

THE UNIVERSITY OF CHICAGO
Graduate School of Business
Business 41202, Spring Quarter 2008, Mr. Ruey S. Tsay

Solutions to Homework Assignment #5

1. For your information, I attached the R commands I used in this answer. You do not need to include any R commands in your solution, however.

Summary table:

	s0	s1	s2	s3	s5	s6
mean	1.217	1.615	1.270	4.467	1.285	2.342
median	0.800	1.137	1.015	3.575	1.023	1.869
max	17.560	15.237	7.135	24.981	8.398	15.140
min	0.000	0.000	0.210	0.737	0.155	0.281

```
> x=read.table("d-ibmohlc0008.txt",header=T)
> x[1,]
  Mon day year  Open  High  Low Close  Volume AdjClose
1   5   9 2008 124.37 124.65 123.63 124.06 5502900   124.06
% reverse the index order.
> T = nrow(x)
> y=x
> for (i in 1:T){
+ ii=T-i+1
+ y[i,]=x[ii,]
+ }
> y[1,]
  Mon day year  Open High  Low Close  Volume AdjClose
1   1   3 2000 112.44  116 111.87   116 10347700   107.43

> ct=y[,7]
> ot=y[,4]
> ht=y[,5]
> lt=y[,6]

> s0=diff(ct)^2
> f=(24-6.5)/24
> s1=(ot[2:T]-ct[1:(T-1)])^2/(2*f)+(ct[2:T]-ot[2:T])^2/(2*(1-f))
> s2=0.3607*(ht-lt)^2
```

```

> s3=0.17*(ot[2:T]-ct[1:(T-1)])^2/f + 0.83*(ht[2:T]-lt[2:T])^2/((1-f)*log(2))
> s5 =0.5*(ht-lt)^2-(2*log(2)-1)*(ct-ot)^2
> s6=0.12*(ot[2:T]-ct[1:(T-1)])^2/f + 0.88*s5[2:T]/(1-f)

% Since s2 and s5 do not involve any lag variable, they have T data points.
% To compare with other methods, select the proper time index (from 2 to T).
> ss=cbind(s0,s1,s2[2:T],s3,s5[2:T],s6)
> ss=sqrt(ss)
% If you don't know the command 'apply', you can use 'mean', 'median', etc.
> apply(ss,2,mean)
      s0      s1      s3      s6
1.217065 1.615238 1.270092 4.467104 1.284646 2.341791
> apply(ss,2,median)
      s0      s1      s3      s6
0.800000 1.136500 1.014985 3.574586 1.022589 1.869095
> apply(ss,2,max)
      s0      s1      s3      s6
17.560000 15.237035 7.134927 24.981124 8.398219 15.139671
> apply(ss,2,min)
      s0      s1      s3      s6
0.0000000 0.0000000 0.2102041 0.7367173 0.1553290 0.2814266

```

2. The plot is shown in Figure 1. The mean, median, maximum and minimum are 0.018, 0.015, 0.043, and 0.000, respectively.

