```
#============================================#
# Michael Hyeong-Seok Beven                  #
# University of Chicago                      #
# Financial Mathematics                      #
# Homework Assignment on Statistical Models #
# 20160510                                   #
#============================================#
```

```
#########
# setup #
#########

rm(list=ls())
library(TTR) # rolling calculations
library(PerformanceAnalytics) # time series data
```

```
####################
# prepare the data #
####################
setwd('/Users/michaelbeven/Documents/06_School/2016 Spring/FINM_2016_SPRING/FINM 3360
1/Week 4')
df <- as.data.frame(read.csv('assignments-Fixed Income Derivatives-Assignment Lecture
4- Statistical Model-StatisticalModelData2014.csv'))
rownames(df) <- as.Date(df$Date,format='%m/%d/%Y')
df <- df[colnames(df)[2:8]]
maturities <- c(0.25,0.5,2,3,5,10,30) #maturities of instruments
print(head(df))
```

```
##             USGG3M USGG6M USGG2YR USGG3YR USGG5YR USGG10YR USGG30YR
## 1981-01-05  13.52  13.09  12.289   12.28  12.294   12.152   11.672
## 1981-01-06  13.58  13.16  12.429   12.31  12.214   12.112   11.672
## 1981-01-07  14.50  13.90  12.929   12.78  12.614   12.382   11.892
## 1981-01-08  14.76  14.00  13.099   12.95  12.684   12.352   11.912
## 1981-01-09  15.20  14.30  13.539   13.28  12.884   12.572   12.132
## 1981-01-12  15.22  14.23  13.179   12.94  12.714   12.452   12.082
```

```
##########################################
# estimate the 3-factor model using PCA #
##########################################

# define factor and factor loadings
df.cov <- cov(df) # covariance matrix
eigenvectors <- eigen(df.cov)$vectors
eigenvalues <- eigen(df.cov)$values
PC <- as.matrix(df) %*% eigenvectors # principal components
print(df.cov)
```

```
##             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
```

```
print(eigenvectors)
```

```
##             [,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
```

```
print(eigenvalues)
```

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

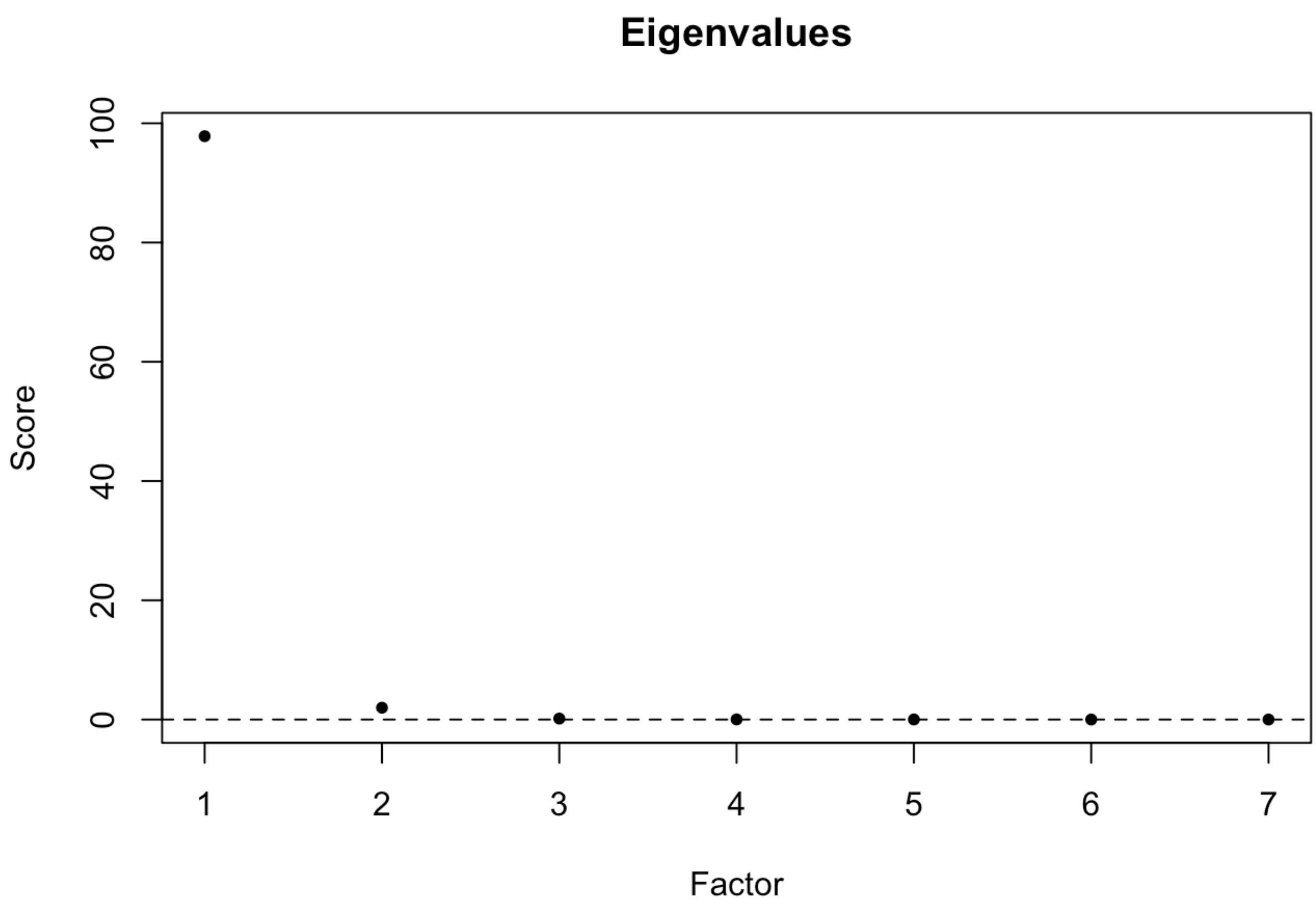
```
print(head(PC))
```

```
##                 [,1]        [,2]     [,3]        [,4]        [,5]
## 1981-01-05 -32.91968 -0.68170784 2.587076 0.023196089 -0.32888444
## 1981-01-06 -32.99556 -0.56963948 2.568304 0.133294846 -0.25992690
## 1981-01-07 -34.35147 -0.05905289 2.770011 0.061574643 -0.24489952
## 1981-01-08 -34.65267  0.11872704 2.759836 0.032743798 -0.11609170
## 1981-01-09 -35.47620  0.25598873 2.787721 0.077612690  0.02665279
## 1981-01-12 -35.04634  0.31958362 3.092435 0.009301648 -0.06563445
##                  [,6]       [,7]
## 1981-01-05 0.11017494 -0.1461522
## 1981-01-06 0.07428603 -0.1332370
## 1981-01-07 0.10444625 -0.1385048
## 1981-01-08 0.14554465 -0.1462464
## 1981-01-09 0.09587359 -0.1314099
## 1981-01-12 0.12648861 -0.1088805
```

```
# calculate relative importance of factors
print(round(eigenvalues/sum(eigenvalues)*100,2))
```

```
## [1] 97.83  1.98  0.16  0.02  0.01  0.00  0.00
```

```
plot(round(eigenvalues/sum(eigenvalues)*100,2),pch=20,main='Eigenvalues',ylab='Score'
,xlab='Factor')
abline(a=0,b=0,lty=2)
```

