##
## Model:
## AssignmentData$Output1 ~ USGG2YR
##         Df Sum of Sq    RSS    AIC
## <none>                2148 -11217
## USGG2YR  1    635252 637400  36033
```

Alternatively, trying for the best fit,

```
#Since full model is a perfect fit, perform pairs to check collinearity
data.ModelSelection <-  AssignmentData[1:8]
pairs(data.ModelSelection)
```



Pairs plot tells us that Each pair of the predictors and the output are positively and linearly correlated.

Using regsubsets: Best Subset Selection - To perform best subset selection, we fit a separate least squares regression for each possible combination of the p predictors. We can perform a best subset search using regsubsets (part of the leaps library), which identifies the best model for a given number of k predictors, where best is quantified using RSS. Here we fit up to a 7-variable model.

```
install.packages("leaps", repos = "http://cran.us.r-project.org")
```

```
##
## The downloaded binary packages are in
##  /var/folders/zr/f_5yx5010f33bqmyk26q08s40000gn/T//RtmpulEeN2/downloaded_packages
```

```
library(leaps)
best_subset <- regsubsets(Output1 ~ ., data.ModelSelection, nvmax = 7)
```

```
summary(best_subset)
```

```
## Subset selection object
## Call: regsubsets.formula(Output1 ~ ., data.ModelSelection, nvmax = 7)
## 7 Variables  (and intercept)
##          Forced in Forced out
## USGG3M       FALSE      FALSE
## USGG6M       FALSE      FALSE
## USGG2YR      FALSE      FALSE
## USGG3YR      FALSE      FALSE
## USGG5YR      FALSE      FALSE
## USGG10YR     FALSE      FALSE
## USGG30YR     FALSE      FALSE
## 1 subsets of each size up to 7
## Selection Algorithm: exhaustive
##          USGG3M USGG6M USGG2YR USGG3YR USGG5YR USGG10YR USGG30YR
## 1  ( 1 ) " "    " "    " "     "*"     " "     " "      " "
## 2  ( 1 ) "*"    " "    " "     " "     "*"     " "      " "
## 3  ( 1 ) "*"    " "    "*"     " "     " "     "*"      " "
## 4  ( 1 ) "*"    "*"    " "     "*"     " "     "*"      " "
## 5  ( 1 ) "*"    "*"    "*"     " "     "*"     " "      "*"
## 6  ( 1 ) "*"    "*"    "*"     " "     "*"     "*"      "*"
## 7  ( 1 ) "*"    "*"    "*"     "*"     "*"     "*"      "*"
```

```
dat <- AssignmentData[1:8]
bestLm <- lm(dat$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG5YR + USGG10YR  + USGG30YR ,data=dat)
summary(bestLm)
```

```
##
## Call:
## lm(formula = dat$Output1 ~ USGG3M + USGG6M + USGG2YR + USGG5YR +
##     USGG10YR + USGG30YR, data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36972 -0.01132 -0.00141  0.00905  0.30202
##
## Coefficients:
##               Estimate Std. Error   t value Pr(>|t|)
## (Intercept) -14.863604   0.001358 -10942.56   <2e-16 ***
## USGG3M        0.389687   0.001332    292.51   <2e-16 ***
## USGG6M        0.357731   0.001997    179.09   <2e-16 ***
## USGG2YR       0.685589   0.001806    379.63   <2e-16 ***
## USGG5YR       0.577177   0.002853    202.33   <2e-16 ***
## USGG10YR      0.347515   0.003788     91.75   <2e-16 ***
## USGG30YR      0.270634   0.002085    129.77   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02113 on 8293 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 2.379e+08 on 6 and 8293 DF,  p-value: < 2.2e-16
```

We're attempting to fit a model against data that provide a perfect fit. So i feel there is nothing to do and further, as it by itself best describes the data. Regardless of this, we are able to get a model that fits perfectly

and have only a few key predictive variables.

Explain your selection.

Based on the fact that drop1 analysis shows all models are equally good fit and based on Best Subset results, I chose the linear model with 6 Predictors USGG3M + USGG6M + USGG2YR + USGG5YR + USGG10YR + USGG30YR, Since A model "as simple as possible, but no simpler" is a good motto for interpretable models and also a model with lesser residual errors is the best suitable one. I went with the 6 predictor model. I did not choose the full model(7 predictor model) as the best model because its a perfect fit and in real world, no model could be a perfect.

Step 6.

```
# Performing rolling window analysis of the yields data.

Window.width<-20; Window.shift<-5
library(zoo)
```

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

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

```
# Calculating rolling mean values for each variable.
# Means
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
```

```
# Creating 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
```

```
# Taking 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
```

```
# Incerting 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]
```

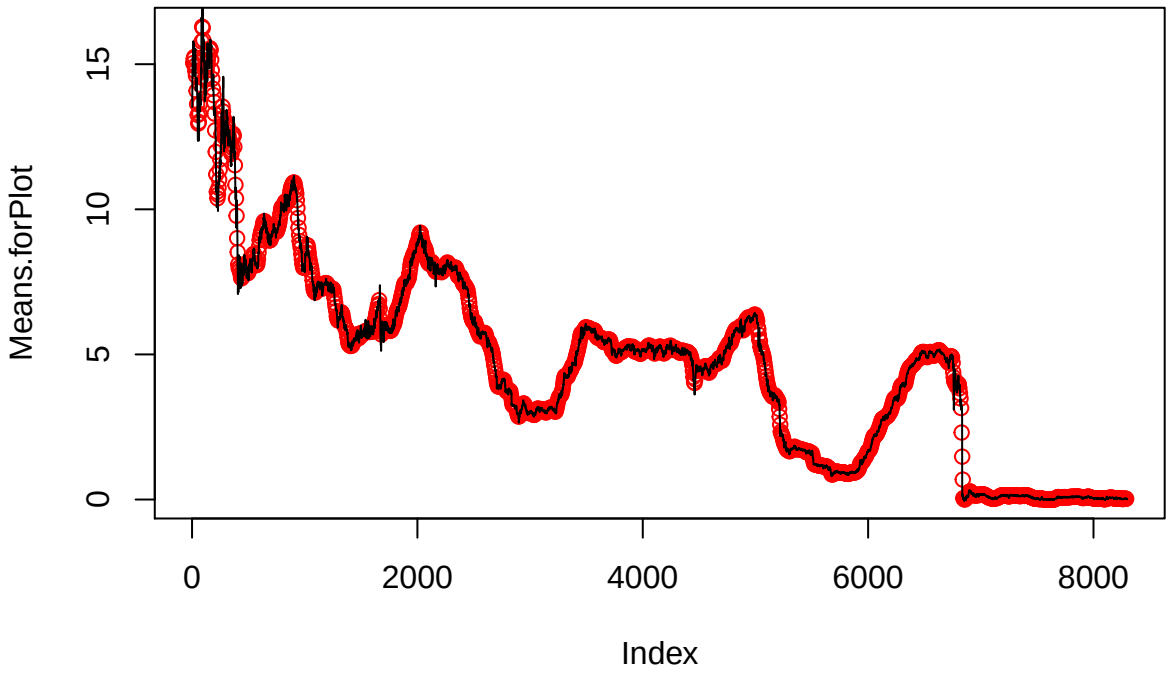
```
##  [1]      NA      NA      NA      NA      NA      NA      NA      NA
##  [9]      NA 15.0405      NA      NA      NA      NA 15.1865      NA
## [17]      NA      NA      NA 15.2480      NA      NA      NA      NA
## [25] 14.9345      NA      NA      NA      NA 14.7545      NA      NA
## [33] 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

```
# Assembling 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])
```




```
# Running rolling daily difference standard deviation of each variable
# Standard deviation
rolling.diff <- rollapply(AssignmentDataRegressionComparison, width = 2, by = 1, by.column = TRUE, diff
rownames(rolling.diff) <- rownames(AssignmentDataRegressionComparison[2:8300,])
rolling.sd<-rollapply(rolling.diff,width=Window.width,by=Window.shift,by.column=TRUE, sd)
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)
```

