Time Series Analysis 5

GARCH Time Series Models, Volatility modeling

Time Series Analysis
Zhe Zheng

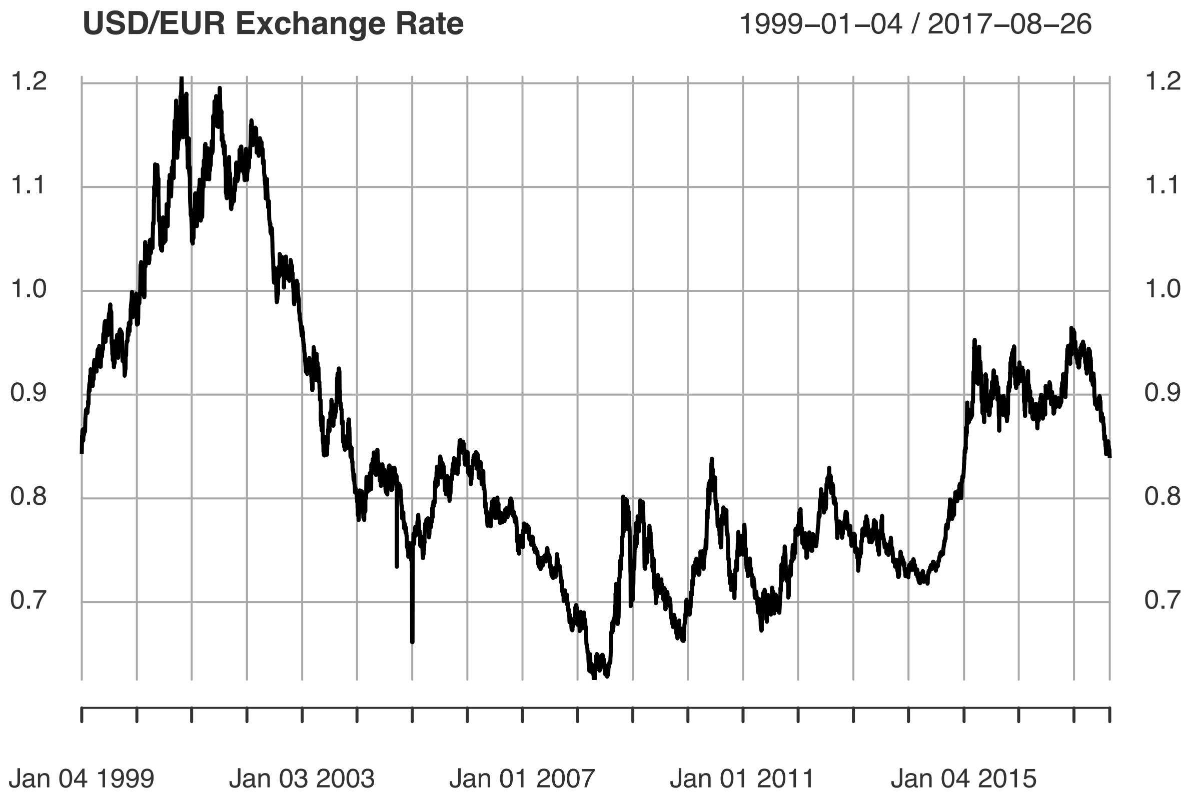
```
## [1] "English_United States.1252"
```

1.Exploratory Data Analysis

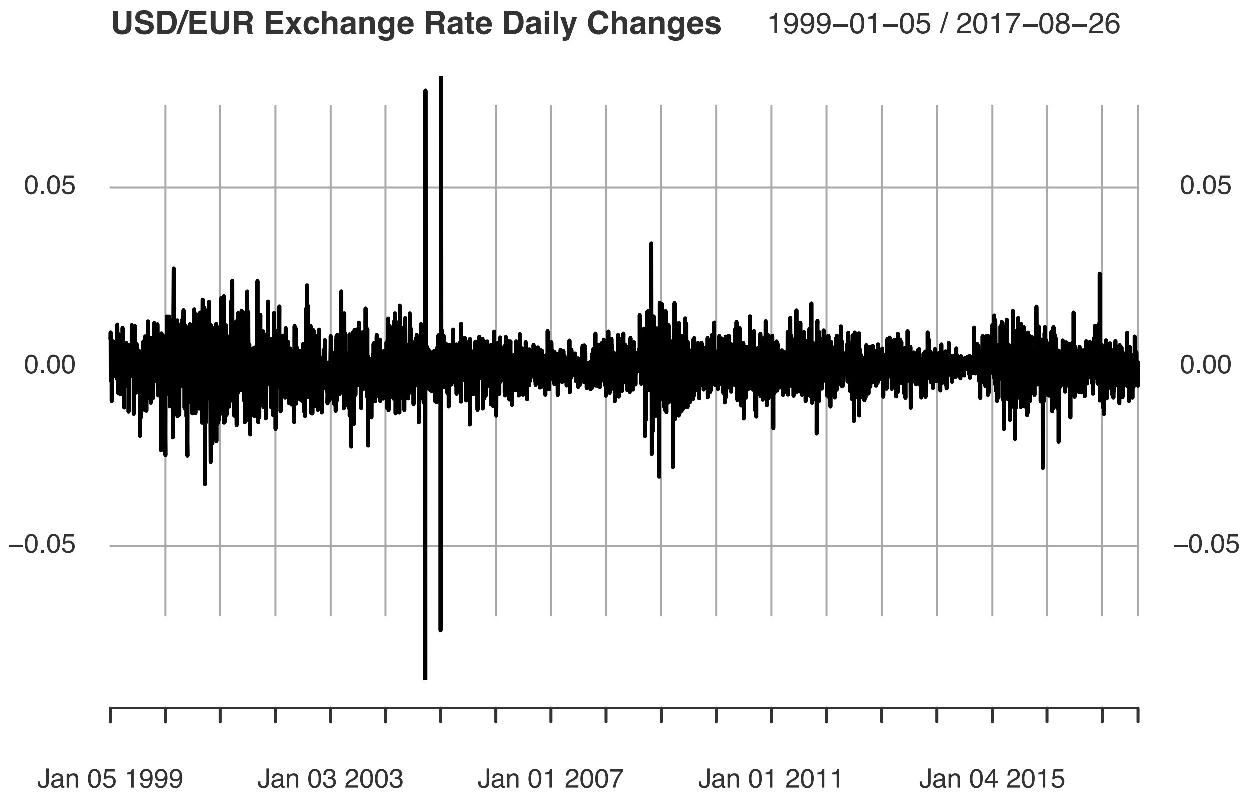
```r
library(xts)
library(rugarch)
#Load data , USD/EUR data
data=read.csv("exchange_rate_USD_EUR_daily.csv",header=TRUE)
data$Date=as.POSIXct(data$Date,format='%m/%d/%Y') #calendar time
data=xts(data[,2],data[,1]) #create an extensible time series(xts) object
colnames(data)="rate"
```