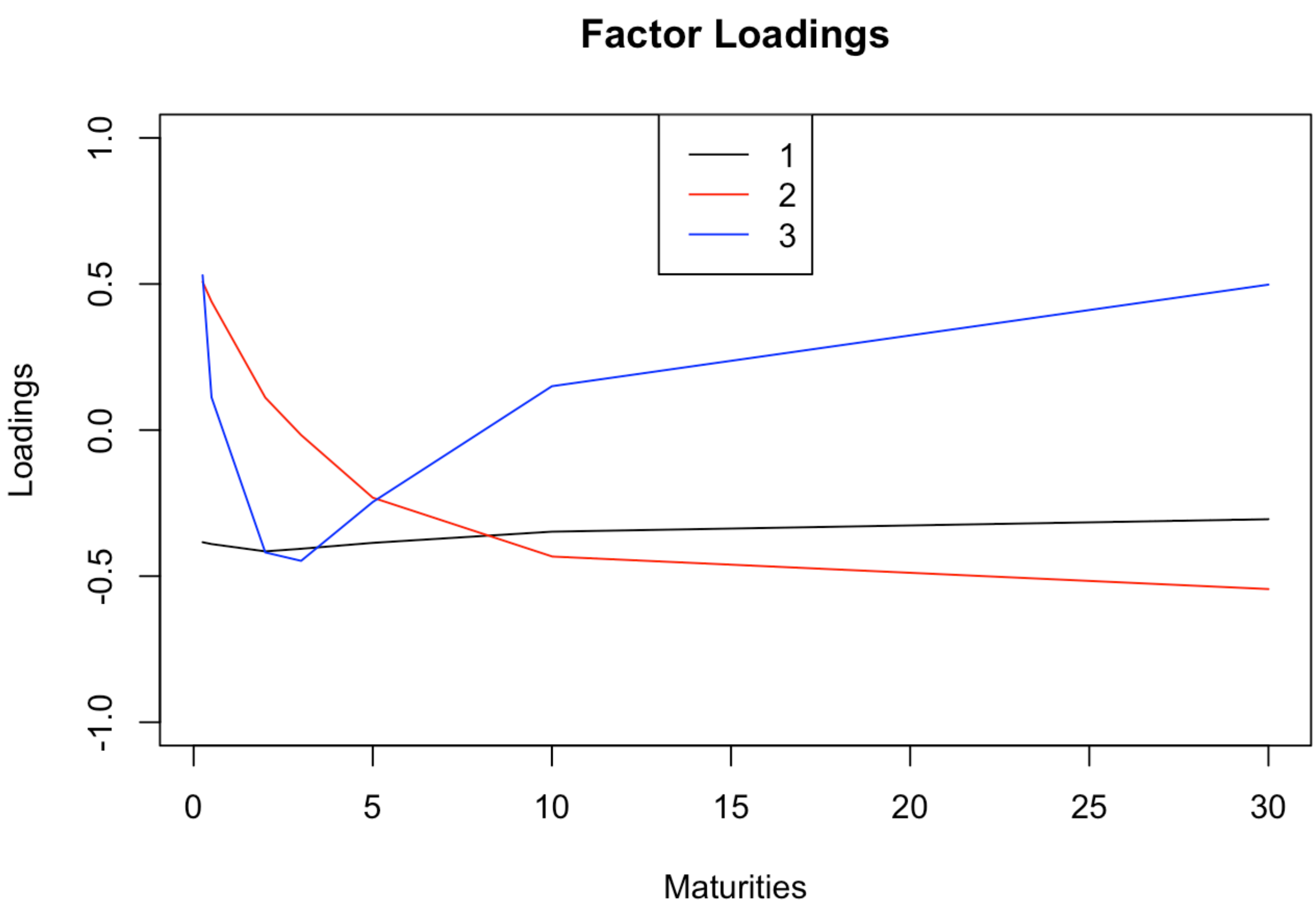


NOTE: Here I also print the cumulative values

```
print(round(cumsum(eigenvalues)/sum(eigenvalues)*100,2))
```

```
## [1]  97.83  99.81  99.97  99.98 100.00 100.00 100.00
```

```
# plot and interpret the shapes of factor loadings
plot(maturities,eigenvectors[,1],type='l',ylim=c(-1,1),main='Factor Loadings',
     xlab='Maturities',ylab='Loadings')
lines(maturities,eigenvectors[,2],col='red')
lines(maturities,eigenvectors[,3],col='blue')
legend('top',c("1","2",'3'),lty=c(1,1),col=c('black','red','blue'),cex=1)
```



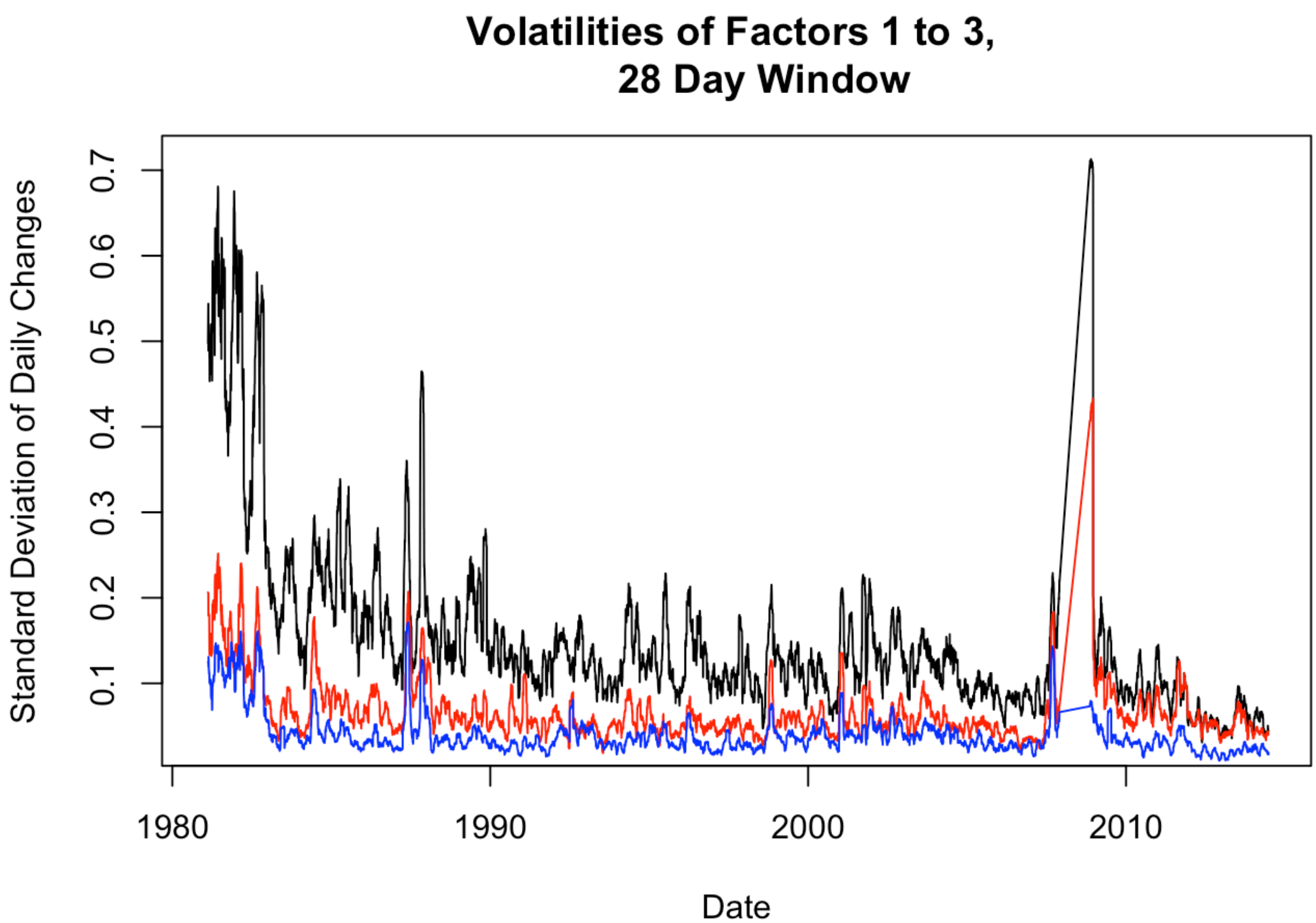

```
##############################################################################
# calculate historical volatilities and correlation coefficients of factors #
##############################################################################

# use the whole period of history to calculate var and cor
delta.fi <- diff(PC[,1:3])
print(diag(var(delta.fi)))
```

```
## [1] 0.034929992 0.006198080 0.002231987
```

```
print(cor(delta.fi))
```

```
##              [,1]       [,2]       [,3]
## [1,] 1.00000000 0.01394172 0.05079403
## [2,] 0.01394172 1.00000000 0.46744527
## [3,] 0.05079403 0.46744527 1.00000000
```

```
# calculate the same variables using a rolling window (approx 1 month)
plot.dates <- as.Date(rownames(runSD(delta.fi[,1])))
plot(plot.dates,runSD(delta.fi[,1],n=28),type='l',
     main='Volatilities of Factors 1 to 3,
     28 Day Window',ylab='Standard Deviation of Daily Changes',xlab='Date')
lines(plot.dates,runSD(delta.fi[,2],n=28),col='red')
lines(plot.dates,runSD(delta.fi[,3],n=28),col='blue')
```

