Project

Jiaqi Jiang

4/11/2020

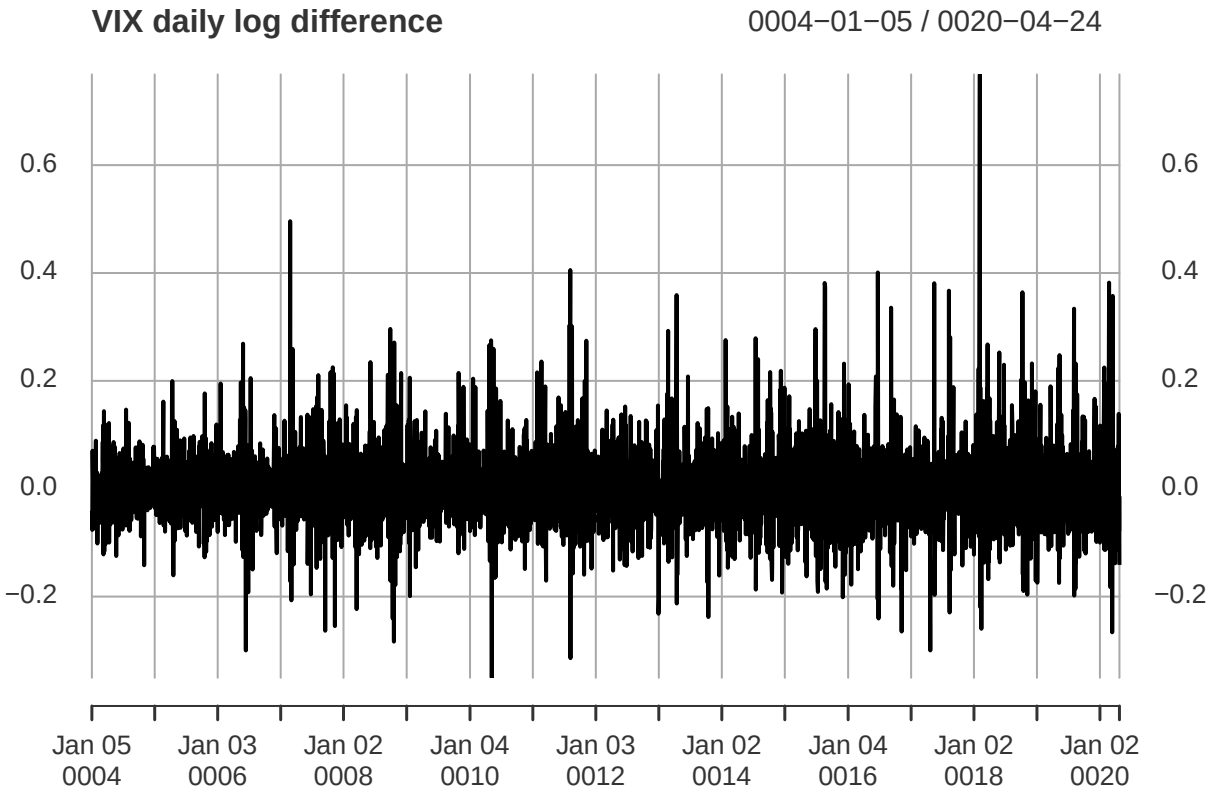
R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
rm(list=ls())
data<- read.csv("vixcurrent.csv",header=T)
data$Date=as.POSIXct(data$Date,format='%m/%d/%Y')
data=xts(data[,5],data[,1])
colnames(data)="rate"
```

```
#Differencing the series
diff.rate=diff(log(data$rate)); diff.rate <- diff.rate[!is.na(diff.rate)]
#Plot differenced series
plot(diff.rate,type='l',main='VIX daily log difference', ylab="Difference")
```




```
#train test split
num=nrow(diff.rate)
n_split = num-9
## Training data
diff.train=diff.rate[1:n_split,]
## Test data
diff.test=diff.rate[n_split:num,]
```

```
#GARCH Order Selection
library(rugarch)
#Select model with smallest BIC
final.bic = Inf
final.order = c(0,0)
for (m in 0:3) for (n in 0:3){
  spec = ugarchspec(variance.model=list(garchOrder=c(m,n)),
    mean.model=list(armaOrder=c(1, 2), include.mean=T),
    distribution.model="std")
    fit = ugarchfit(spec, diff.train, solver = 'hybrid')
       current.bic = infocriteria(fit)[2]
    if (current.bic < final.bic){
     final.bic = current.bic
        final.order = c(m, n)
    }}
final.order
```

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