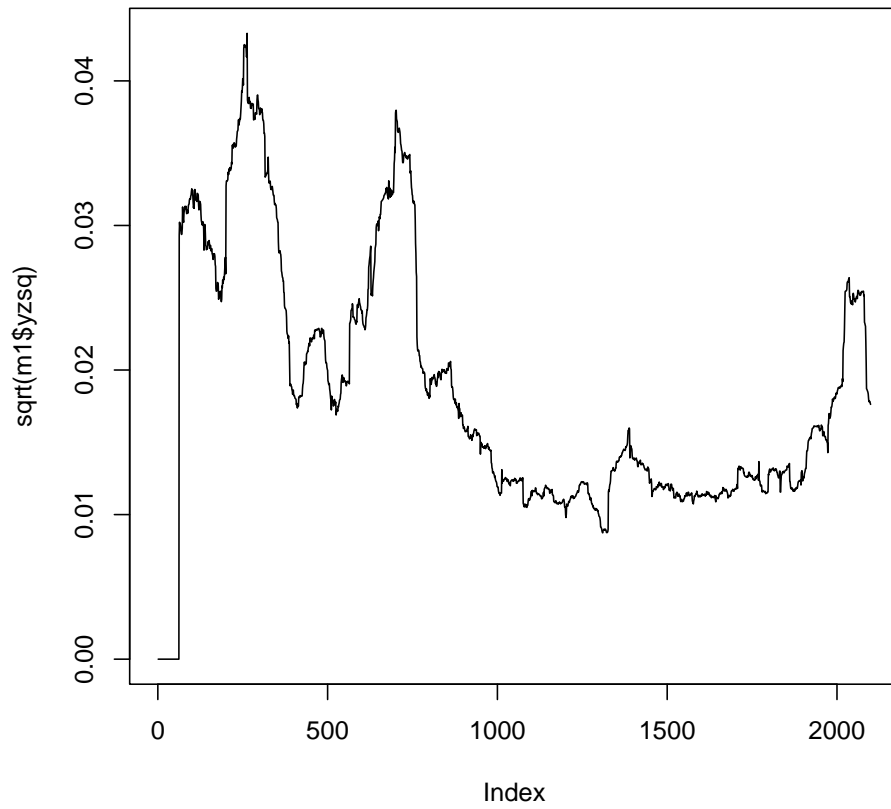
```

> source("r-yz.txt")
> m1=yz(ot,ht,lt,ct)
> plot(sqrt(m1$yzsq),type='l')
> v1=sqrt(m1$yzsq)
> mean(v1)
[1] 0.01837832
> median(v1)
[1] 0.01540002
> max(v1)
[1] 0.04328757
> min(v1)
[1] 0

```

3. The monthly U.S. unemployment rates.

Figure 1: Time plot of daily volatility based on the Yang and Zhang method



- (a) ARIMA(2, 1, 1)(1, 0, 1)₁₂, i.e. regular order c(2,1,1) and seasonal order c(1,0,1) with period 12.

Answer: The model is

$$(1 - 0.57B - 0.23B^2)(1 - 0.56B^{12})x_t = (1 - 0.58B)(1 - 0.82B^{12})a_t, \quad \sigma_a^2 = 0.0369.$$

The model fits the data well. For the 1-step ahead out-of-sample forecasts, the root mean squares of forecast errors is 0.122.

- (b) Your own choice of an AR model.

Answer: I used a seasonal AR model

$$(1 + 0.005B - 0.210B^2 - 0.187B^3 - 0.107B^4)(1 + 0.1945B^{12} + 0.186B^{24})r_t = a_t.$$

The 1-step ahead out-of-sample forecasts show RMSE = 0.123.

- (c) For the neural network, the 1-step ahead out-of-sample RMSE is 0.164.

```
> x=read.table("m-unrate.txt")
> dim(x)
[1] 722 4
> rate=x[,4]
> m1=arima(rate,order=c(2,1,1),seasonal=list(order=c(1,0,1),period=12))
> m1
```

Call:

```
arima(x = rate, order = c(2, 1, 1), seasonal = list(order = c(1, 0, 1), period = 12))
```

Coefficients:

	ar1	ar2	ma1	sar1	sma1
	0.5674	0.2349	-0.5799	0.5580	-0.8190
s.e.	0.0669	0.0396	0.0623	0.0725	0.0531

sigma² estimated as 0.03693: log likelihood = 164.49, aic = -316.98

```
> tsdiag(m1,gof.lag=24) % The model fits the data well.
```

```
> source("r-backtest.txt")
> m2=backtest(m1,rate,600,1)
[1] "RMSE of out-of-sample forecasts"
[1] 0.1223544
```

```
> m3=arima(rate,order=c(4,1,0),seasonal=list(order=c(2,0,0),period=12))
> m3
```