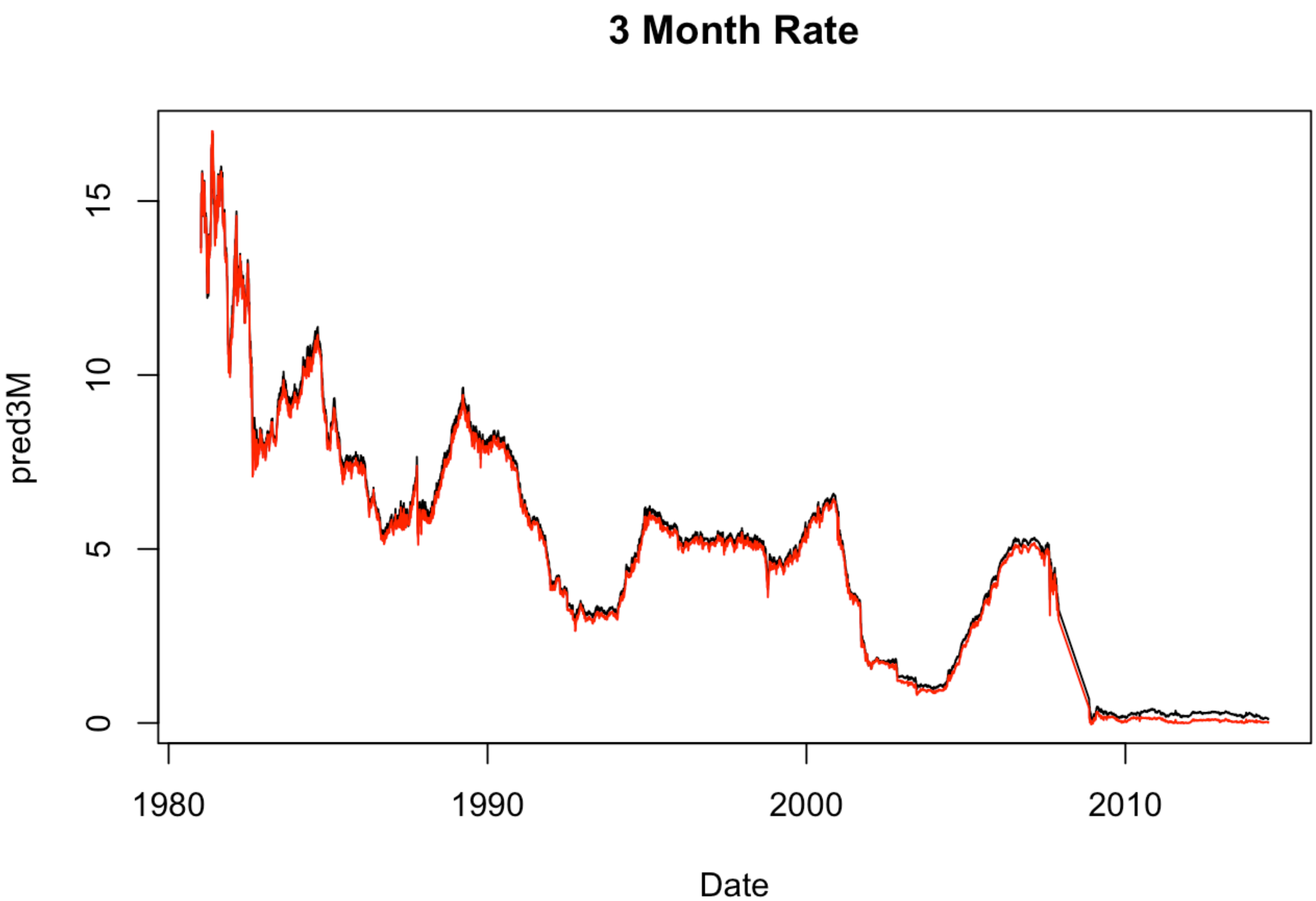


```
######################################################################
# find historical estimates of volatilities of the first 3 factors #
# corresponding to the last month of the observed period           #
######################################################################

PC.sub <- subset(PC[,1:3],rownames(PC)>='2014-06-01')
print(diag(var(PC.sub)))
```

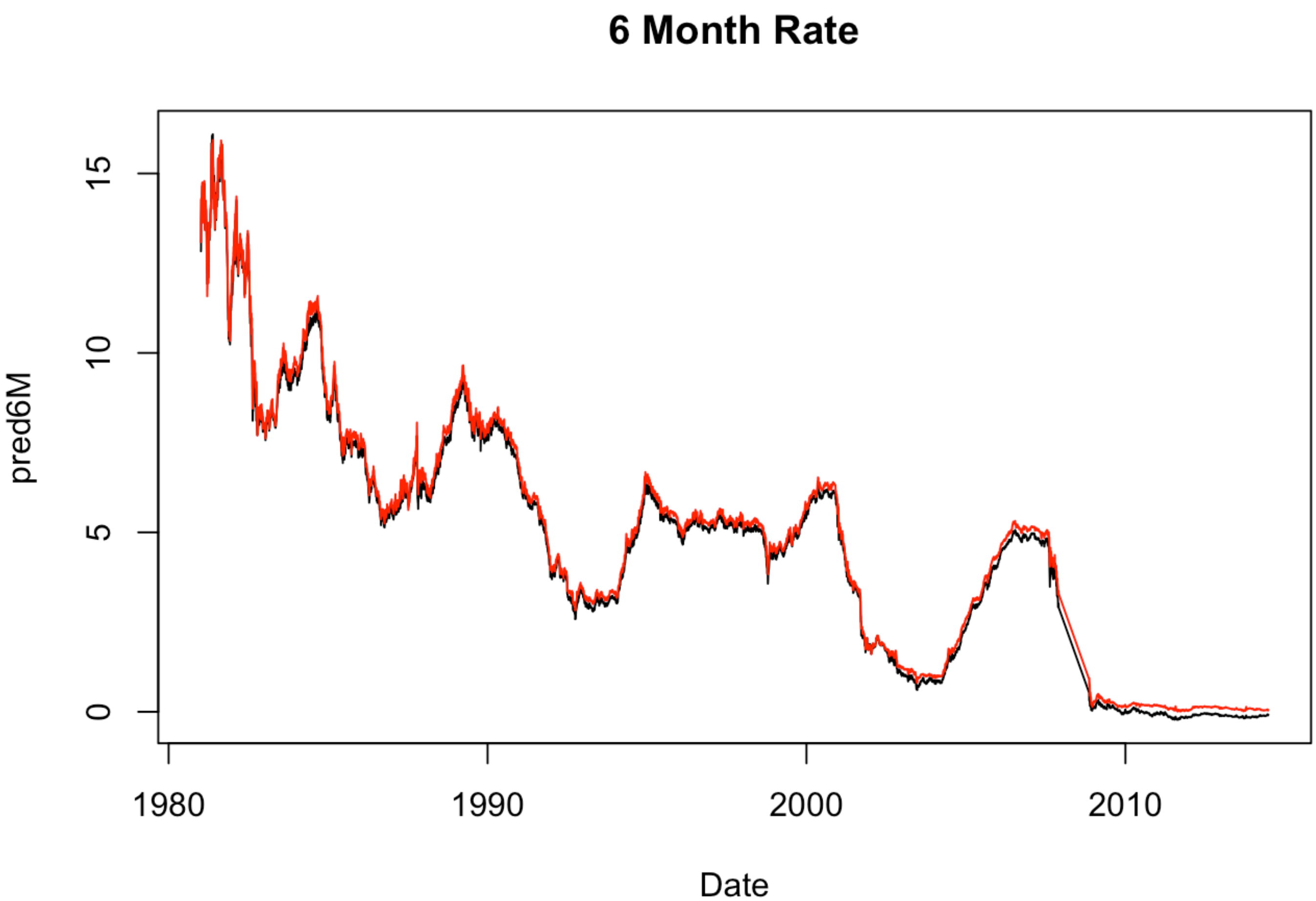
```
## [1] 0.003411459 0.001527943 0.002283805
```

NOTE: Here I have calculated and plotted each time series of the seven rates.

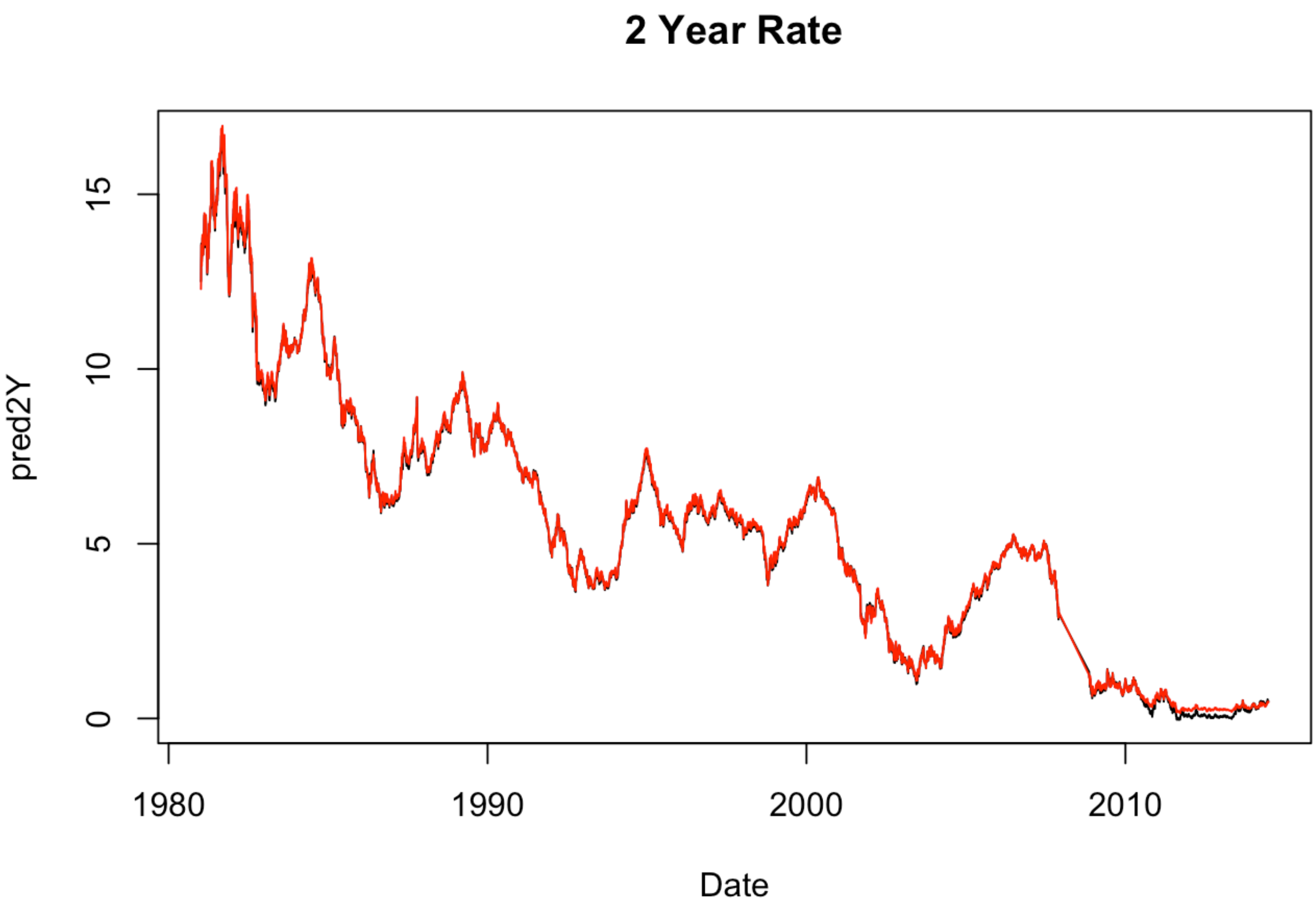
```
############################################################################
# calculate time series of each of the seven rates predicted by the model #
############################################################################

pred3M <- PC[,1:3] %*% eigenvectors[1,][1:3]
plot.dates <- as.Date(rownames(pred3M))
plot(plot.dates,pred3M,type='l',main='3 Month Rate',xlab='Date')
lines(plot.dates,df[,1],col='red')
```