```
#Refine the ARMA order
final.bic = Inf
final.order.arma = c(0,0)
for (p in 0:6) for (q in 0:6){
  spec = ugarchspec(variance.model=list(garchOrder=c(1,1)),
    mean.model=list(armaOrder=c(p, q), include.mean=T),
    distribution.model="std")
    fit = ugarchfit(spec, diff.train, solver = 'hybrid')
    current.bic = infocriteria(fit)[2]
    if (current.bic < final.bic){
       final.bic = current.bic
       final.order.arma = c(p, q)
    }
}
```

```
## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1
```

```
final.order.arma
```

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

```
#Final GARCH Order Selection
library(rugarch)
#Select model with smallest BIC
final.bic = Inf
final.order = c(0,0)
for (m in 0:3) for (n in 0:3){
  spec = ugarchspec(variance.model=list(garchOrder=c(m,n)),
    mean.model=list(armaOrder=c(2, 2), include.mean=T),
    distribution.model="std")
    fit = ugarchfit(spec, diff.train, solver = 'hybrid')
       current.bic = infocriteria(fit)[2]
    if (current.bic < final.bic){
     final.bic = current.bic
        final.order = c(m, n)
    }}
```

```
## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1
```

```
## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1

## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1
```

```r
final.order
```

```
## [1] 2 2
```

```r
spec.1 = ugarchspec(variance.model=list(garchOrder=c(1,1)),
mean.model=list(armaOrder=c(1,2), include.mean=T), distribution.model="std")
final.model.1 = ugarchfit(spec.1, diff.train, solver = 'hybrid')
infocriteria(final.model.1)
```

```
##
## Akaike       -2.656901
## Bayes        -2.644561
## Shibata      -2.656908
## Hannan-Quinn -2.652532
```

```
spec.2 = ugarchspec(variance.model=list(garchOrder=c(1,1)),
mean.model=list(armaOrder=c(2,2), include.mean=T), distribution.model="std")
final.model.2 = ugarchfit(spec.2, diff.train, solver = 'hybrid')
```

```
## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1
```

```
infocriteria(final.model.2)
```

```
##
## Akaike       -2.655675
## Bayes        -2.641793
## Shibata      -2.655684
## Hannan-Quinn -2.650760
```

```
#Prediction of the return time series and the volatility sigma
nfore = length(diff.test)
fore.series.1 = NULL
fore.sigma.1 = NULL
for(f in 1: nfore){
  diff = diff.train
  if(f>2){
    diff = c(diff.train,diff.test[1:(f-1)])}
  final.model.1 = ugarchfit(spec.1, diff, solver = 'hybrid')
    fore = ugarchforecast(final.model.1, n.ahead=1)
    fore.series.1 = c(fore.series.1, fore@forecast$seriesFor)
    fore.sigma.1 = c(fore.sigma.1, fore@forecast$sigmaFor)
}
```

```
#MSPE
mean((fore.series.1-diff.test))^2
```

```
## [1] 8.142082e-06
```

```
#mean absolute prediction error (MAE)
mean(abs(fore.series.1-diff.test))
```

```
## [1] 0.05945001
```

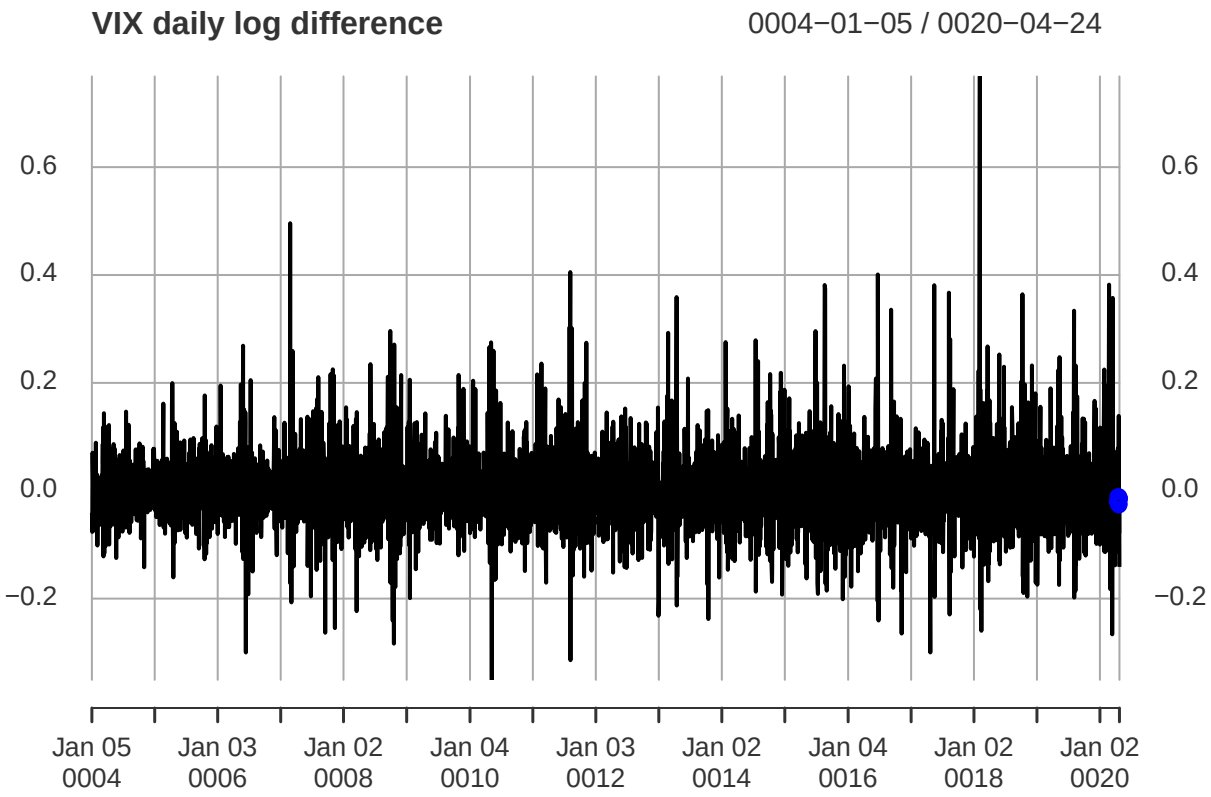
```
#Mean absolute percentage error (MAPE)
mean(abs(fore.series.1-diff.test)/(diff.test+0.000001))
```