Plot original exchange rates and differenced series

```r
# Plot original series
plot(data$rate,type='l',main='USD/EUR Exchange Rate',ylab="Exchange rate")
```

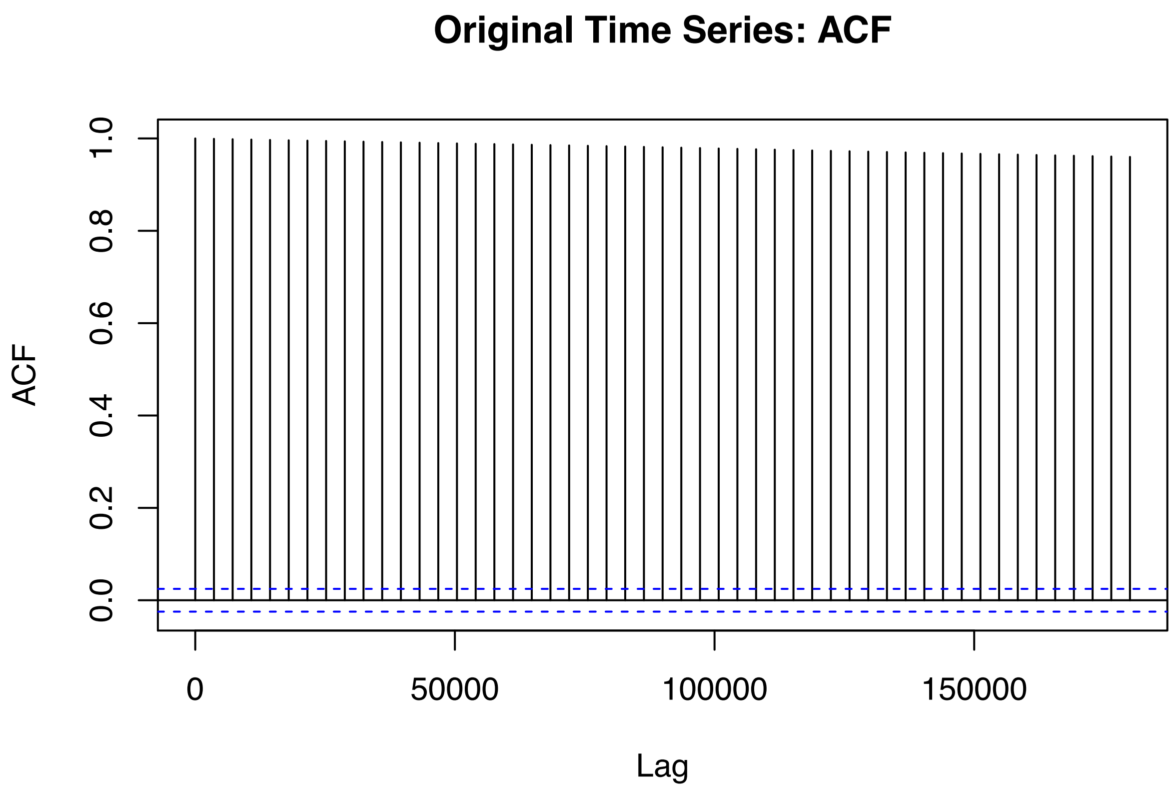



```r
#Differencing the series
diff.rate=diff(data$rate); diff.rate <- diff.rate[!is.na(diff.rate)]
#Plot differenced series
plot(diff.rate,type='l',main='USD/EUR Exchange Rate Daily Changes',ylab="Difference")
```