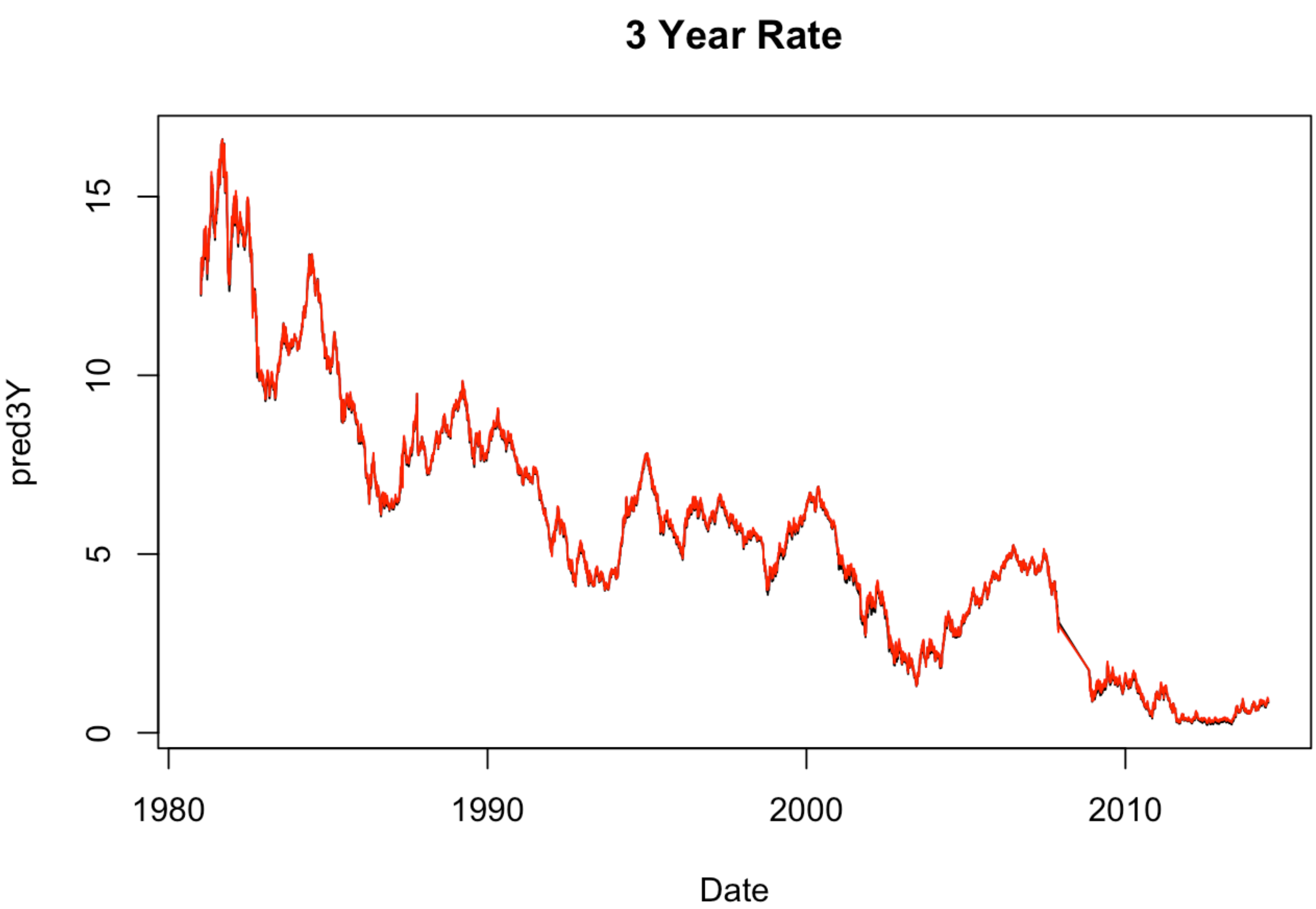

```
pred6M <- PC[,1:3] %*% eigenvectors[2,][1:3]
plot.dates <- as.Date(rownames(pred6M))
plot(plot.dates,pred6M,type='l',main='6 Month Rate',xlab='Date')
lines(plot.dates,df[,2],col='red')
```



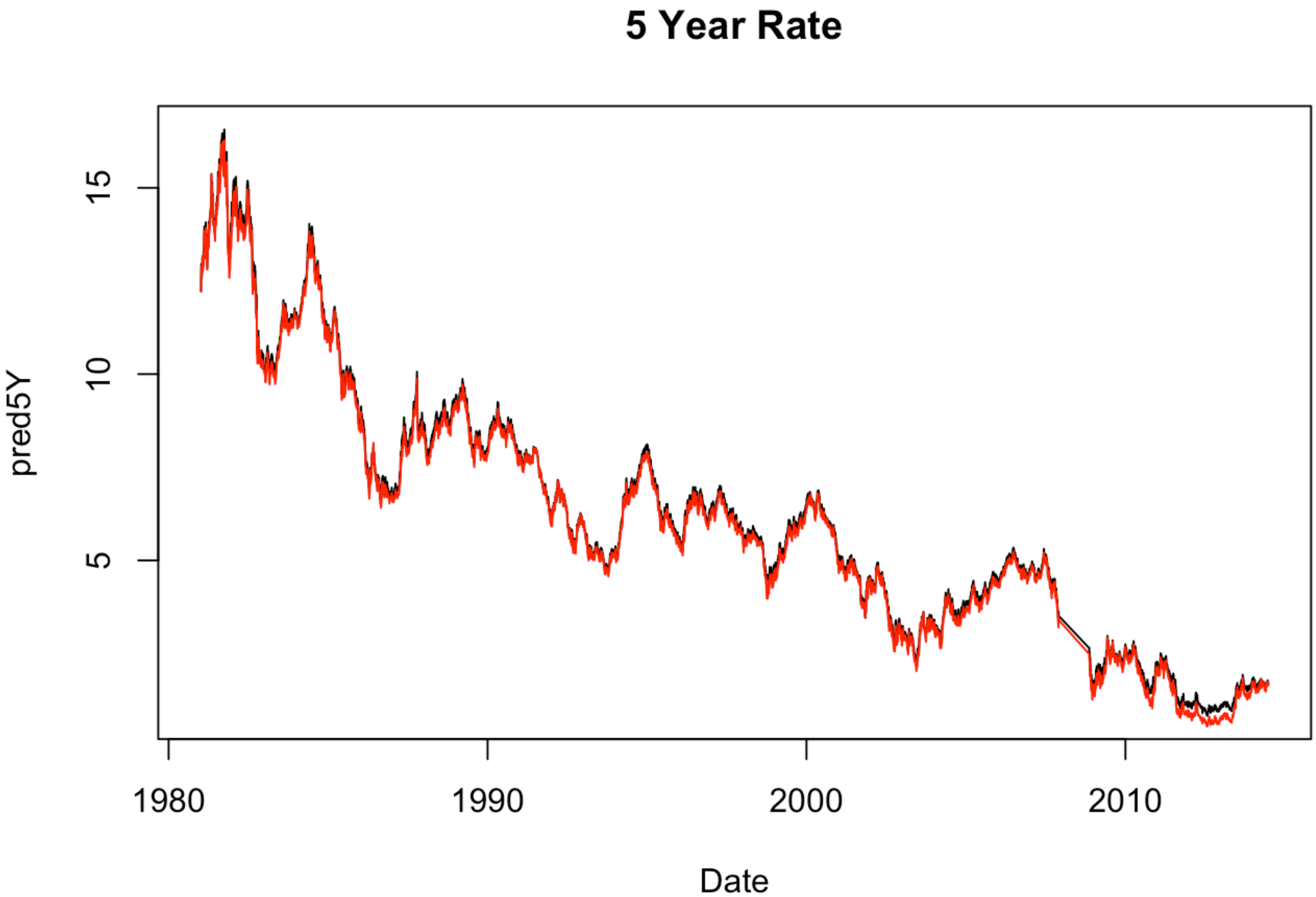

```
pred2Y <- PC[,1:3] %*% eigenvectors[3,][1:3]
plot.dates <- as.Date(rownames(pred2Y))
plot(plot.dates,pred2Y,type='l',main='2 Year Rate',xlab='Date')
lines(plot.dates,df[,3],col='red')
```