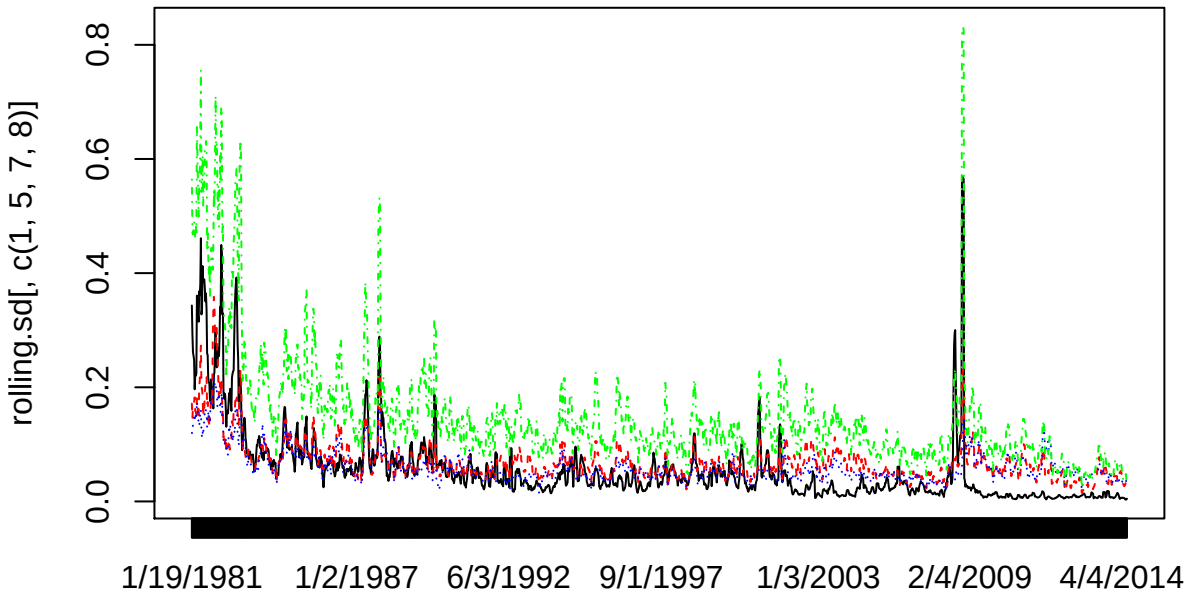
```
##      [,1]        [,2]        [,3]        [,4]        [,5]
## [1,] "1/6/1981"  "1/7/1981"  "1/8/1981"  "1/9/1981"  "1/12/1981"
## [2,] "1/13/1981" "1/14/1981" "1/15/1981" "1/16/1981" "1/19/1981"
## [3,] "1/20/1981" "1/21/1981" "1/22/1981" "1/23/1981" "1/26/1981"
## [4,] "1/27/1981" "1/28/1981" "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))
```



Showing periods of high volatility.

The plot says that Interest rates of short term bonds are more volatile than the long term bonds. 3M has highhest volatality and the volatality decreases with the 30YR being the least volatile.

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

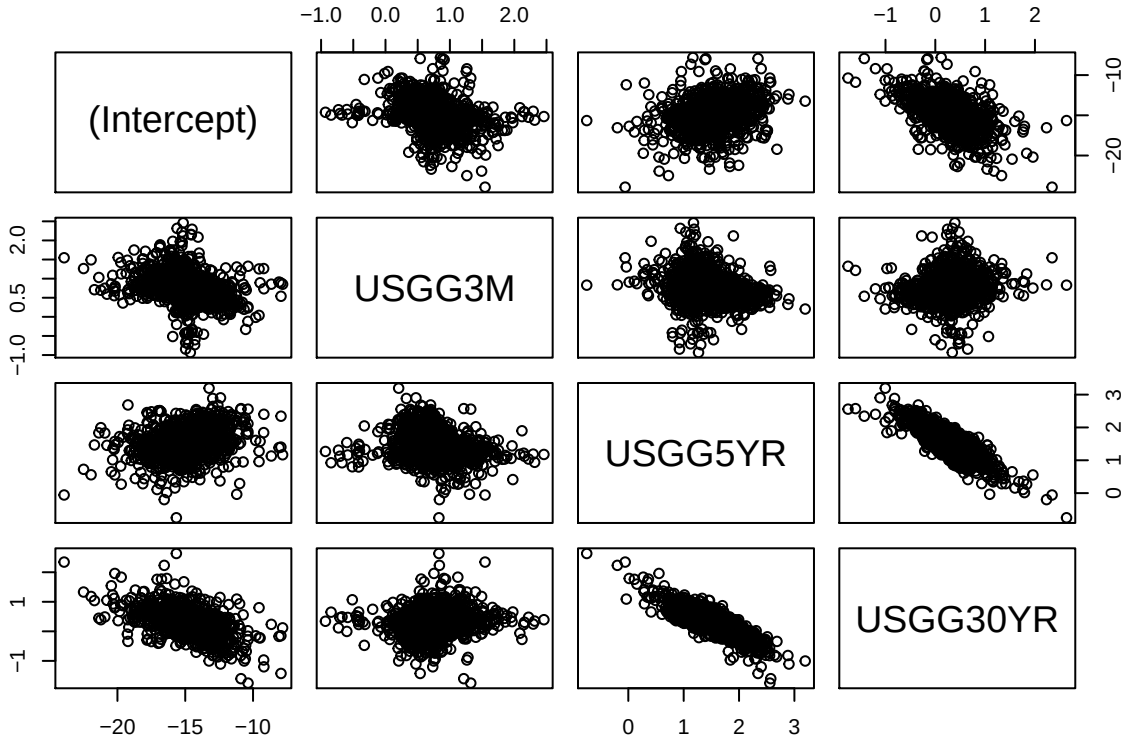
```
##  [1] "1/19/1981"  "3/25/1981"  "4/1/1981"   "4/8/1981"   "4/23/1981"
##  [6] "4/30/1981"  "5/7/1981"   "5/14/1981"  "5/21/1981"  "5/29/1981"
## [11] "6/5/1981"   "6/12/1981"  "6/19/1981"  "6/26/1981"  "7/6/1981"
## [16] "7/13/1981"  "7/20/1981"  "7/27/1981"  "10/28/1981" "11/5/1981"
## [21] "11/13/1981" "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"
```

```
# Fiting linear model to rolling window data using 3 months, 5 years and 30 years variables as predicto

# 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
```

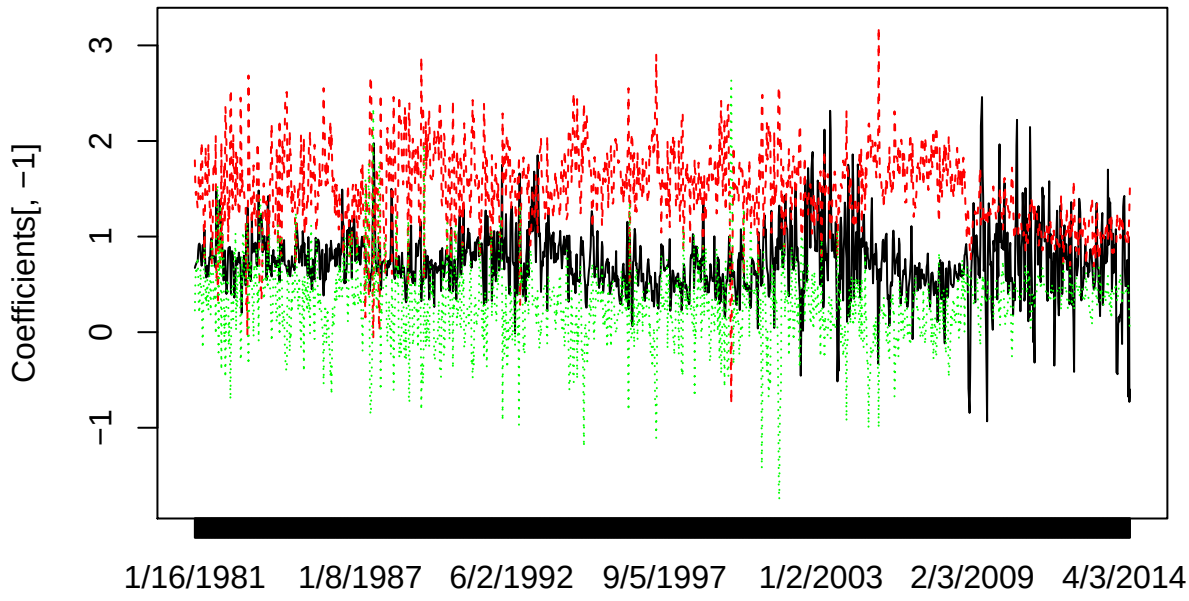
```r
# Pairs plot of Coefficients
pairs(Coefficients)
```



Interpreting the pairs plot.

Each of these pairs of Coefficients are negatively correlated except 3M & 30YR yields. When 5Yr yield effect decreases, the combined effect of 3M & 30YR yields increases.

```r
# 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]
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"
```

Picture of coefficients Vs picture of pairs?

Yes, the pictures of coefficients shows the 3M & 30YR rise and fall together.