```
## [1] 0.008108263
```

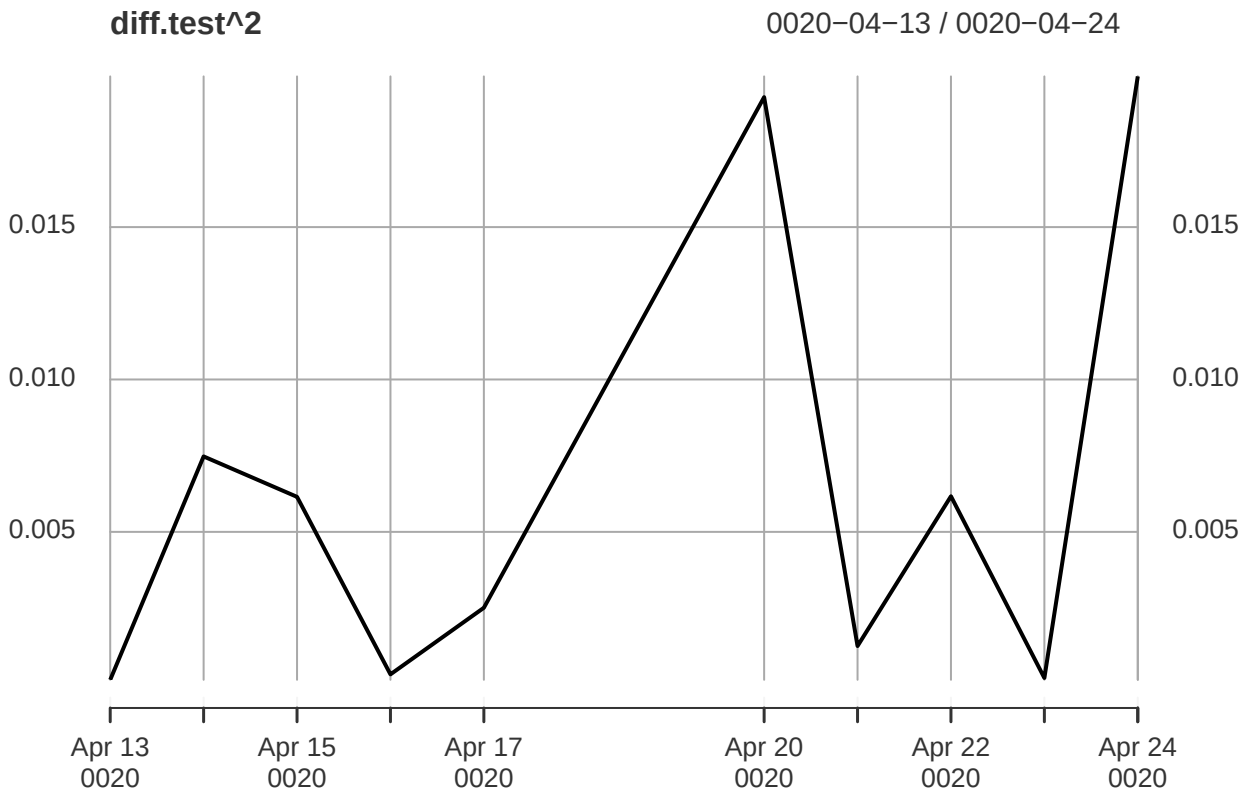
```
# Precision Measure (PM)
sum((fore.series.1-diff.test)^2)/sum((diff.test-mean(diff.test))^2)
```