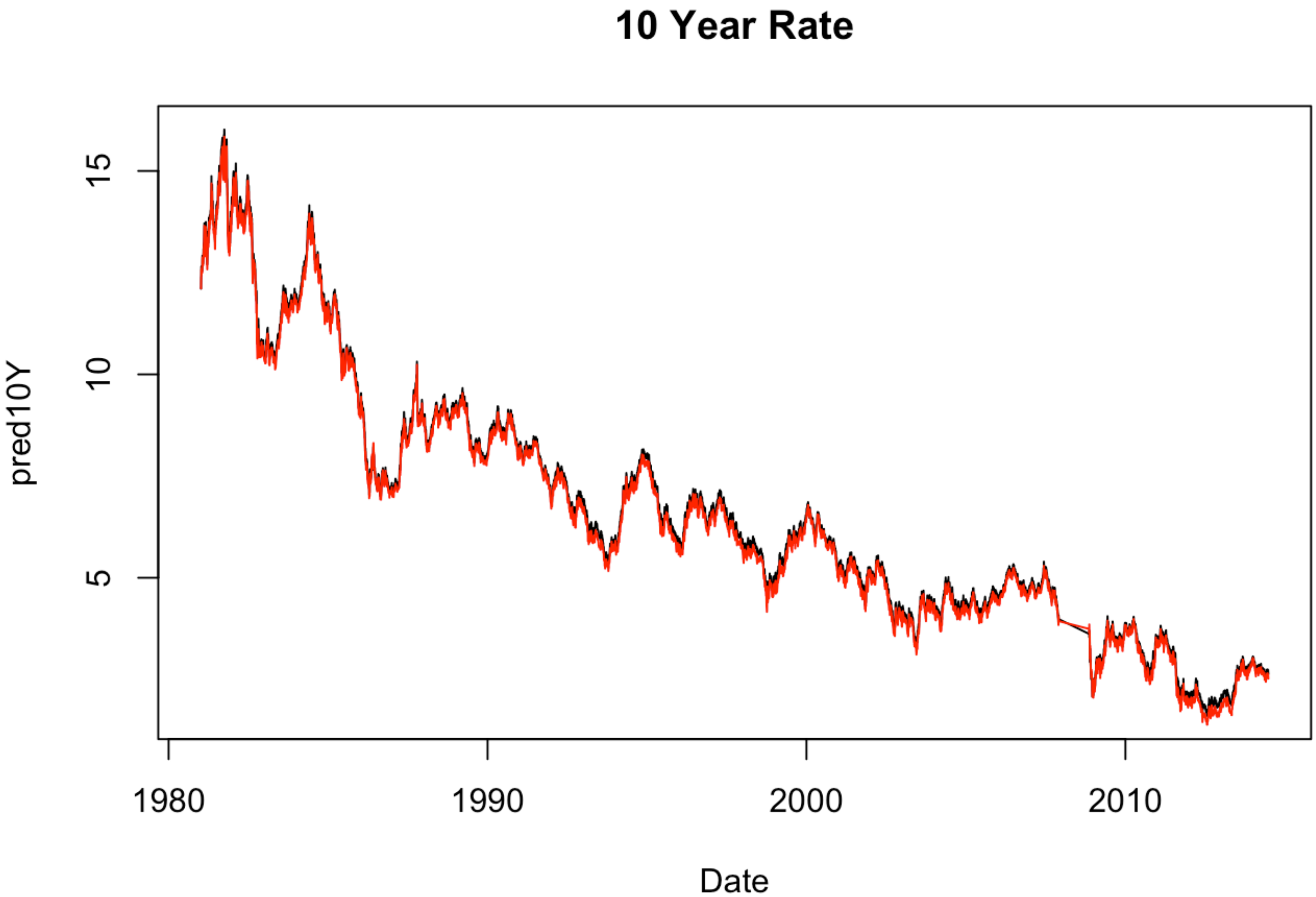

```
pred3Y <- PC[,1:3] %*% eigenvectors[4,][1:3]
plot.dates <- as.Date(rownames(pred3Y))
plot(plot.dates,pred3Y,type='l',main='3 Year Rate',xlab='Date')
lines(plot.dates,df[,4],col='red')
```



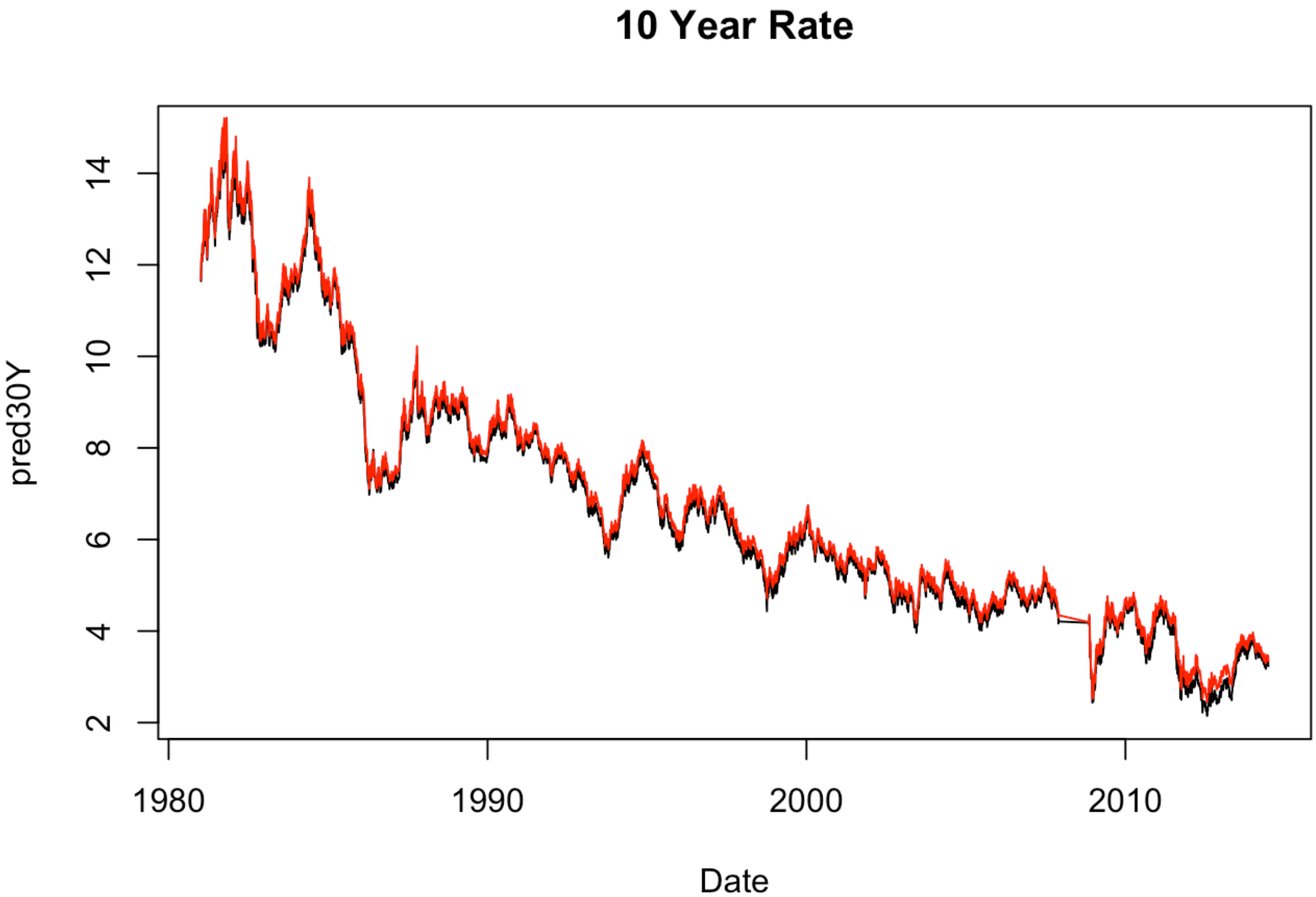

```
pred5Y <- PC[,1:3] %*% eigenvectors[5,][1:3]
plot.dates <- as.Date(rownames(pred5Y))
plot(plot.dates,pred5Y,type='l',main='5 Year Rate',xlab='Date')
lines(plot.dates,df[,5],col='red')
```




```
pred10Y <- PC[,1:3] %*% eigenvectors[6,][1:3]
plot.dates <- as.Date(rownames(pred10Y))
plot(plot.dates,pred10Y,type='l',main='10 Year Rate',xlab='Date')
lines(plot.dates,df[,6],col='red')
```




```
pred30Y <- PC[,1:3] %*% eigenvectors[7,][1:3]
plot.dates <- as.Date(rownames(pred30Y))
plot(plot.dates,pred30Y,type='l',main='10 Year Rate',xlab='Date')
lines(plot.dates,df[,7],col='red')
```