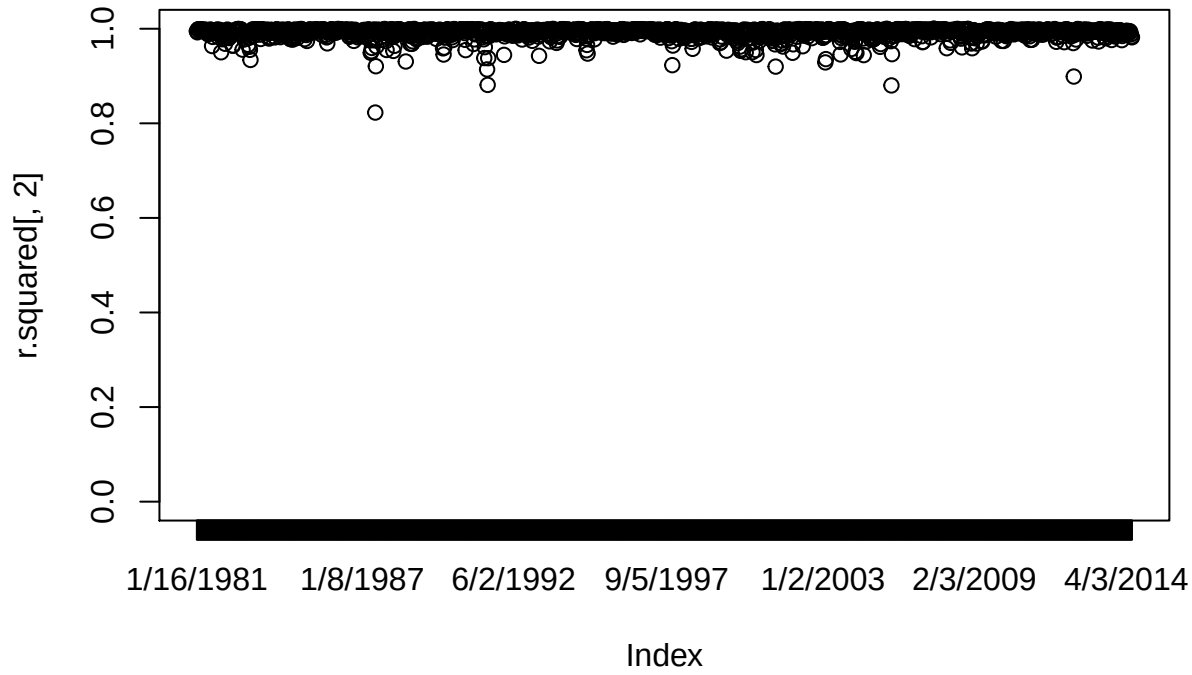
This is what we interpreted in the pairwise plot aswell. And our assumptions looks to match.

```
# 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 R^2 ?

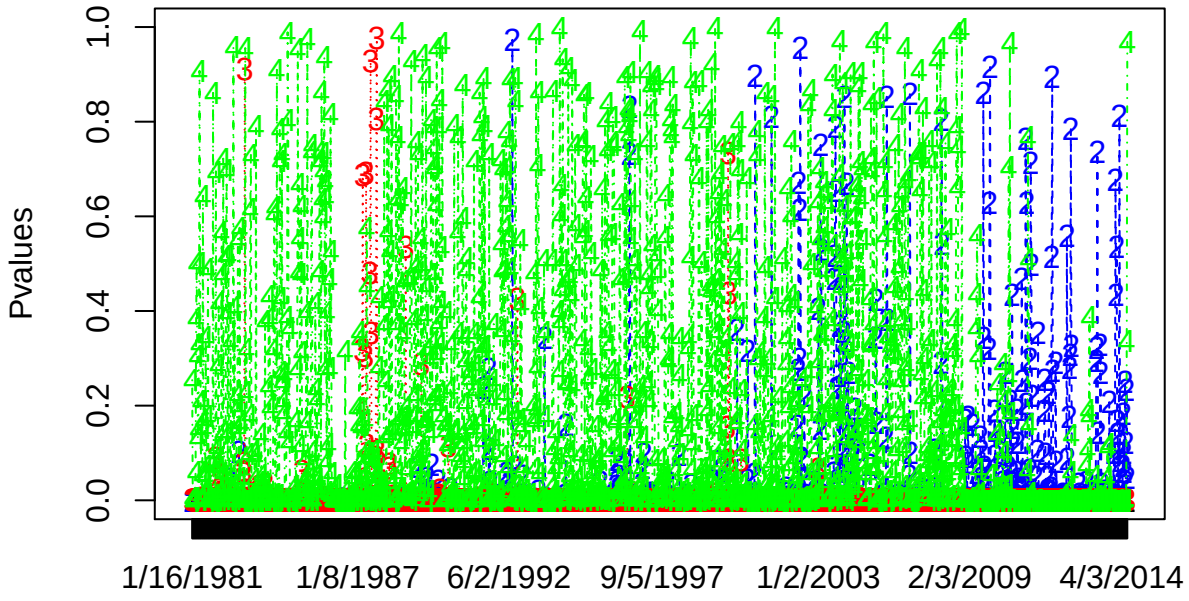
I could see that in the above plot, In every 20 day period we create the relationships for 3 of the 7 rates in output1, and usually it turns out to be very close to each other. But in some 20 day period, it turns out that the remaining 4 rates move in ways that the 3 rates(3YR, 2YR, 6M) we have don't support. That is may be indicated by the spike, when index was 2/5/1986, the r squared is around 0.8, indicating that something happened at this date, so the rates were not a good predictor of output1. This could be a cause of decrease of R squared. In other terms I think, If output 1 is even weights of all the 7 rates, then there must be some rates where the 6M rates and the 3YR rates stay levelled, but the 2YR rates is spiked. This would have created disturbance in the output, which would have shown up in the plot(point referring to is when index was 2/5/1986, the r squared is around 0.8).

```
# 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)
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"
```

Interpreting the plot.

As we can see, p values for these predictors are mostly very small, meaning that we can reject the null hypothesis. However, we also observe, the blue values for USGG3M predictor became more frequent as time progresses, meaning during that time the predictor was not explaining the output as the p levels were higher. Interestingly, USGG30YR predictor seems to be insignificant throughout most of the time, meaning that even if R squared is high for that predictor, its p-values ovwr time shows that its actually insignificant at explaining the output whereas the USGG10YR predictor is mostly significant all the time.

Step 7.

```
# Performing 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
```

```
# Selecting 3 variables. Exploring dimensionality and correlation
# Explore the dimensionality of the set of 3M, 2Y and 5Y yields.
AssignmentData.3M_2Y_5Y<-AssignmentData[,c(1,3,5)]
#pairs(AssignmentData.3M_2Y_5Y)
```

```
#system("defaults write org.R-project.R force.LANG en_US.UTF-8")
#install.packages("rgl", dependencies = TRUE)
#install.packages("rgl",repos = "http://cran.us.r-project.org")
#install.packages("rgl")
library("rgl")
rgl.points(AssignmentData.3M_2Y_5Y)
```

```
# Analyzing the covariance matrix of the data. Comparing results of manual calculation and cov().
k <- ncol(AssignmentData) #number of variables
n <- nrow(AssignmentData) #number of subjects

#Creating means for each column
M_mean <- matrix(data=1, nrow=n) %*% cbind(mean(AssignmentData[,1]),mean(AssignmentData[,2]),mean(Assig
                                           mean(AssignmentData[,5]),mean(AssignmentData[,6]),mean(Assig