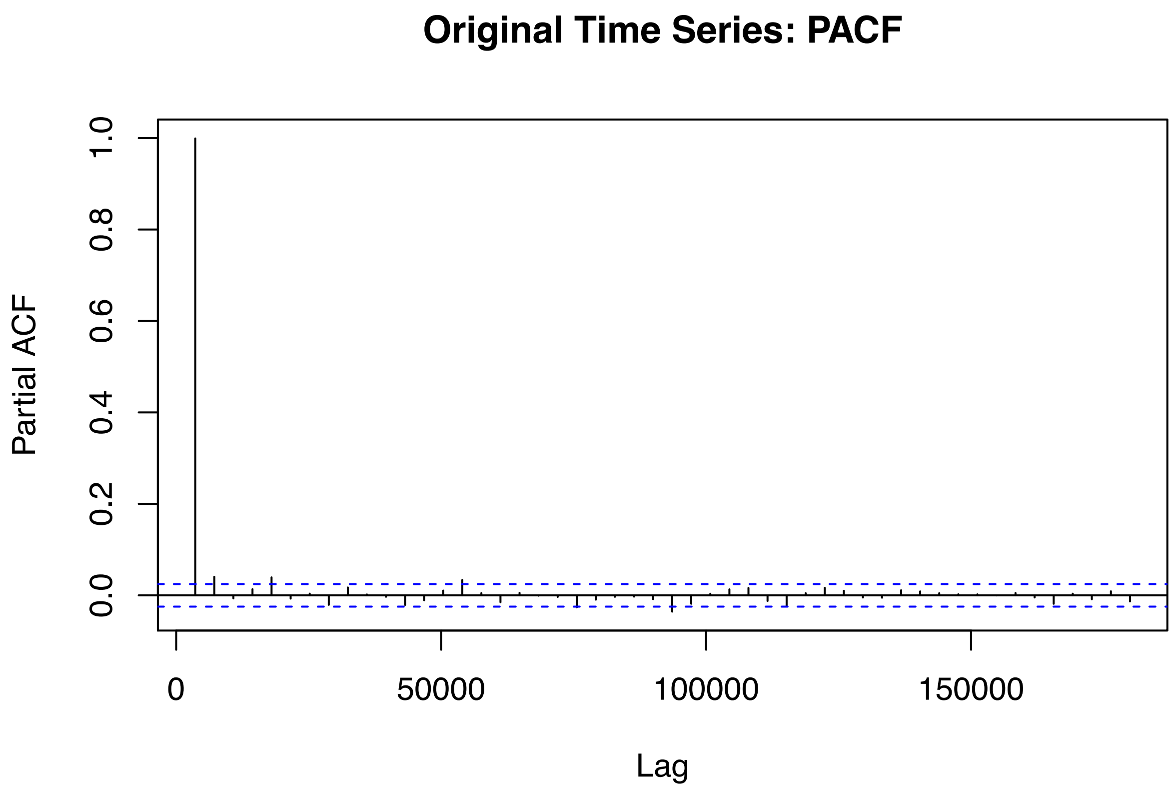


ACF and PACF

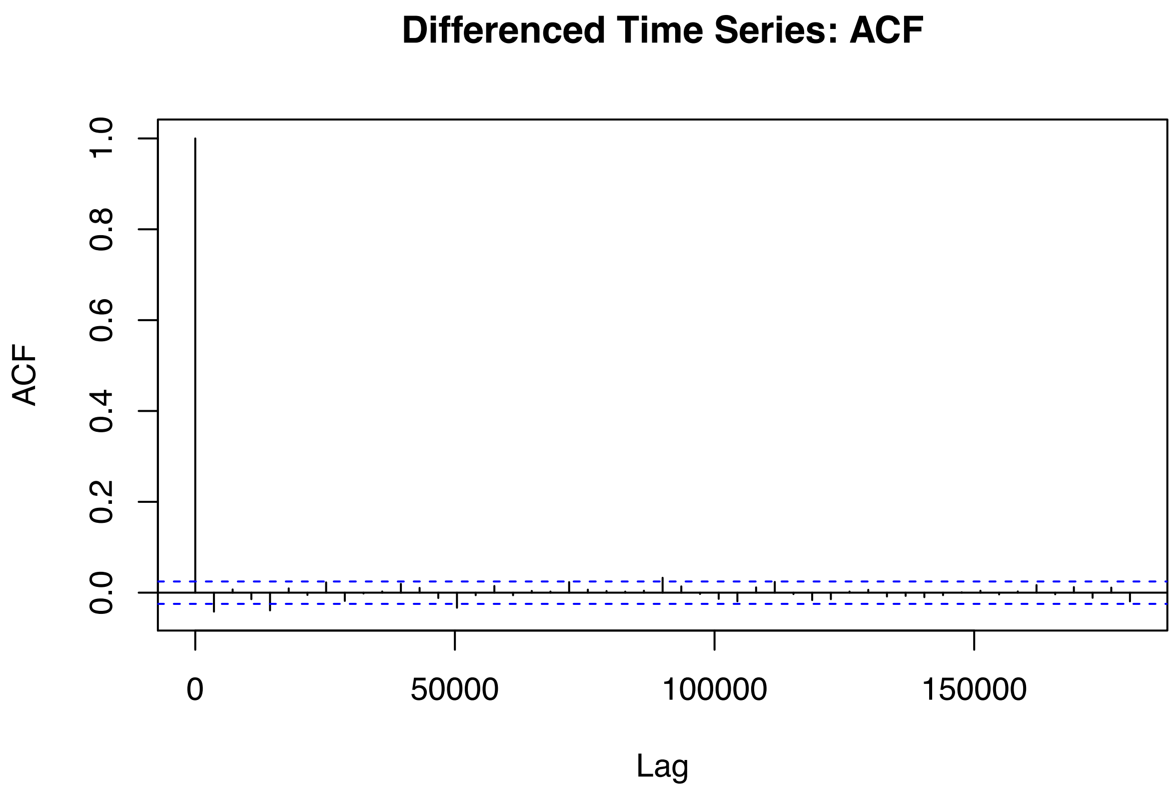
```
acf(data, main='Original Time Series: ACF',lag.max=50)
```