NOTE: I spoke to Marek in the TA session about this, and he said that I could run the parameter optimization for the loading at each maturity, which I have done below. The maturity is coded as `tau` .

```
#################################################################
# fit parametric forms from slide 32 to each of the first three #
# vectors of factor loadings                                    #
#################################################################

Loading.1 <- matrix(c(0.320,0.006,36.550,0.070,0.285,-0.292),nrow=3,ncol=2)
rownames(Loading.1) <- c(1,2,3)
colnames(Loading.1) <- c('a','b')
L.bound.1 <- c(0,0,0,-Inf,-Inf,-Inf)

Loading.2 <- matrix(c(0.650,0.004,-1.130,0.539),nrow=2,ncol=2)
rownames(Loading.2) <- c(1,2)
colnames(Loading.2) <- c('a','b')
L.bound.2 <- c(0,0,-Inf,-Inf)

Loading.3 <- matrix(c(4.200e-01,5e-08,5e-01,2.876,-1.92,0.62,-0.41,3.035),nrow=4,ncol
=2)
rownames(Loading.3) <- c(1,2,3,4)
colnames(Loading.3) <- c('a','b')
L.bound.3 <- c(0,0,0,0,-Inf,-Inf,-Inf,-Inf)
```

```
fn <- function(mat) {
  mat <- matrix(mat,ncol=2)
  return(abs(pca.loading - sum(mat[,2] * (1-exp(-mat[,1]*tau))/(mat[,1]*tau))))}
```

```
# Loading 1
tau <- 0.25
pca.loading <- eigenvectors[1,1]
print(round(optim(Loading.1,fn,method='L-BFGS-B')$par,4))
```

```
##         a       b
## 1  0.3244 -0.2800
## 2  0.0204 -0.0789
## 3 36.5497 -0.3318
```

```
tau <- 0.5
pca.loading <- eigenvectors[2,1]
print(round(optim(Loading.1,fn,method='L-BFGS-B',lower=L.bound.1)$par,4))
```

```
##          a       b
## 1  0.3264 -0.2912
## 2  0.0342 -0.1053
## 3 36.5498 -0.3135
```

```
tau <- 2
pca.loading <- eigenvectors[3,1]
print(round(optim(Loading.1,fn,method='L-BFGS-B',lower=L.bound.1)$par,4))
```

```
##          a       b
## 1  0.4121 -0.3114
## 2  0.2713 -0.2579
## 3 36.5499 -0.2990
```

```
tau <- 3
pca.loading <- eigenvectors[4,1]
print(round(optim(Loading.1,fn,method='L-BFGS-B',lower=L.bound.1)$par,4))
```

```
##          a       b
## 1  0.4405 -0.3321
## 2  0.4315 -0.3912
## 3 36.5500 -0.2976
```

```
tau <- 5
pca.loading <- eigenvectors[5,1]
print(round(optim(Loading.1,fn,method='L-BFGS-B',lower=L.bound.1)$par,4))
```

```
##          a       b
## 1  0.3372 -0.3948
## 2  0.5340 -0.5555
## 3 36.5500 -0.2972
```

```
tau <- 10
pca.loading <- eigenvectors[6,1]
print(round(optim(Loading.1,fn,method='L-BFGS-B',lower=L.bound.1)$par,4))
```

```
##          a       b
## 1  0.2131 -0.6636
## 2  0.8104 -0.5879
## 3 36.5499 -0.2991
```

```
tau <- 30
pca.loading <- eigenvectors[7,1]
print(round(optim(Loading.1,fn,method='L-BFGS-B',lower=L.bound.1)$par,4))
```

```
##         a       b
## 1  0.3196 -2.6587
## 2  1.0898 -0.8862
## 3 36.5498 -0.3168
```

```
# Loading 2
tau <- 0.25
pca.loading <- eigenvectors[1,2]
print(round(optim(Loading.2,fn,method='L-BFGS-B',lower=L.bound.2)$par,4))
```

```
##        a       b
## 1 0.7192 -0.6280
## 2 0.0020  1.0825
```

```
tau <- 0.5
pca.loading <- eigenvectors[2,2]
print(round(optim(Loading.2,fn,method='L-BFGS-B',lower=c(0,0,0,0))$par,4))
```

```
##        a      b
## 1 0.6500 0.0000
## 2 0.0282 0.4426
```

```
tau <- 2
pca.loading <- eigenvectors[3,2]
print(round(optim(Loading.2,fn,method='L-BFGS-B',lower=c(0,0,0,0))$par,4))
```

```
##        a      b
## 1 0.6500 0.0594
## 2 0.4936 0.1226
```

```
tau <- 3
pca.loading <- eigenvectors[4,2]
print(round(optim(Loading.2,fn,method='L-BFGS-B',lower=L.bound.2)$par,4))
```

```
##        a       b
## 1 0.6359 -1.1420
## 2 0.0258  0.5121
```

```
tau <- 5
pca.loading <- eigenvectors[5,2]
print(round(optim(Loading.2,fn,method='L-BFGS-B',lower=L.bound.2)$par,4))
```

```
##        a       b
## 1 0.8354 -1.0661
## 2 0.8629  0.0871
```

```
tau <- 10
pca.loading <- eigenvectors[6,2]
print(round(optim(Loading.2,fn,method='L-BFGS-B')$par,4))
```

```
##        a       b
## 1 0.2794 -1.3163
## 2 0.9938  0.0984
```

```
tau <- 30
pca.loading <- eigenvectors[7,2]
print(round(optim(Loading.2,fn,method='L-BFGS-B')$par,4))
```

```
##        a       b
## 1 0.0797 -1.4494
## 2 1.0899  0.2124
```

```
# Loading 3
tau <- 0.25
pca.loading <- eigenvectors[1,3]
print(round(optim(Loading.3,fn,method='L-BFGS-B')$par,4))
```

```
##        a       b
## 1 0.4169 -1.9330
## 2 0.0011  0.6063
## 3 0.4994 -0.4228
## 4 2.8793  3.0253
```

```
tau <- 0.5
pca.loading <- eigenvectors[2,3]
print(round(optim(Loading.3,fn,method='L-BFGS-B')$par,4))
```

```
##        a       b
## 1 0.4168 -1.9266
## 2 0.0013  0.6127
## 3 0.4992 -0.4165
## 4 2.8784  3.0310
```

```
tau <- 2
pca.loading <- eigenvectors[3,3]
print(round(optim(Loading.3,fn,method='L-BFGS-B')$par,4))
```

```
##        a       b
## 1 0.4179 -1.9212
## 2 0.0011  0.6181
## 3 0.4996 -0.4111
## 4 2.8763  3.0347
```

```
tau <- 3
pca.loading <- eigenvectors[4,3]
print(round(optim(Loading.3,fn,method='L-BFGS-B')$par,4))
```

```
##        a       b
## 1 0.3846 -1.9354
## 2 0.0252  0.5929
## 3 0.4934 -0.4241
## 4 2.8793  3.0318
```

```
tau <- 5
pca.loading <- eigenvectors[5,3]
print(round(optim(Loading.3,fn,method='L-BFGS-B')$par,4))
```

```
##        a       b
## 1 0.3892 -1.9298
## 2 0.0348  0.5970
## 3 0.4945 -0.4186
## 4 2.8777  3.0334
```

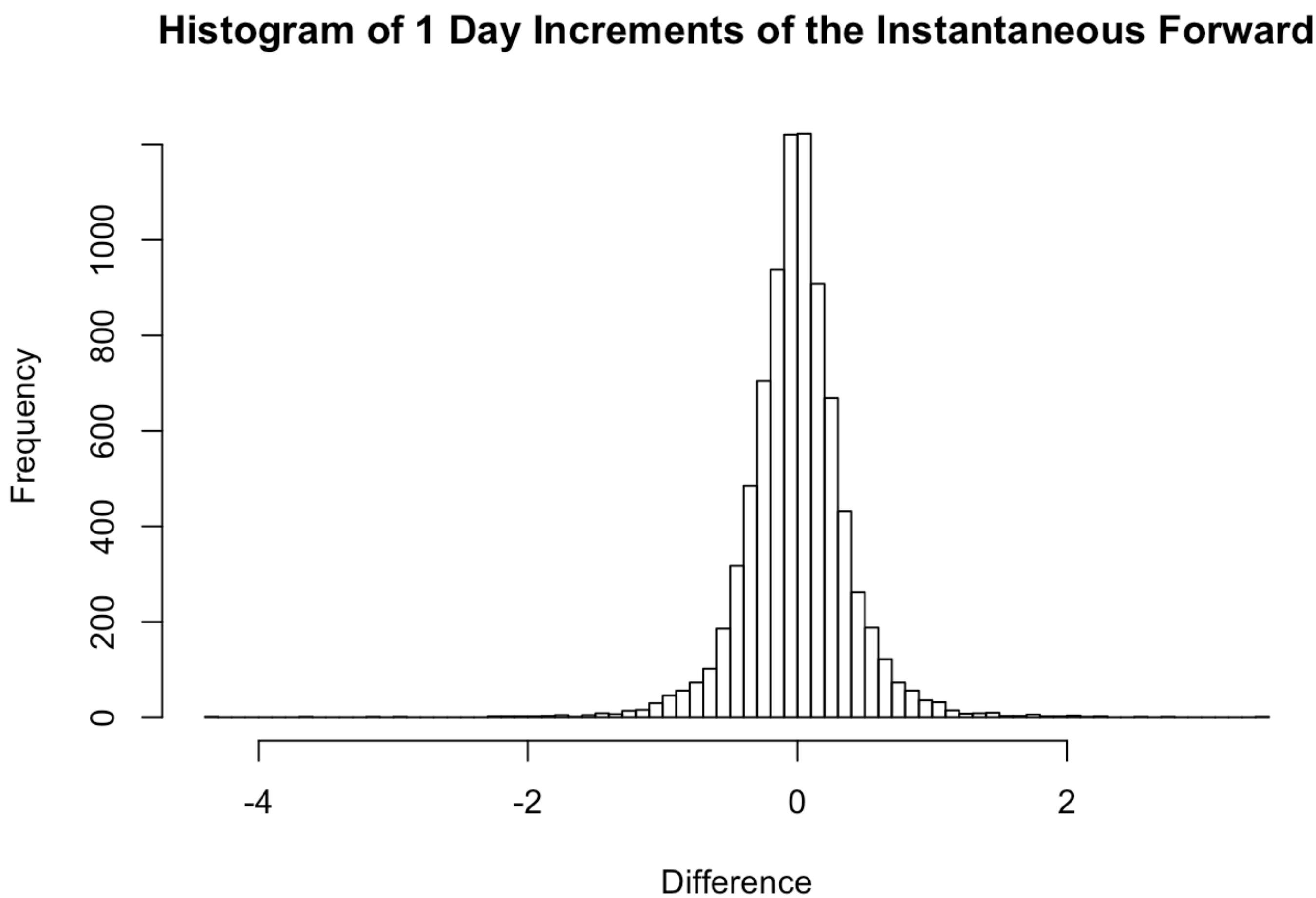
```
tau <- 10
pca.loading <- eigenvectors[6,3]
print(round(optim(Loading.3,fn,method='L-BFGS-B')$par,4))
```

```
##        a       b
## 1 0.4161 -1.9209
## 2 0.0120  0.6161
## 3 0.4994 -0.4108
## 4 2.8761  3.0349
```