#Creating a difference matrix
D <- AssignmentData - M_mean

#Creating the covariance matrix
(Manual.Covariance.Matrix <- (C <- (n-1)^-1 * t(D) %*% D))
```

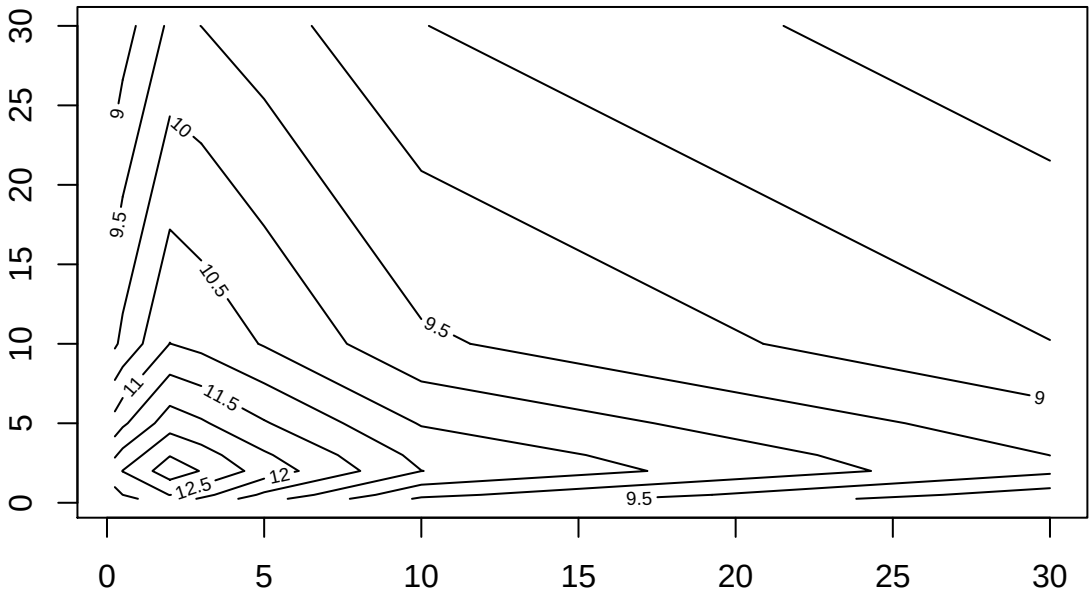
```
##             USGG3M    USGG6M   USGG2YR   USGG3YR   USGG5YR  USGG10YR
## USGG3M   11.760393 11.855287 12.303031 11.942035 11.188856  9.924865
## USGG6M   11.855287 12.000510 12.512434 12.158422 11.406959 10.128890
## USGG2YR  12.303031 12.512434 13.284203 12.977542 12.279514 11.005377
## USGG3YR  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
```

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

```
##             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
```

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




```
# Performing the PCA by manually.
rownames(AssignmentData) <- c()
Y <- AssignmentData
YMeans <- mean(Y)
# Creating Centered Matrix
Y0 <- Y - YMeans
V <-  cov(Y0)
# Eigenvalue decomposition of V
```

```r
Eigen.Decomposition <- eigen(V)
# Eigenvectors
L <- Eigen.Decomposition$vectors
# Loadings
Loadings <- L[,1:3]
# Factor scores
F <- Y0 %*% L
Factors <-  F[,1:3]
Loadings[,1:3]
```

```
##            [,1]        [,2]       [,3]
## [1,] -0.3839609 -0.50744508  0.5298222
## [2,] -0.3901870 -0.43946144  0.1114737
## [3,] -0.4151851 -0.11112721 -0.4187873
## [4,] -0.4063541  0.01696988 -0.4476561
## [5,] -0.3860610  0.23140317 -0.2462364
## [6,] -0.3477544  0.43245979  0.1500903
## [7,] -0.3047124  0.54421228  0.4979195
```

```r
# Define the model
(YMeans + F %*%t(L))[1:5,]
```

```
##       [,1]  [,2]   [,3]  [,4]   [,5]   [,6]   [,7]
## [1,] 13.52 13.09 12.289 12.28 12.294 12.152 11.672
## [2,] 13.58 13.16 12.429 12.31 12.214 12.112 11.672
## [3,] 14.50 13.90 12.929 12.78 12.614 12.382 11.892
## [4,] 14.76 14.00 13.099 12.95 12.684 12.352 11.912
## [5,] 15.20 14.30 13.539 13.28 12.884 12.572 12.132
```

```r
Y[1:5,]
```

```
##      USGG3M USGG6M USGG2YR USGG3YR USGG5YR USGG10YR USGG30YR
## [1,]  13.52  13.09  12.289   12.28  12.294   12.152   11.672
## [2,]  13.58  13.16  12.429   12.31  12.214   12.112   11.672
## [3,]  14.50  13.90  12.929   12.78  12.614   12.382   11.892
## [4,]  14.76  14.00  13.099   12.95  12.684   12.352   11.912
## [5,]  15.20  14.30  13.539   13.28  12.884   12.572   12.132
```

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

```r
head(F)
```

```
##           [,1]       [,2]     [,3]        [,4]        [,5]         [,6]
## [1,] -17.88097 -0.2717588 1.578720 -0.17666381 -0.25268829  0.010201771
## [2,] -17.95685 -0.3838272 1.559948 -0.06656505 -0.18373075 -0.025687141
## [3,] -19.31276 -0.8944138 1.761655 -0.13828525 -0.16870337  0.004473080
## [4,] -19.61396 -1.0721937 1.751480 -0.16711610 -0.03989555  0.045571482
## [5,] -20.43749 -1.2094554 1.779366 -0.12224720  0.10284894 -0.004099577
## [6,] -20.00763 -1.2730503 2.084079 -0.19055825  0.01056170  0.026515437
##             [,7]
## [1,] -0.08563796
## [2,] -0.07272276
## [3,] -0.07799051
```

```
## [4,] -0.08573213
## [5,] -0.07089560
## [6,] -0.04836618
```

```
L
```

```
##            [,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
```

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

```
Eigen.Decomposition$values
```

```
## [1] 76.804437933  1.551521258  0.122380498  0.014155405  0.008321381
## [6]  0.002248917  0.001555835
```

```
Eigen.Decomposition$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
```

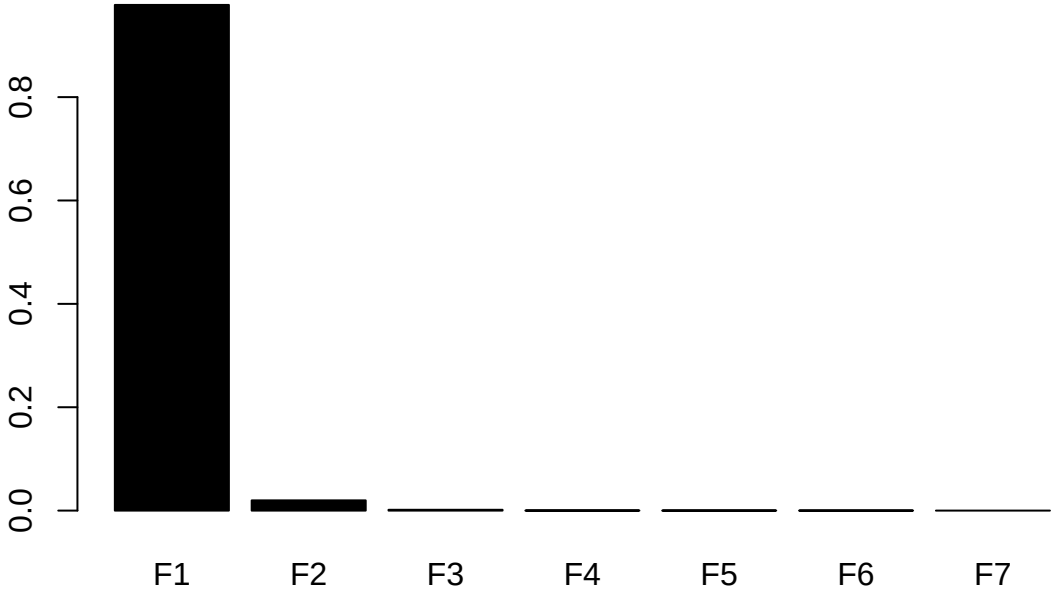
```
head(Y0)
```