```
## [1] 0.9628865
```

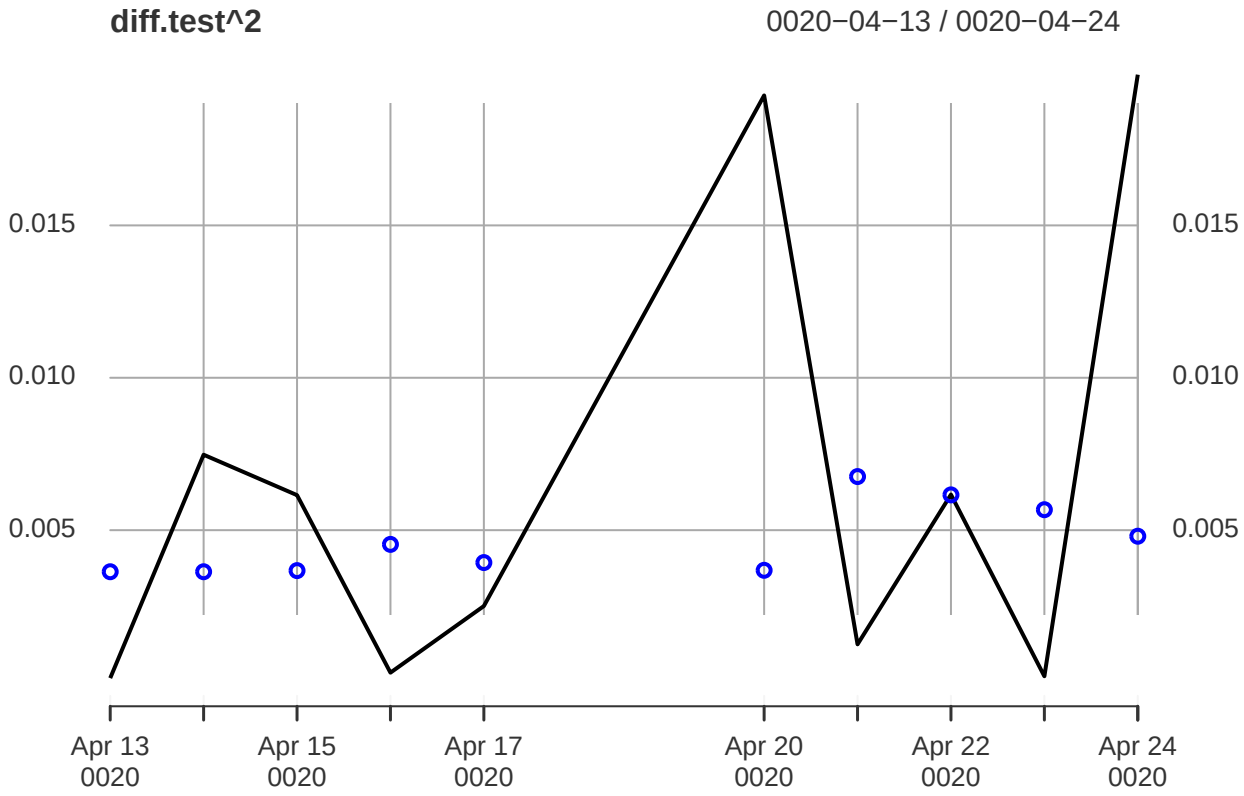
```
#Mean Prediction Comparison Plots
n=length(diff.rate)
diff.plot = diff.test
names(diff.plot)="Fore"
diff.plot$Fore=fore.series.1
points(diff.plot,lwd= 2, col="blue")
```




```
#Compare squared observed time series with variance forecasts
ymin = min(c(as.vector(diff.test^2),fore.sigma.1^2))
ymax = max(c(as.vector(diff.test^2),fore.sigma.1^2))
plot(diff.test^2,type="l", ylim=c(ymin,ymax), xlab=" ", ylab="USD/EUR Exchange Rate")
```




```
diff.plot$Fore=fore.sigma.1^2
points(diff.plot,lwd= 2, col="blue")
```



```
fore.series.1
```

```
##  [1] -0.02019326 -0.02019326 -0.01123292 -0.02031822 -0.01506412 -0.01090570
##  [7] -0.02635679 -0.02386590 -0.01400609 -0.01460691
```

This will be output to Python for plot purposes