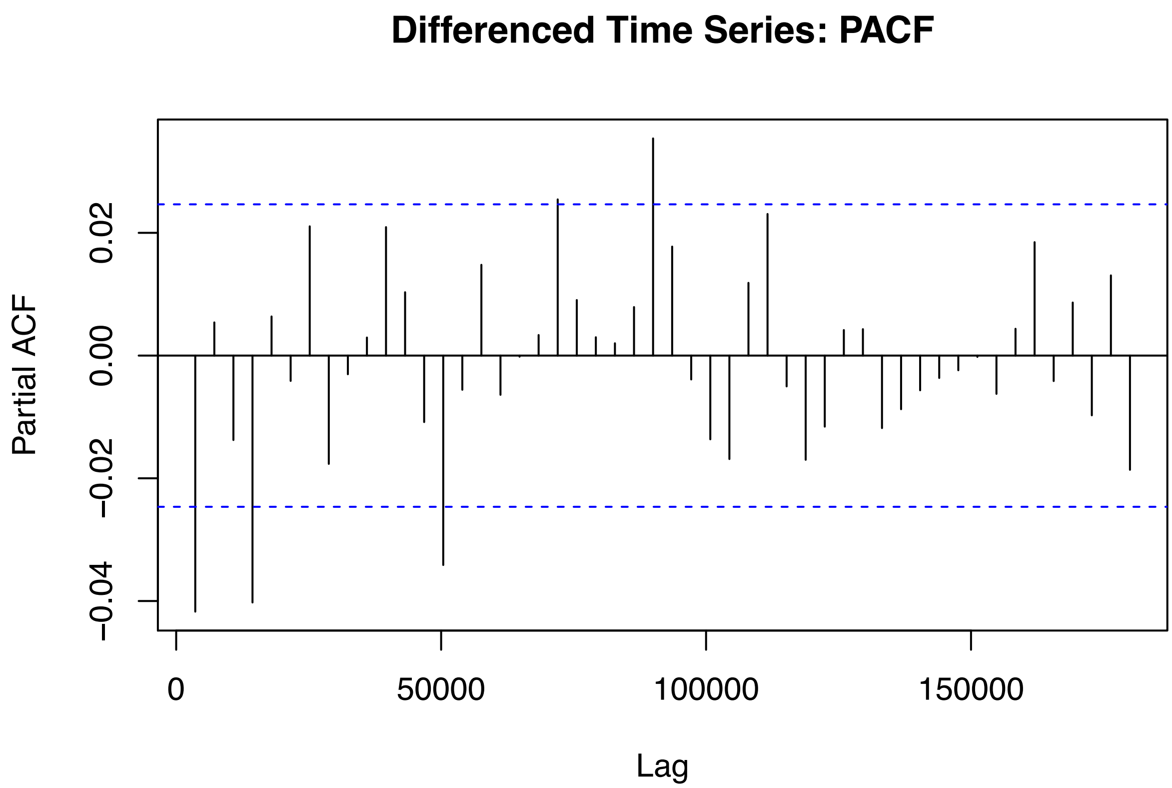

```
pacf(data, main='Original Time Series: PACF',lag.max=50)
```




```
acf(diff.rate, main='Differenced Time Series: ACF',lag.max=50)
```




```
pacf(diff.rate, main='Differenced Time Series: PACF',lag.max=50)
```



2.ARCH+GARCH Order Selection on differenced data, Select model with smallest BIC with itertative method

```
#divide data into data.train and data.test
n = nrow(diff.rate)
data.train= diff.rate[c(1:(n-170))]
data.test= diff.rate[c((n-169):n)]
```

```
# the order selection result is ARMA(0,0)+GARCH(2,1), this procedure is taking so long so I comment thi
# library(rugarch)

# 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(6, 3), include.mean=T),
# distribution.model="std")
# fit = ugarchfit(spec, data.train, solver = 'hybrid')
#    current.bic = infocriteria(fit)[2]
# if (current.bic < final.bic){
# final.bic = current.bic
#     final.order = c(m, n) # the result is c(1,2)
```

```r
# }}
#
# #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(1,2)),
# mean.model=list(armaOrder=c(p, q), include.mean=T),
# distribution.model="std")
# fit = ugarchfit(spec, data.train, solver = 'hybrid')
# current.bic = infocriteria(fit)[2]
# if (current.bic < final.bic){
#    final.bic = current.bic
#    final.order.arma = c(p, q) # the result is c(0,0)
# }
# }
#
# final.bic = Inf
# final.order.garch = 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(final.order.arma[1], final.order.arma[2]),
#      include.mean=T), distribution.model="std")
#      fit = ugarchfit(spec, data.train, solver = 'hybrid')
#      current.bic = infocriteria(fit)[2]
#      if (current.bic < final.bic){
#    final.bic = current.bic
#    final.order.garch = c(m, n)
#      }
# }
```

ARMA+GARCH: Compare Goodness of Fit

```r
# model 1: ARMA(6,3)+GARCH(2,1)
spec.1 = ugarchspec(variance.model=list(garchOrder=c(1,2)),
  mean.model=list(armaOrder=c(6,3), include.mean=T), distribution.model="std")
  final.model.1 = ugarchfit(spec.1, data.train, solver = 'hybrid')
# model 2: ARMA(0,0)+GARCH(2,1)
spec.2 = ugarchspec(variance.model=list(garchOrder=c(1,2)),
  mean.model=list(armaOrder=c(0,0), include.mean=T), distribution.model="std")
  final.model.2 = ugarchfit(spec.2, data.train, solver = 'hybrid')
#Compare Information Criteria
infocriteria(final.model.1)
```

```
##
## Akaike       -8.153892
## Bayes        -8.137500
## Shibata      -8.153903
## Hannan-Quinn -8.148207
```

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

```
##
## Akaike       -8.154069
## Bayes        -8.147512
## Shibata      -8.154071
## Hannan-Quinn -8.151795
```

Not big difference in information criteria.