```
##        USGG3M   USGG6M  USGG2YR  USGG3YR  USGG5YR USGG10YR USGG30YR
## [1,] 7.811008 7.381008 6.580008 6.571008 6.585008 6.443008 5.963008
## [2,] 7.871008 7.451008 6.720008 6.601008 6.505008 6.403008 5.963008
```

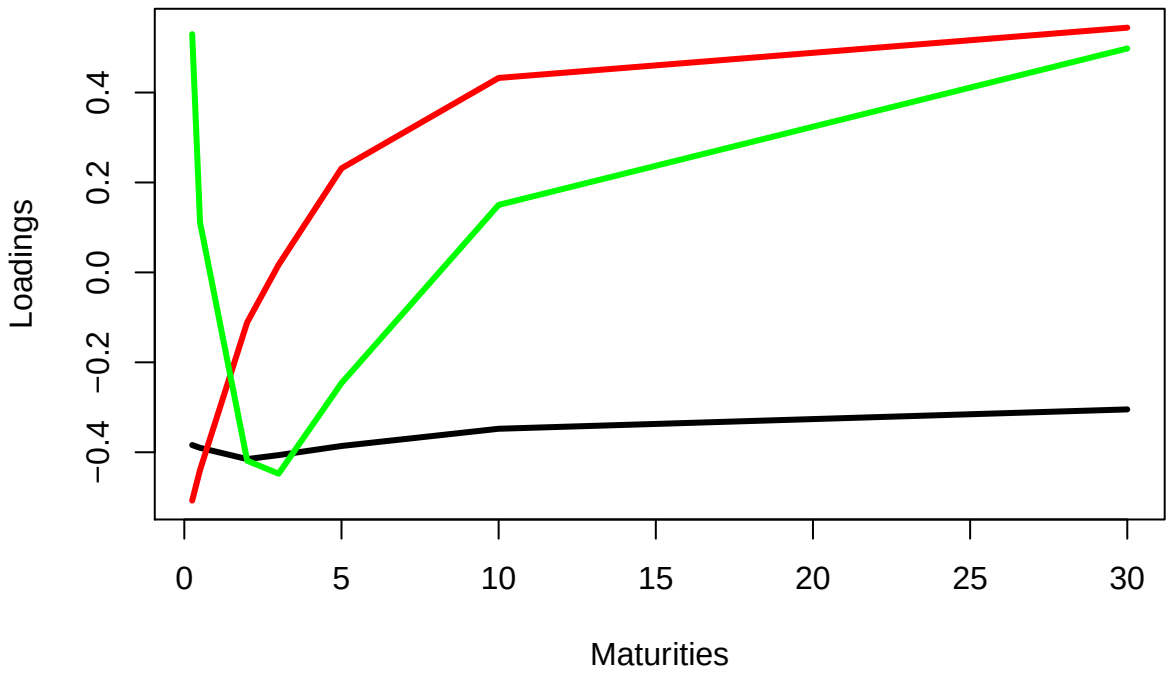
```
## [3,] 8.791008 8.191008 7.220008 7.071008 6.905008 6.673008 6.183008
## [4,] 9.051008 8.291008 7.390008 7.241008 6.975008 6.643008 6.203008
## [5,] 9.491008 8.591008 7.830008 7.571008 7.175008 6.863008 6.423008
## [6,] 9.511008 8.521008 7.470008 7.231008 7.005008 6.743008 6.373008
```

```
Loadings <- L[,1:3]
Factors <-  F[,1:3]
```

```
# Importance of factors.
barplot(Eigen.Decomposition$values/sum(Eigen.Decomposition$values),width=2,col = "black",
        names.arg=c("F1","F2","F3","F4","F5","F6","F7"))
```



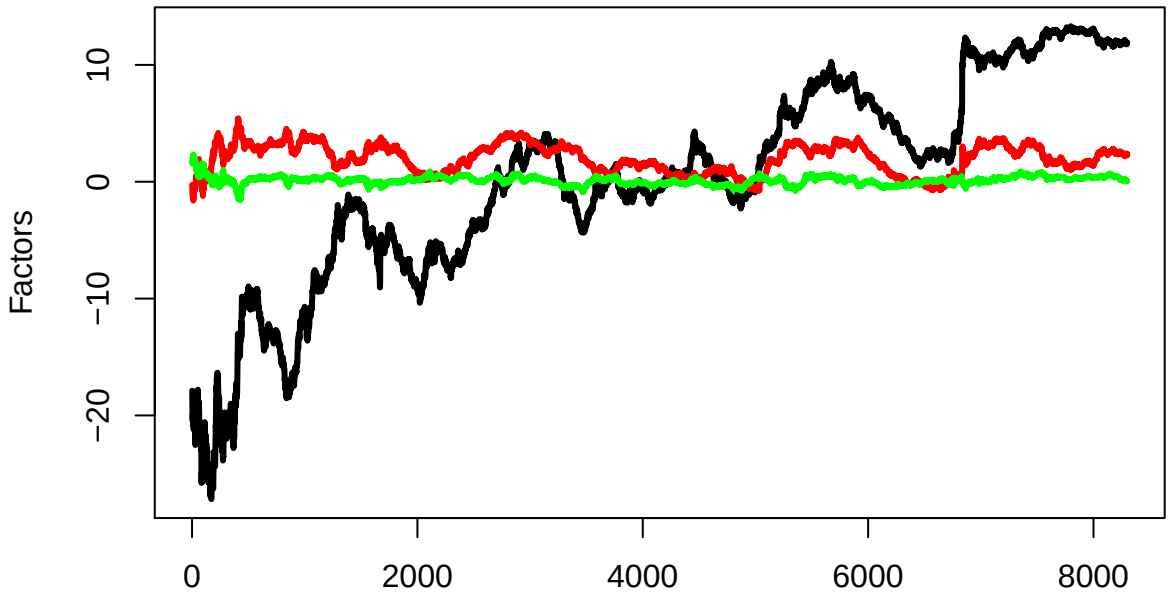

```
matplot(Maturities,Loadings,type="l",lty=1,col=c("black","red","green"),lwd=3)
```



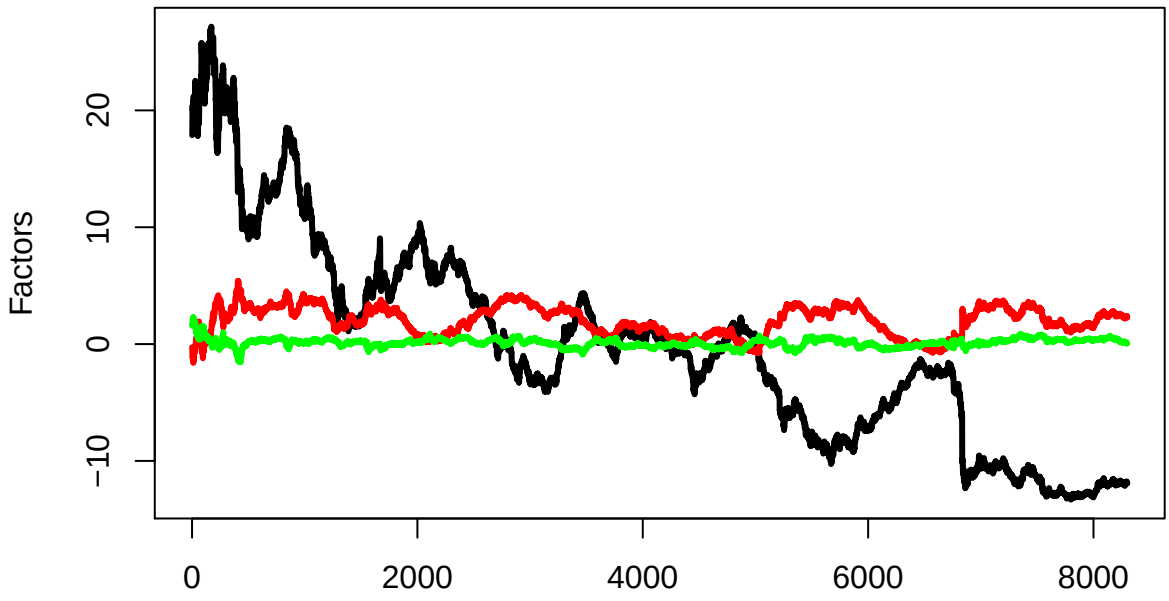
Interpreting the factors by looking at the shapes of the loadings.

1. All of the weights of first factor are below zero, and it almost seems to be equally weighted around -0.4 and -0.3
2. The second factor puts negative weightage on the nearest rate yields like 3M,6M and zero for say 2YR and positive for all other long term rates.
3. The third factor takes positive rates for the nearest term maturities like 3M and negative rates for the next few maturities and positive rates again. This looks like a butterfly curvature of the yield curve.
4. When there is movements, all the rates rise or fall together.
5. New term rates fall whereas long term rates are high or vice versa.
6. Looks like a butterfly where the mid term(middle yields) dips relative to the either hand(short term and long term maturities).

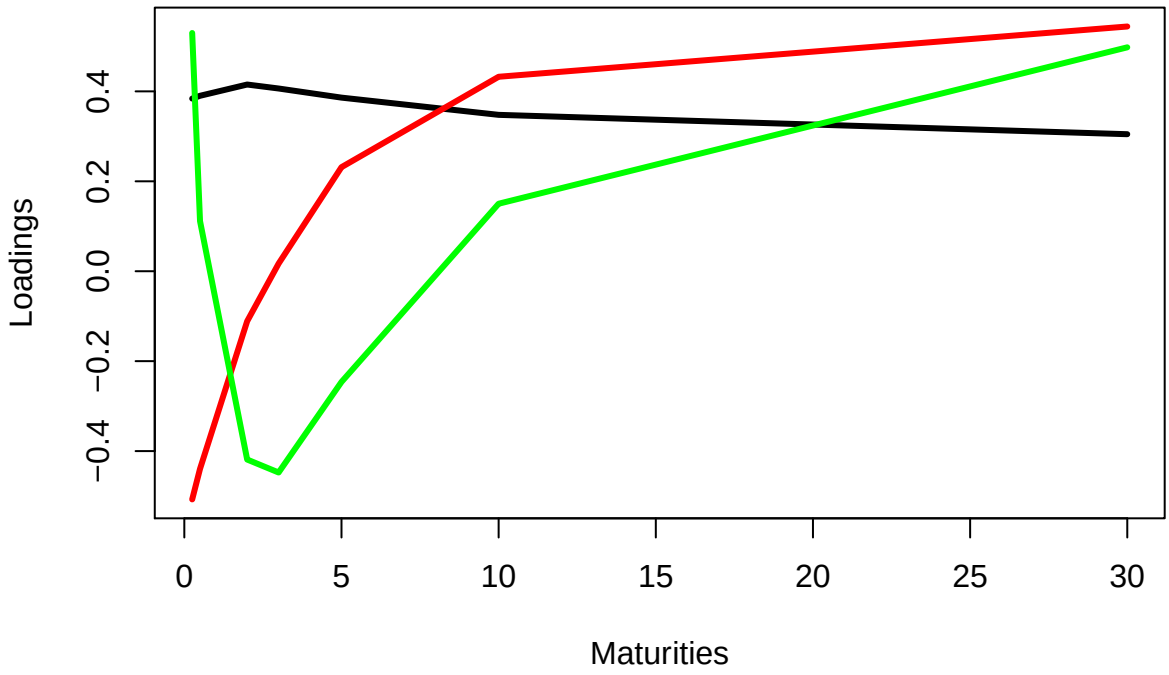
```
# Calculating and ploting 3 selected factors
matplot(Factors,type="l",col=c("black","red","green"),lty=1,lwd=3)
```



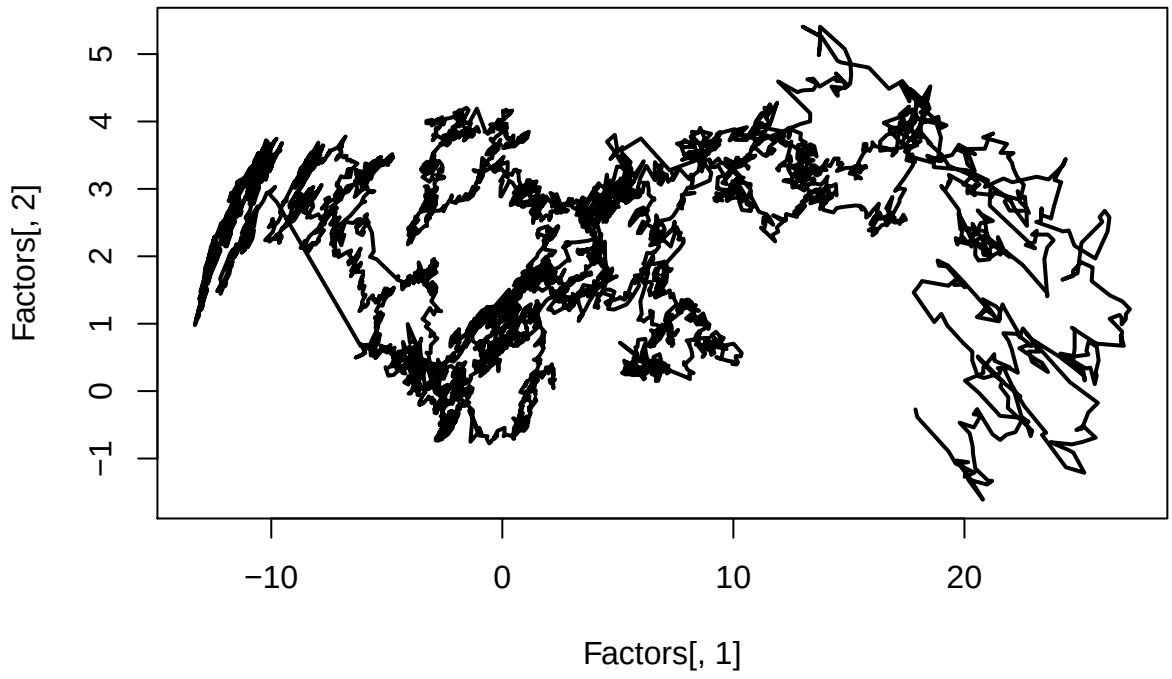