Call:

```
arima(x = rate, order = c(4, 1, 0), seasonal = list(order = c(2, 0, 0), period = 12
```

Coefficients:

	ar1	ar2	ar3	ar4	sar1	sar2
	-0.0045	0.210	0.1868	0.1071	-0.1945	-0.1859
s.e.	0.0372	0.037	0.0366	0.0377	0.0377	0.0402

```
sigma^2 estimated as 0.03806: log likelihood = 154.55, aic = -295.11
```

```
> tsdiag(m3,gof.lag=24)
```

```
> m4=backtest(m3,rate,600,1)
```

```
[1] "RMSE of out-of-sample forecasts"
```

```
[1] 0.1230524
```

```
>
```

```
> rate.x=cbind(rate[24:721],rate[23:720],rate[22:719],rate[13:710],  
rate[12:709],rate[1:698])
```

```
> y=rate[25:722]
```

```
> m5=backnnet(rate.x,y,nsize=2,orig=600,nl=T,nsk=T,miter=3000)
```

```
[1] "RMSE of out-of-sample forecasts"
```

```
[1] 0.1641434
```

4. The US-UK exchange rates. (a) You may use any of the following two models.

Answer: With Gaussian innovation, the GJR(1,1) model is

$$r_t = 0.00024 + a_t, \quad a_t = \sigma_t \epsilon_t, \quad \epsilon_t \sim \text{iid } N(0, 1),$$
$$\sigma_t^2 = 0.04 \times 10^{-4} + (0.04 - 0.012N_{t-1})a_{t-1}^2 + 0.951\sigma_{t-1}^2,$$

where $N_{t-1} = 1$ if $a_{t-1} < 0$, $= 0$, otherwise.

With Student-t innovations, the fitted GJR model is

$$r_t = 0.0003 + a_t, \quad a_t = \sigma_T \epsilon_t, \quad \epsilon_t \sim t_{12.35},$$
$$\sigma_t^2 = 0.002 \times 10^{-4} + (0.04 - 0.015N_{t-1})a_{t-1}^2 + 0.958\sigma_{t-1}^2.$$

For both models, the standardized residuals and their squared series fail to reject the models.

- (b) Is the leverage parameter significantly different from zero at the 5% level?

Answer: No, the leverage parameter is not statistically significant at the 5% level.

5. Again, consider the log returns of USUK exchange rates of Problem 4. (a) The sum of squares of the residuals for the 5-2-1 network with direct link is 0.04164 whereas that of the network without the direct link is 0.04177. The difference is small.

- (b) The percentage of hits is 51.3%..

```

> x=read.table("d-usuk0107.txt")
> dim(x)
[1] 1588    4
> rt=diff(log(x[,4]))
> y=rt[6:1587]
> xx=cbind(rt[5:1586],rt[4:1585],rt[3:1584],rt[2:1583],rt[1:1582])
> m6=nnet(xx,y,size=2,linout=T,maxit=3000)
# weights:  15
initial  value 240.116674
final    value 0.041771
converged
> yhat=predict(m6,xx)
> sum((y-yhat)^2)
[1] 0.04177070
> m6=nnet(xx,y,size=2,linout=T,skip=T,maxit=3000)
# weights:  20
initial  value 277.525453
.....
iter   20 value 0.041638
final  value 0.041638
converged
> yd=ifelse(y>0,1,0)
> m7=nnet(xx,yd,size=2,linout=F,skip=T,maxit=3000)
# weights:  20
initial  value 395.901371
iter   10 value 394.693774
final  value 394.684033
converged

> ydhat=ifelse(m7$fitted.values>0.5,1,0)
> length(yd)
[1] 1582
> icnt=0
> for (i in 1:1582){
+ if(yd[i]==ydhat[i])icnt=icnt+1
+ }
> icnt
[1] 811
> icnt/length(yd)
[1] 0.5126422

```