3.Do forecasting and compare performance

```r
nfore = length(data.test)
fore.series.1 = NULL
fore.sigma.1 = NULL
for(f in 1: nfore){
    data = data.train
    if(f>2){
      data = c(data.train,data.test[1:(f-1)])
      final.model.1 = ugarchfit(spec.1, data, 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)}
}
fore.series.1[!is.finite(fore.series.1)]=NaN
fore.series.1 = na.fill(fore.series.1,"extend")
fore.series.1=c(fore.series.1,c(0,0))
fore.sigma.1=c(fore.sigma.1,c(0,0))

fore.series.2 = NULL
fore.sigma.2 = NULL
for(f in 1: nfore){
    data = data.train
    if(f>2)
data = c(data.train,data.test[1:(f-1)])
final.model.2 = ugarchfit(spec.2, data, solver = 'hybrid')
fore = ugarchforecast(final.model.2, n.ahead=1)
fore.series.2 = c(fore.series.2, fore@forecast$seriesFor)
fore.sigma.2 = c(fore.sigma.2, fore@forecast$sigmaFor)
}
fore.series.2[!is.finite(fore.series.2)]=NaN
fore.series.2 = na.fill(fore.series.2,"extend")
```

Prediction Accuracy, 4 different criteria. Generally it shows model 2 (ARMA(0,0)+GARCH(2,1)) is better

```r
# Mean Squared Prediction Error (MSPE)
mean((fore.series.1-data.test)^2)
```

```
## [1] 1.117046e-05
```

```r
mean((fore.series.2-data.test)^2)
```

```
## [1] 1.113547e-05
```

```r
# Mean Absolute Prediction Error (MAE)
mean(abs(fore.series.1-data.test))
```

```
## [1] 0.002298531
```

```r
mean(abs(fore.series.2-data.test))
```

```
## [1] 0.002290316
```

```r
# Mean Absolute Percentage Error (MAPE)
mean(abs(fore.series.1-data.test)/(data.test+0.000001))
```

```
## [1] 5.857638
```

```r
mean(abs(fore.series.2-data.test)/(data.test+0.000001))
```

```
## [1] 1.734555
```

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

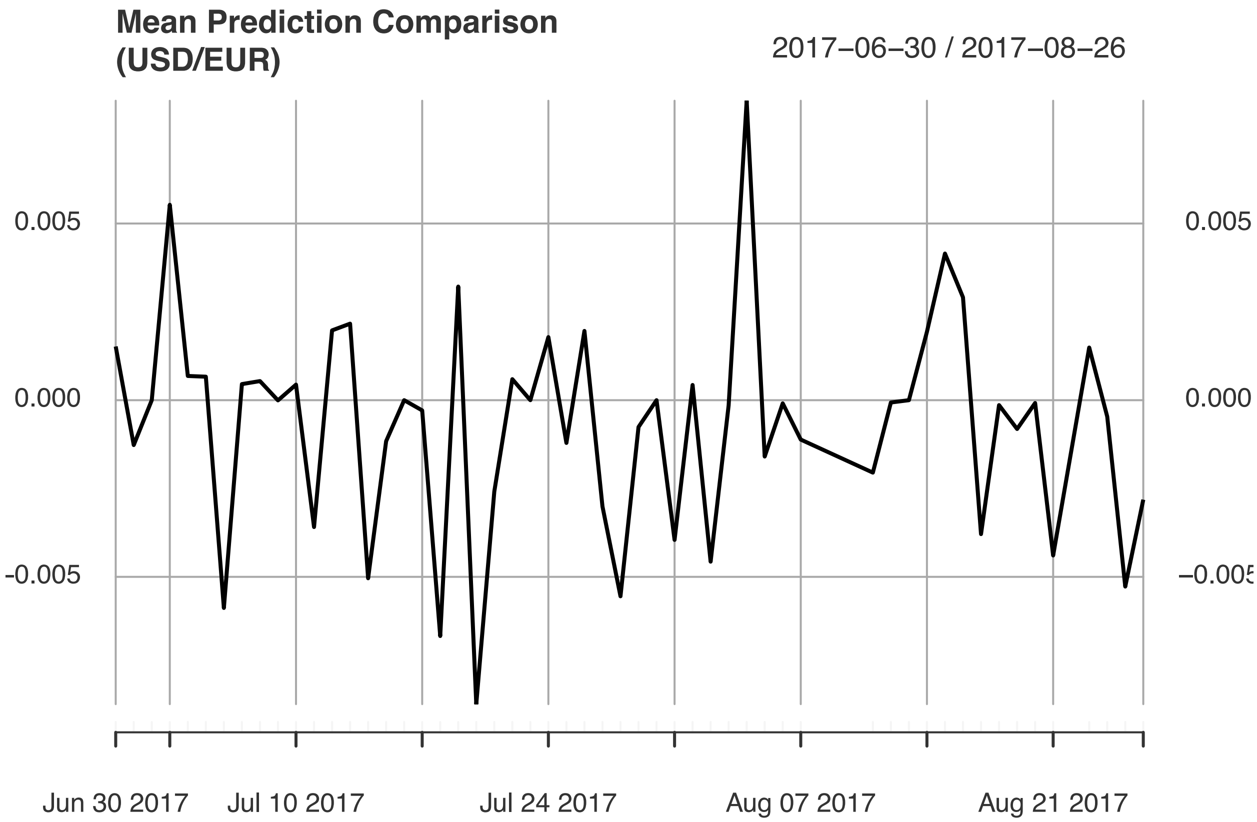
```
## [1] 1.040523
```

```r
sum((fore.series.2-data.test)^2)/sum((data.test-mean(data.test))^2)
```