```
# Changing the signs of the first factor and the corresponding factor loading.
Loadings[,1]<--Loadings[,1]
Factors[,1]<--Factors[,1]
matplot(Factors,type="l",col=c("black","red","green"),lty=1,lwd=3)
```




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




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



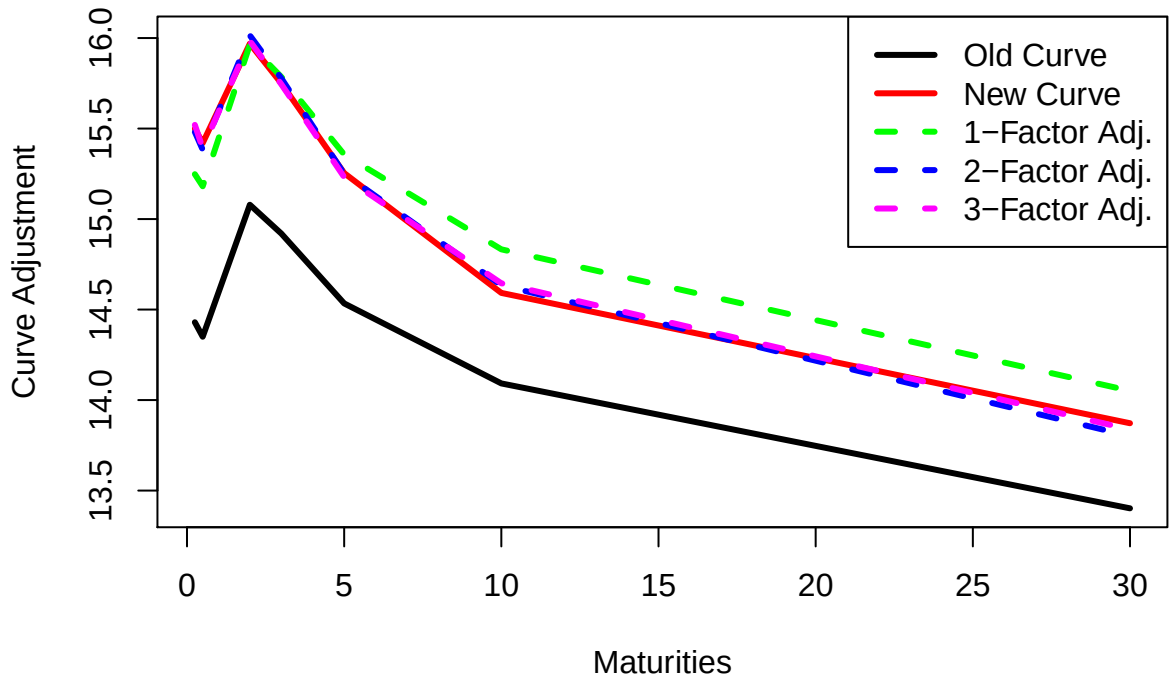
Drawing conclusions from the plot of the first two factors above.

–Factor 1 is almost the unweighted average of all the 7 yeilds. +ve vlaue of factor 1 = all yeilds are high.

1. When we look at it as dots, we could infer that few lines are straight in the second part of the graph indicating the day to day movement is large and it also looks more deliberate and few are squirrelled in the graph may be indicating the day to day movement is clumsy and looks like it is more uncertain.
2. As volatility is the measure of standard deviation of daily differences and here we could see that the daily differences are almost the same in the first half of the data, we can conclude that there is no volatility in the first part of the data even though there is movement. whereas the second part has more volatility.
3. One factor is following the other. Factor 2 fell, factor 1 fell, factor 2 rose and factor 1 rose. Factor1 is following factor2. This is only on one direction. We can further extend our analysis on what is the direction on these circles.
4. I could see some straight lines between one point and other point, but I couldnt infer much. MAy be they were stead and more deliberate during the particular time period.