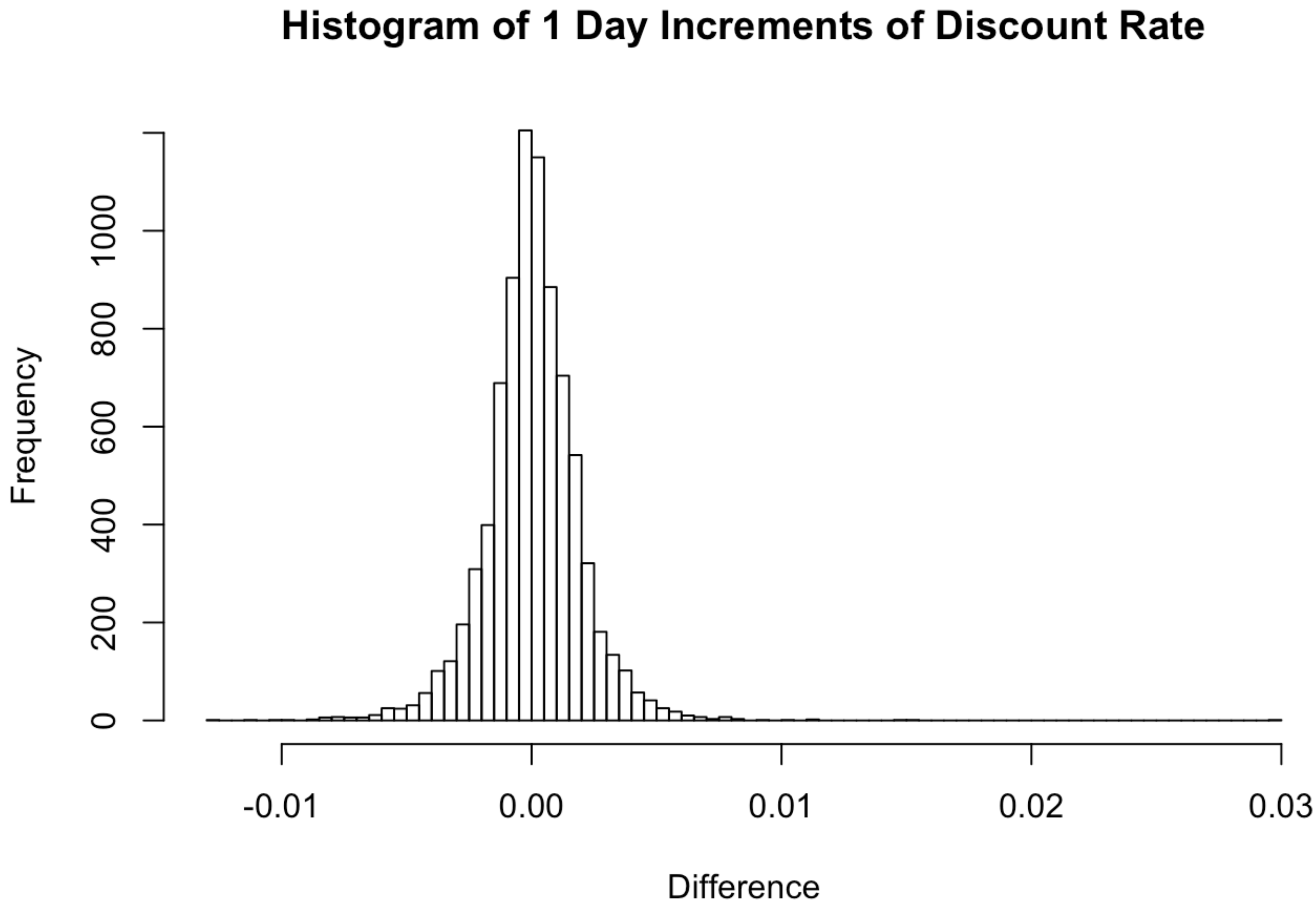
```
tau <- 30
pca.loading <- eigenvectors[7,3]
print(round(optim(Loading.3,fn)$par,4))
```

```
##        a       b
## 1 0.4379 -1.8366
## 2 0.0002  0.6214
## 3 0.5135 -0.2696
## 4 2.8982  3.0775
```

```
###############################################################################
# calculate time series of instataneous forward rates with maturity 5 years #
# and discount bonds with maturity 4.5 year for whole period of observation #
###############################################################################

forward <- 5*(PC[,1:3] %*% eigenvectors[5,][1:3]) + colMeans(df)[5]*5
loadings <- eigenvectors[5,][1:3] - (eigenvectors[5,][1:3] - eigenvectors[4,][1:3])/4
disc.mean <- colMeans(df)[5] - (colMeans(df)[5] - colMeans(df)[4])/2
disc.rates <- exp(-(((4.5*disc.mean) + (PC[,1:3] %*% loadings * 4.5)))/100)

hist(diff(forward),breaks=100,main='Histogram of 1 Day Increments of the Instantaneou
s Forward',
     xlab='Difference')
```