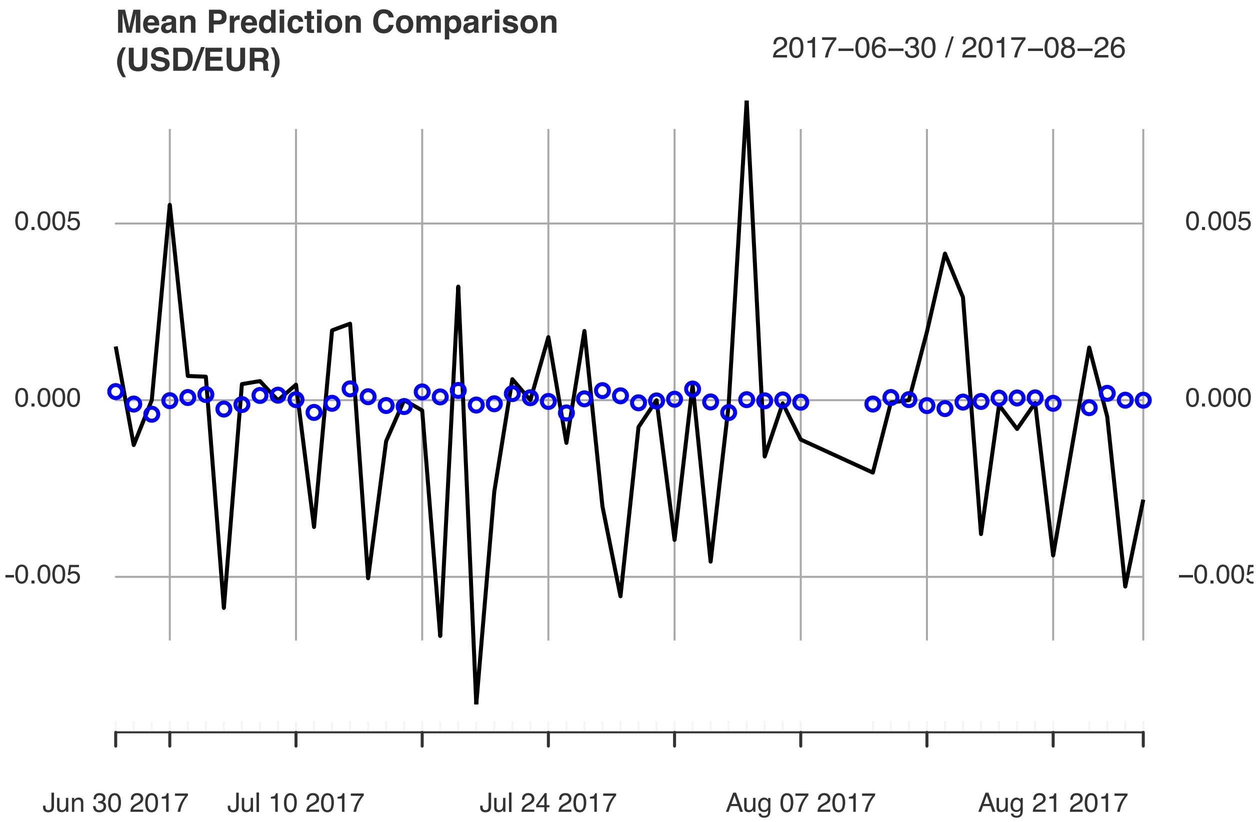
```
## [1] 1.037265
```

Mean Prediction Comparison

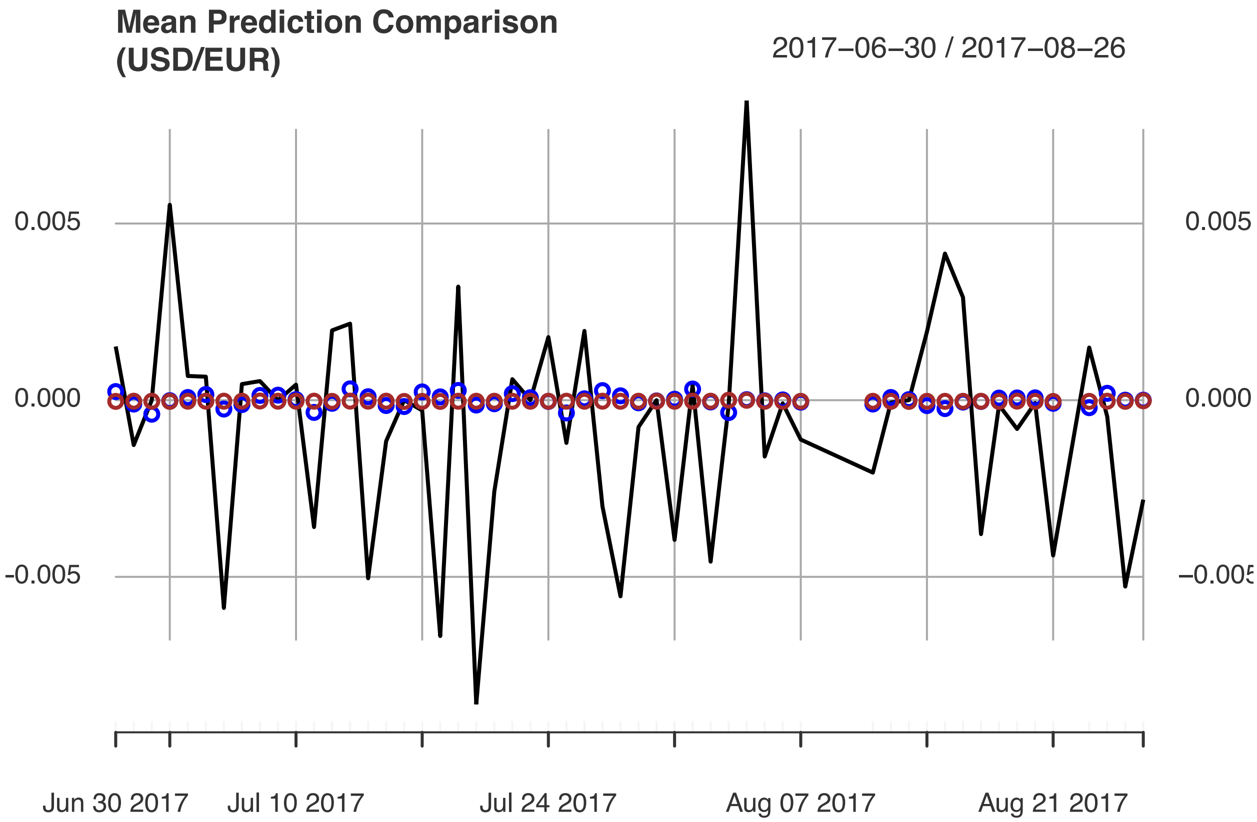
```r
n = nrow(diff.rate)
# ymin = min(c(as.vector(data.test),fore.series.1,fore.series.2))
# ymax = max(c(as.vector(data.test),fore.series.1,fore.series.2))
data.plot = data.test
names(data.plot)="Fore"
plot(diff.rate[c((n-53):n),1],type="l",
     #ylim=c(ymin,ymax),
     xlab=" ",
ylab="USD/EUR Exchange Rate",main="Mean Prediction Comparison
(USD/EUR)")
```




```
data.plot$Fore=fore.series.1
points(data.plot,lwd= 2, col="blue")
```