```
# Analyzing 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
```




```
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
```

Explaining how shapes of the loadings affect the adjustnents using only factor 1, factors 1 and 2, and all 3 factors.

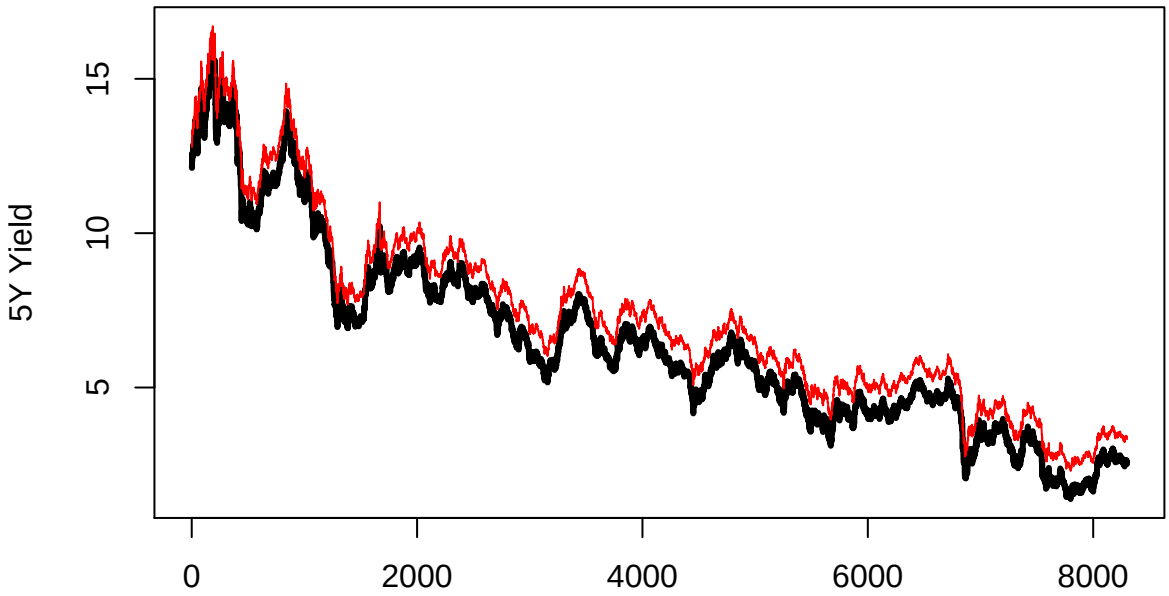
On one day the shortest term maturities had an interest rate of 14.5 whereas the longer term maturities was around 13.5. On another day, everything seemed to be crashed and the shortest term maturities had an interest rate of 15.5 whereas the longer term maturities was around 14. We can think about this difference as the added effect of shifting according to the first factor, second factor and third factor.It also, seems like by dropping the old curve below the new curve, we have overcorrected the short rates and we have not corrected for the longest rates. We try to fit the new curve closer and closer to the factors and that is the red line in the plot. The solid data are actually the real data from day1 to some subsequent day. The dashed lines is when we take the actual day 1 curve and we shift it only along the dimensions of the first factor which is the green and then from the subsequent corrections by the subsequent factors. Hence, its basically adjusting for whatever each factor can't explain.

```
# Goodness of fit for the example of 10Y yield.
# Glancing at how close is the approximation for each maturity?
# 5Y
cbind(Maturities,Loadings)
```

```
##      Maturities
```

```
## [1,]       0.25 0.3839609 -0.50744508  0.5298222
## [2,]       0.50 0.3901870 -0.43946144  0.1114737
## [3,]       2.00 0.4151851 -0.11112721 -0.4187873
## [4,]       3.00 0.4063541  0.01696988 -0.4476561
## [5,]       5.00 0.3860610  0.23140317 -0.2462364
## [6,]      10.00 0.3477544  0.43245979  0.1500903
## [7,]      30.00 0.3047124  0.54421228  0.4979195
```

```
Means <- colMeans(AssignmentData)
Model.10Y<-Means[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
```




```
# Repeating the PCA using princomp.

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

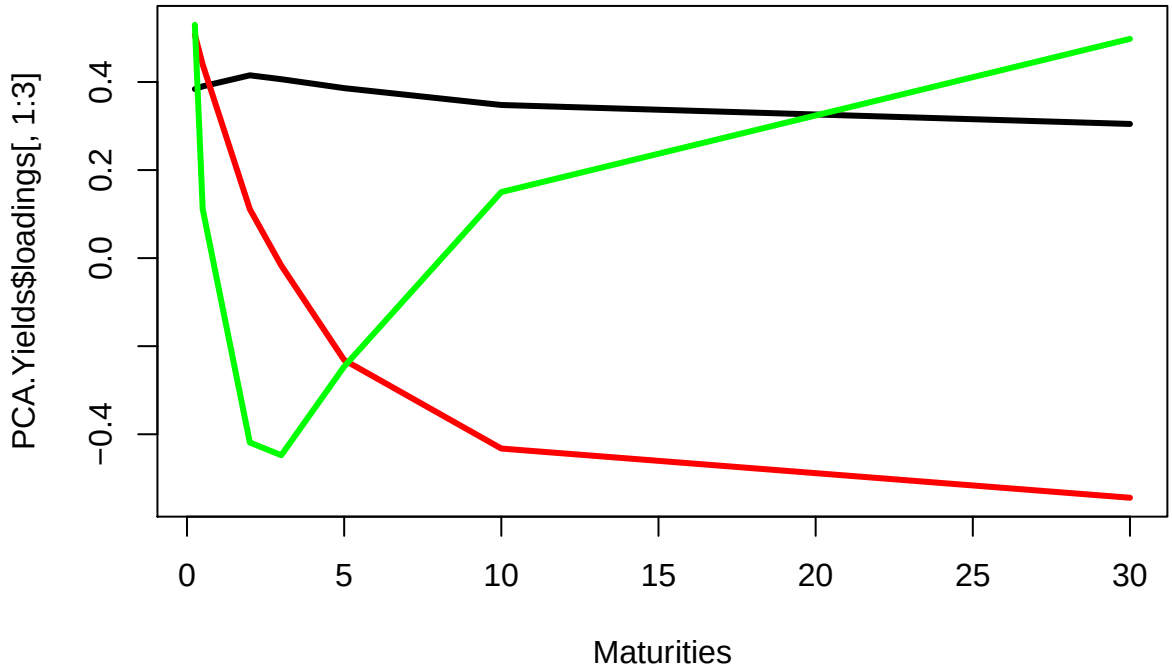
```
## [1] "sdev"     "loadings" "center"   "scale"    "n.obs"    "scores"
## [7] "call"
```

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

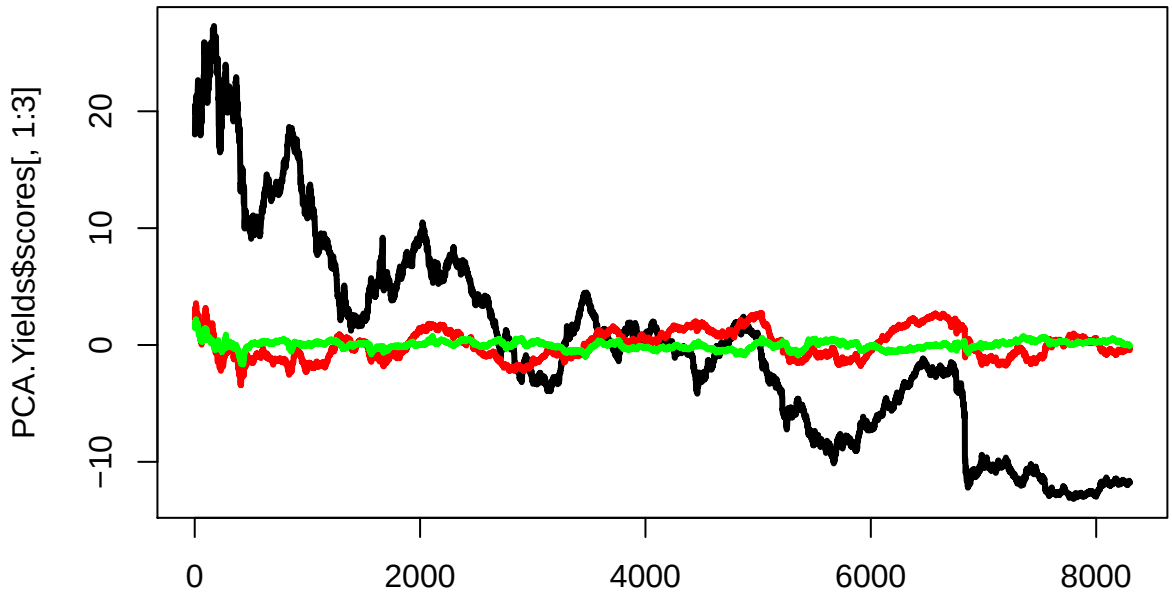
```
##             Comp.1      Comp.2     Comp.3 Maturities
## USGG3M   0.3839609  0.50744508  0.5298222       0.25 -0.3839609
## USGG6M   0.3901870  0.43946144  0.1114737       0.50 -0.3901870
## USGG2YR  0.4151851  0.11112721 -0.4187873       2.00 -0.4151851
## USGG3YR  0.4063541 -0.01696988 -0.4476561       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
```