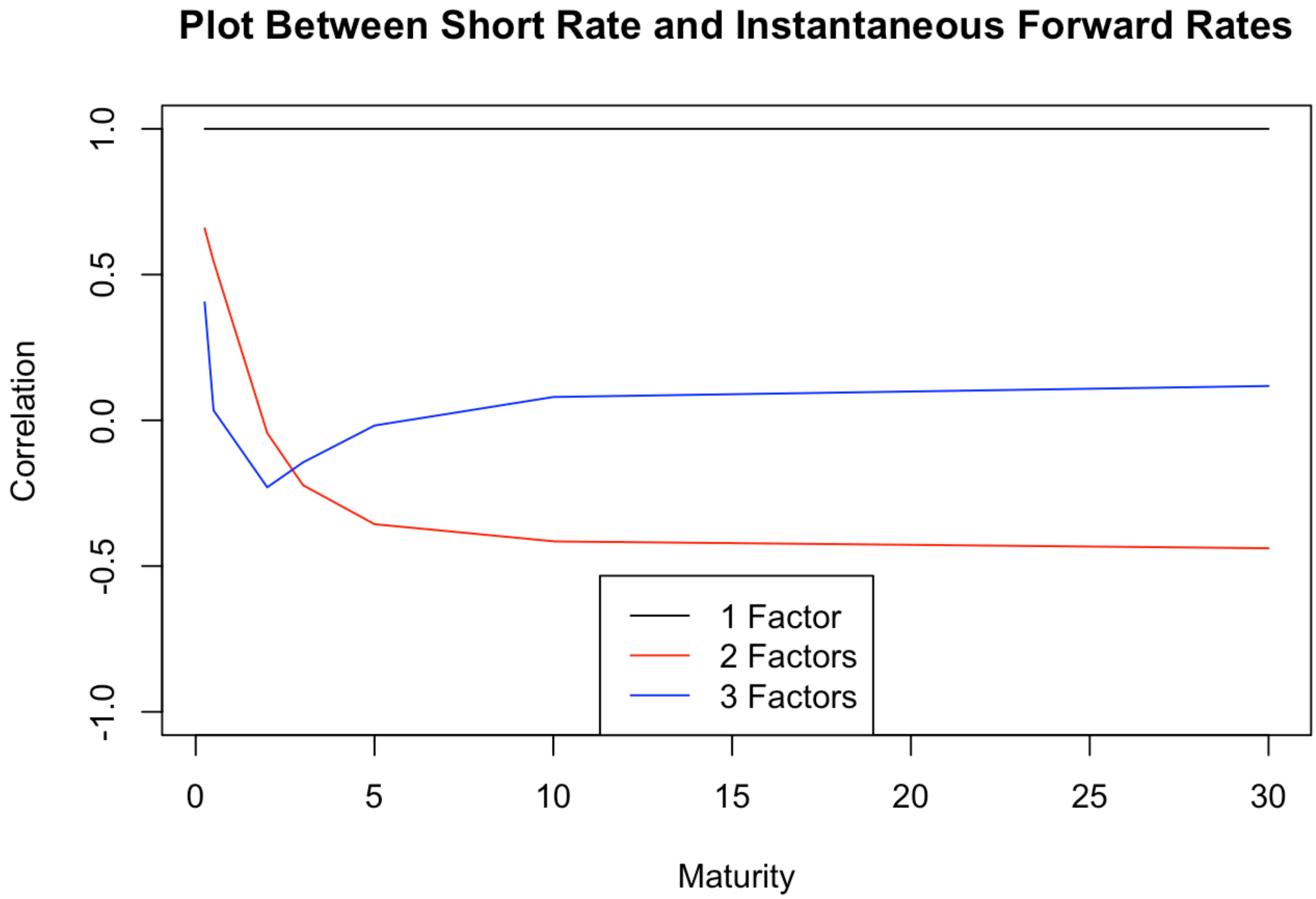

```
hist(diff(disc.rates),breaks=100,main='Histogram of 1 Day Increments of Discount Rate
',
     xlab='Difference')
```




```
####################################################################
# calculate correlations between the short rate and instantaneous #
# forward rates                                                    #
####################################################################

B1 <- function(x) {sum(Loading.1[,'b']*exp(-Loading.1[,'a']*x))}
B2 <- function(x) {sum(Loading.2[,'b']*exp(-Loading.2[,'a']*x))}
B3 <- function(x) {sum(Loading.3[,'b']*exp(-Loading.3[,'a']*x))}

sig2 <- diag(var(diff(PC[,1:3])))
B.mat <- matrix(c(B1(0),B2(0),B3(0),
                  B1(0.25),B2(0.25),B3(0.25),
                  B1(0.5),B2(0.5),B3(0.5),
                  B1(2),B2(2),B3(2),
                  B1(3),B2(3),B3(3),
                  B1(5),B2(5),B3(5),
                  B1(10),B2(10),B3(10),
                  B1(30),B2(30),B3(30),
                  sig2[1],sig2[2],sig2[3]),ncol=9)
```

```
rownames(B.mat) <- c('B1','B2','B3')
colnames(B.mat) <- c('0M','3M','6M','2Y','3Y','5Y','10Y','30Y','sig2')

# one factor model correlations
rho10 <- B.mat['B1','0M']*B.mat['B1','3M']/sqrt(B.mat['B1','0M']^2*B.mat['B1','3M']^2
)
rho11 <- B.mat['B1','0M']*B.mat['B1','6M']/sqrt(B.mat['B1','0M']^2*B.mat['B1','6M']^2
)
rho12 <- B.mat['B1','0M']*B.mat['B1','2Y']/sqrt(B.mat['B1','0M']^2*B.mat['B1','2Y']^2
)
rho13 <- B.mat['B1','0M']*B.mat['B1','3Y']/sqrt(B.mat['B1','0M']^2*B.mat['B1','3Y']^2
)
rho14 <- B.mat['B1','0M']*B.mat['B1','5Y']/sqrt(B.mat['B1','0M']^2*B.mat['B1','5Y']^2
)
rho15 <- B.mat['B1','0M']*B.mat['B1','10Y']/sqrt(B.mat['B1','0M']^2*B.mat['B1','10Y']
^2)
rho16 <- B.mat['B1','0M']*B.mat['B1','30Y']/sqrt(B.mat['B1','0M']^2*B.mat['B1','30Y']
^2)
one.factor.rho <- c(rho10,rho11,rho12,rho13,rho14,rho15,rho16)

# two factor model correlations
B.mat.tmp <- B.mat[1:2,]
rho20 <- sum(B.mat.tmp[,'0M']*B.mat.tmp[,'3M']*B.mat.tmp[,'sig2'])/
  sqrt(sum(B.mat.tmp[,'0M']^2*B.mat.tmp[,'sig2'])*sum(B.mat.tmp[,'3M']^2*B.mat.tmp[,'
sig2']))
rho21 <- sum(B.mat.tmp[,'0M']*B.mat.tmp[,'6M']*B.mat.tmp[,'sig2'])/
  sqrt(sum(B.mat.tmp[,'0M']^2*B.mat.tmp[,'sig2'])*sum(B.mat.tmp[,'6M']^2*B.mat.tmp[,'
sig2']))
rho22 <- sum(B.mat.tmp[,'0M']*B.mat.tmp[,'2Y']*B.mat.tmp[,'sig2'])/
  sqrt(sum(B.mat.tmp[,'0M']^2*B.mat.tmp[,'sig2'])*sum(B.mat.tmp[,'2Y']^2*B.mat.tmp[,'
sig2']))
rho23 <- sum(B.mat.tmp[,'0M']*B.mat.tmp[,'3Y']*B.mat.tmp[,'sig2'])/
  sqrt(sum(B.mat.tmp[,'0M']^2*B.mat.tmp[,'sig2'])*sum(B.mat.tmp[,'3Y']^2*B.mat.tmp[,'
sig2']))
rho24 <- sum(B.mat.tmp[,'0M']*B.mat.tmp[,'5Y']*B.mat.tmp[,'sig2'])/
  sqrt(sum(B.mat.tmp[,'0M']^2*B.mat.tmp[,'sig2'])*sum(B.mat.tmp[,'5Y']^2*B.mat.tmp[,'
sig2']))
rho25 <- sum(B.mat.tmp[,'0M']*B.mat.tmp[,'10Y']*B.mat.tmp[,'sig2'])/
  sqrt(sum(B.mat.tmp[,'0M']^2*B.mat.tmp[,'sig2'])*sum(B.mat.tmp[,'10Y']^2*B.mat.tmp[,
'sig2']))
rho26 <- sum(B.mat.tmp[,'0M']*B.mat.tmp[,'30Y']*B.mat.tmp[,'sig2'])/
  sqrt(sum(B.mat.tmp[,'0M']^2*B.mat.tmp[,'sig2'])*sum(B.mat.tmp[,'30Y']^2*B.mat.tmp[,
'sig2']))
two.factor.rho <- c(rho20,rho21,rho22,rho23,rho24,rho25,rho26)

# three factor model correlations
rho30 <- sum(B.mat[,'0M']*B.mat[,'3M']*B.mat[,'sig2'])/
```

```
  sqrt(sum(B.mat[,'0M']^2*B.mat[,'sig2'])*sum(B.mat[,'3M']^2*B.mat[,'sig2']))
rho31 <- sum(B.mat[,'0M']*B.mat[,'6M']*B.mat[,'sig2'])/
  sqrt(sum(B.mat[,'0M']^2*B.mat[,'sig2'])*sum(B.mat[,'6M']^2*B.mat[,'sig2']))
rho32 <- sum(B.mat[,'0M']*B.mat[,'2Y']*B.mat[,'sig2'])/
  sqrt(sum(B.mat[,'0M']^2*B.mat[,'sig2'])*sum(B.mat[,'2Y']^2*B.mat[,'sig2']))
rho33 <- sum(B.mat[,'0M']*B.mat[,'3Y']*B.mat[,'sig2'])/
  sqrt(sum(B.mat[,'0M']^2*B.mat[,'sig2'])*sum(B.mat[,'3Y']^2*B.mat[,'sig2']))
rho34 <- sum(B.mat[,'0M']*B.mat[,'5Y']*B.mat[,'sig2'])/
  sqrt(sum(B.mat[,'0M']^2*B.mat[,'sig2'])*sum(B.mat[,'5Y']^2*B.mat[,'sig2']))
rho35 <- sum(B.mat[,'0M']*B.mat[,'10Y']*B.mat[,'sig2'])/
  sqrt(sum(B.mat[,'0M']^2*B.mat[,'sig2'])*sum(B.mat[,'10Y']^2*B.mat[,'sig2']))
rho36 <- sum(B.mat[,'0M']*B.mat[,'30Y']*B.mat[,'sig2'])/
  sqrt(sum(B.mat[,'0M']^2*B.mat[,'sig2'])*sum(B.mat[,'30Y']^2*B.mat[,'sig2']))
three.factor.rho <- c(rho30,rho31,rho32,rho33,rho34,rho35,rho36)

plot(maturities,one.factor.rho,type='l',ylim=c(-1,1),main='Plot Between Short Rate an
d Instantaneous Forward Rates',
     ylab='Correlation',xlab='Maturity')
lines(maturities,two.factor.rho,col='red')
lines(maturities,three.factor.rho,col='blue')
legend('bottom',c("1 Factor","2 Factors",'3 Factors'),lty=c(1,1),col=c('black','red',
'blue'),cex=1)
```



Plot Between Short Rate and Instantaneous Forward Rates
Correlation
1.0
0.5
0.0
-0.5
-1.0
0
5
10
15
20
25
30
Maturity
1 Factor
2 Factors
3 Factors