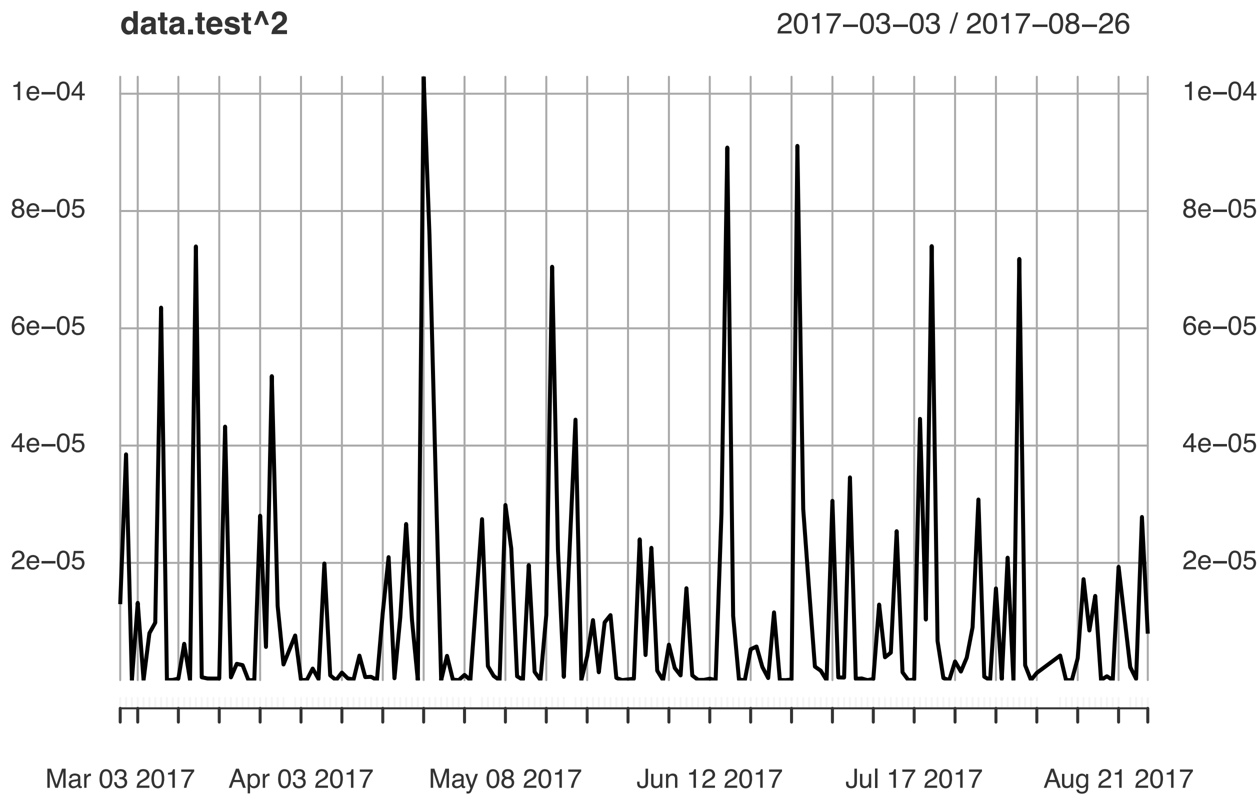



```
data.plot$Fore=fore.series.2
points(data.plot,lwd= 2, col="brown")
```