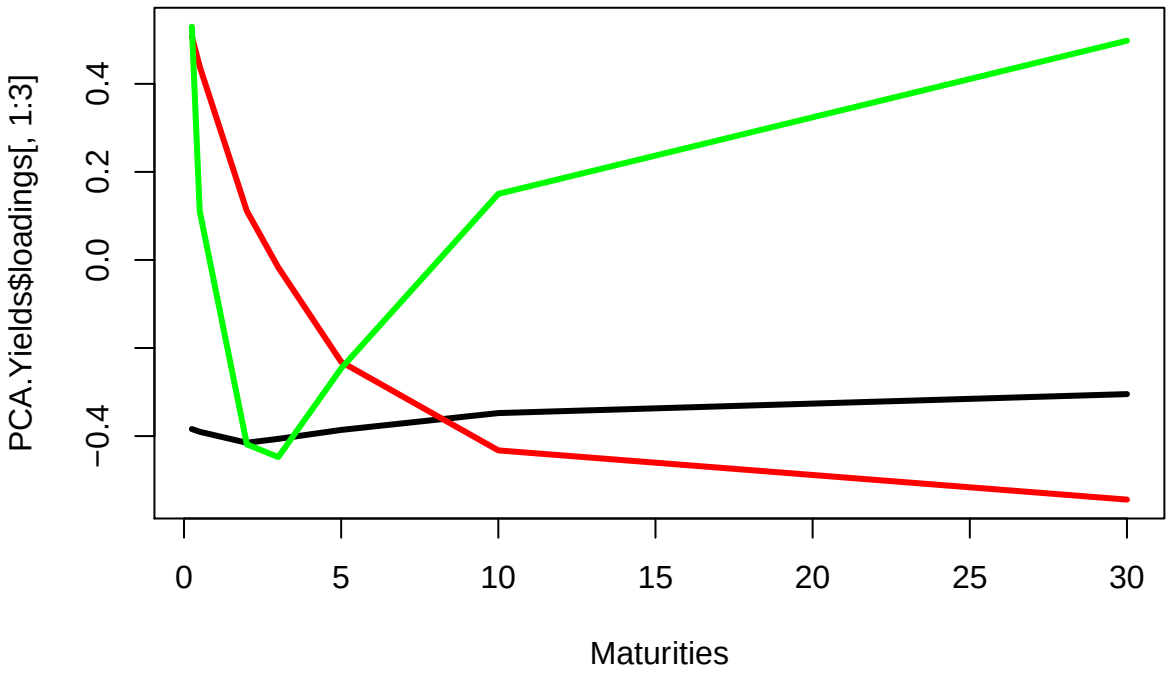
```
matplot(Maturities,PCA.Yields$loadings[,1:3],type="l",col=c("black","red","green"),lty=1,lwd=3)
```



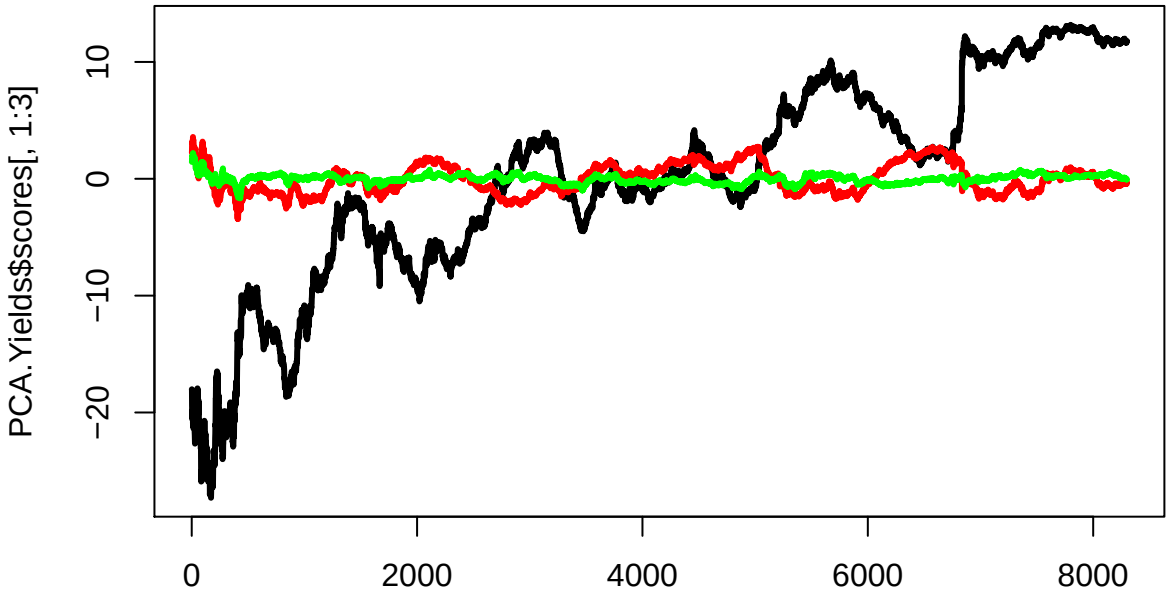

```
matplot(PCA.Yields$scores[,1:3],type="l",col=c("black","red","green"),lwd=3,lty=1)
```




```
# Changing the signs of the first factor and factor loading again.
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)
```



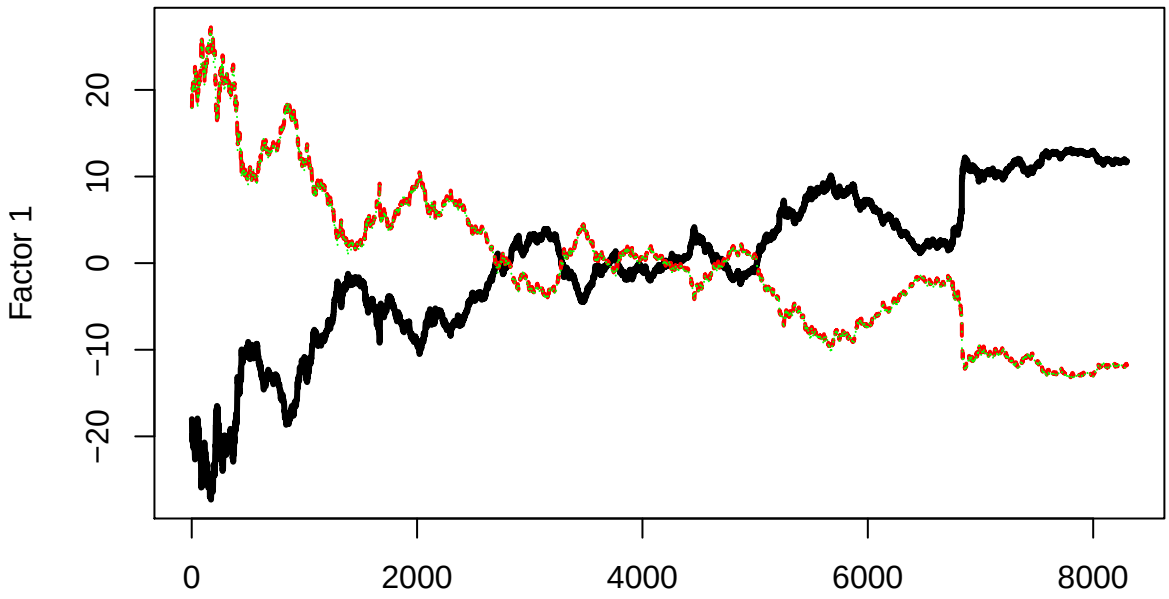

```
matplot(PCA.Yields$scores[,1:3],type="l",col=c("black","red","green"),lwd=3,lty=1)
```



Output:

Output 1 is the first factor of PCA, which is the linear combination of original variables(all the 7 rates), which explains the most variance. i.e, If we are given the tomorrows 3M, 2YR, 3YR, 5YR, 10YR, 30YR, then we can tell the 6M rates with 99% confidence using factor 1.

```
# Uncovering the mystery of the Output in column 8.

matplot(cbind(PCA.Yields$scores[,1],AssignmentData.Output,Factors[,1]),type="l",col=c("black","red","gr
```




```
#Comparing the regression coefficients from Step 2 and Step 3 with factor loadings.

#First, looking 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
```

```
# Checking if the same is true in the opposite direction: is there a correspondence between the coeffic
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
```

```
# Recovering the loading of the first factor by doing regression, using 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
## -0.3839609 -0.3901870 -0.4151851 -0.4063541 -0.3860610 -0.3477544
##   USGG30YR
## -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
## -0.3839609 -0.3901870 -0.4151851 -0.4063541 -0.3860610 -0.3477544
##   USGG30YR
## -0.3047124
```