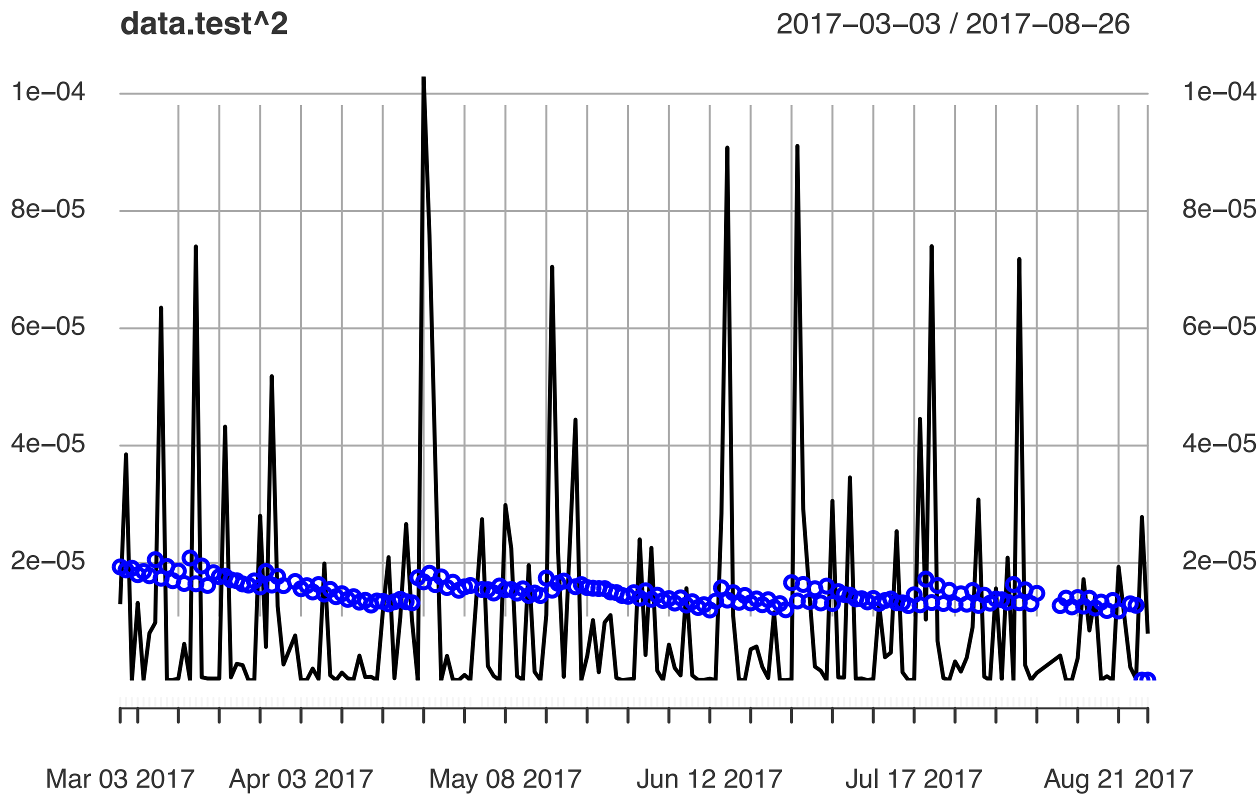


Compare squared observed time series with variance forecasts

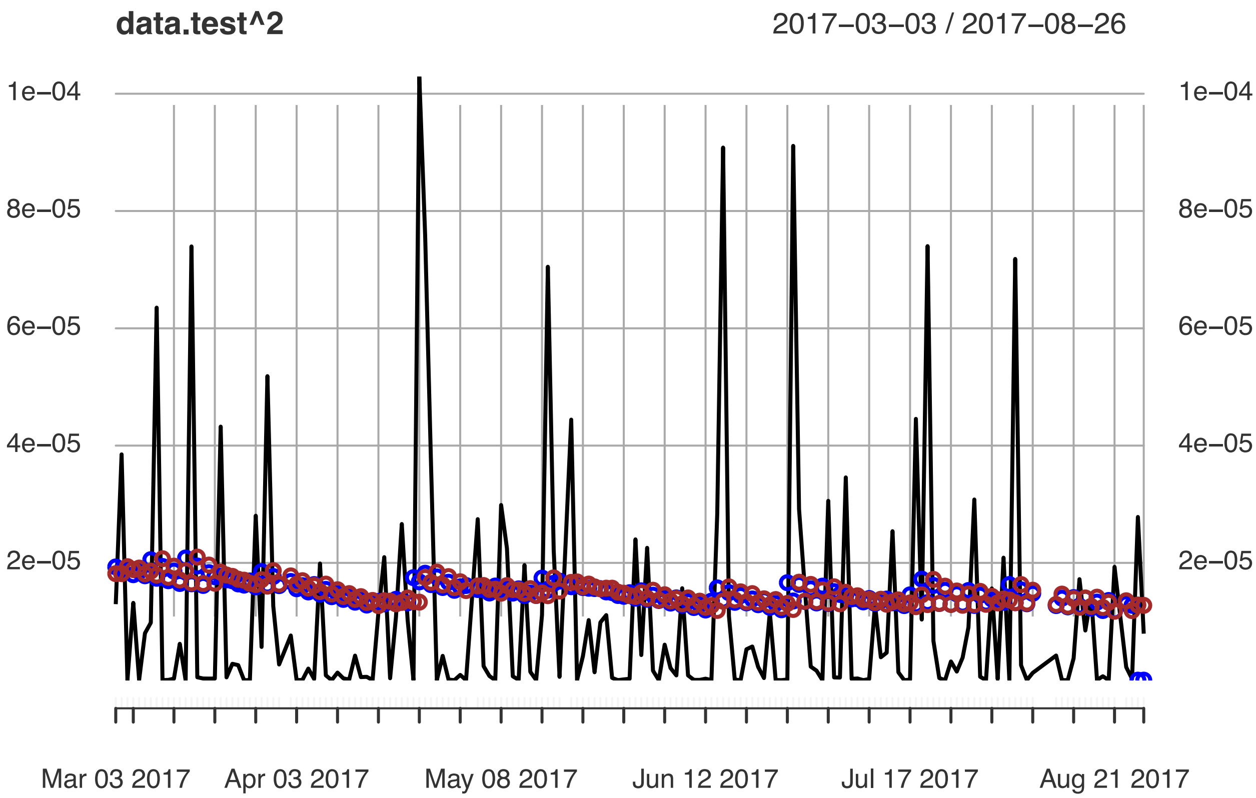
```
#ymin = min(c(as.vector(data.test^2),fore.sigma.1^2,fore.sigma.2^2))
#ymax = max(c(as.vector(data.test^2),fore.sigma.1^2,fore.sigma.2^2))
plot(data.test^2,type="l",
      xlab=" ", ylab="USD/EUR Exchange Rate")
```




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




```
data.plot$Fore=fore.sigma.2^2
points(data.plot,lwd= 2, col="brown")
```



Variance Prediction Comparison with advanced models