Assignment 2

We'll look at *GARCH* models in this analysis. GARCH models describe financial markets in which volatility can change, becoming more volatile during periods of financial crises or world events and less volatile during periods of relative calm and steady economic growth.

With 5000 observations, we design an AR (1) - GARCH (1, 1) model:

$$y_t = 0.0078 + 0.0290 y_{t-1} + \varepsilon_t$$

$$\varepsilon_t = \sigma_t z_t \sim N(0,1)$$

$$\sigma_t^2 = 0.13 + 0.036542\,(\varepsilon_{t-1})^2 + 0.90(\sigma_{\tau-1})^2$$

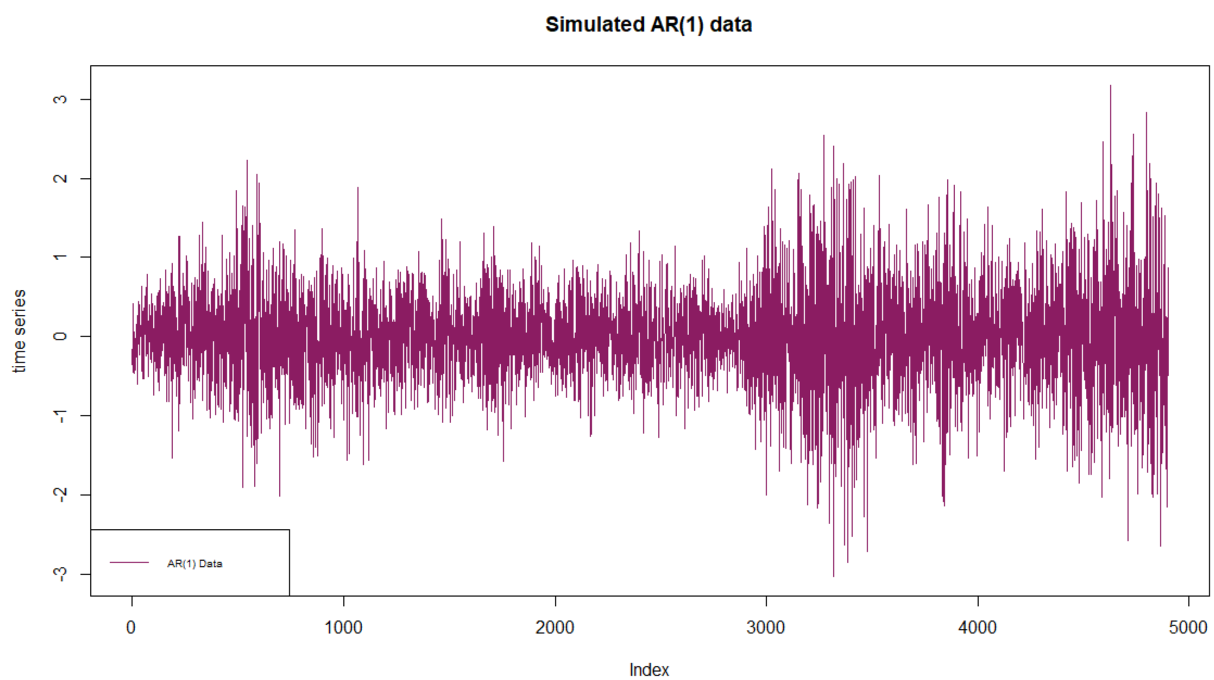

Figure 1.Simulated AR (1) Data

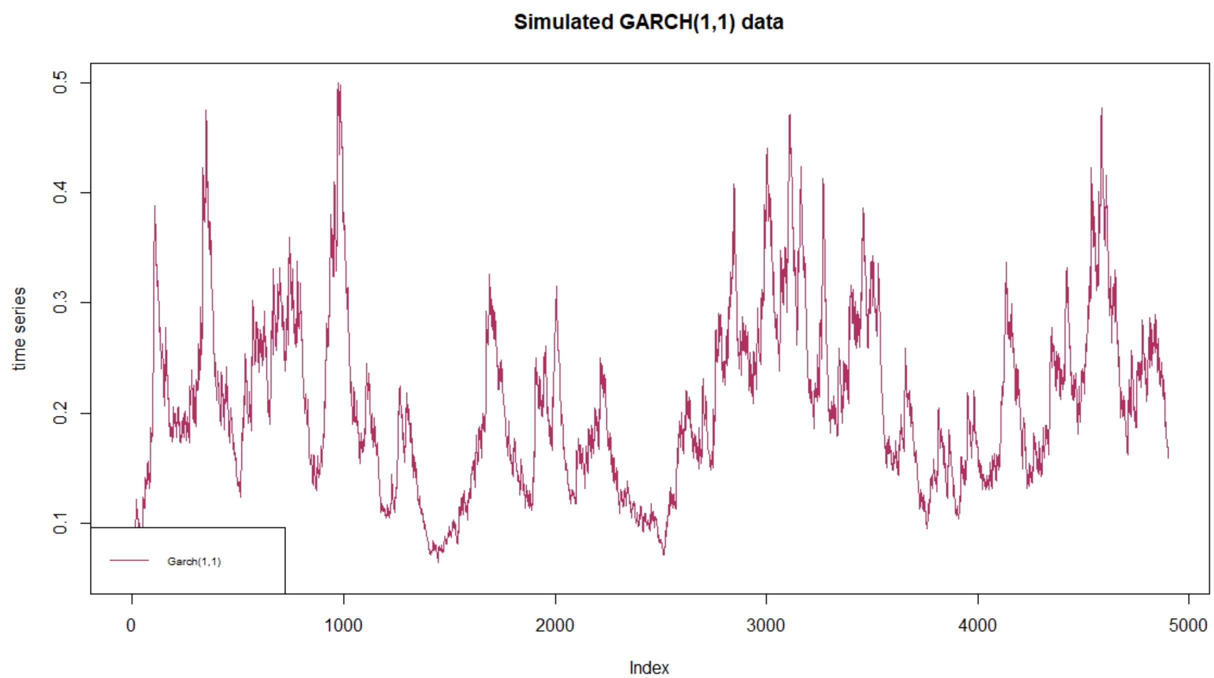

Figure 2.Simulated GARCH(1,1) Data

Table 1.GARCH Maximum Likelihood Coefficients

0	0.04325531	0.001	0.03736236	0.96085956

Table 1 shows that the coefficients estimated by maximum likelihood are very comparable to those used by the original model, with the exception of the second coefficient, which differs slightly. Var was then calculated using a confidence level of 1%. The desired outcome is for Var to be negative, as this indicates that the portofolio is more likely to be profitable. The estimated Var is **-2.247895**.

EGARCH

There is a stylized fact that the *EGARCH* model captures that is not contemplated by the GARCH model, which is the empirically observed fact that negative shocks at time t-1 have a stronger impact in the variance at time t than positive shocks. An EGARCH model is a dynamic model that addresses conditional heteroscedasticity, or volatility clustering, in an innovations process. Volatility clustering occurs when an innovations process does not exhibit significant autocorrelation, but the variance of the process changes with time.

Table 2.EGARCH Maximum Likelihood Confidents

-0.0023370	0.0310264	-0.8177146	0.1000000	-0.0430736	0.3354930

Table 2 shows that the coefficients deviate from the original model. Var was then calculated using a confidence level of 1%. The desired outcome is for Var to be negative, as this indicates that the portofolio is more likely to be profitable. The estimated Var is **-1.430673**

GJR-GARCH

The *GJR* function returns a GJR object specifying the functional form of a GJR(P,Q) model, and stores its parameter values. The key components of a gjr model include the: GARCH polynomial, which is composed of lagged conditional variances. According to research (Laurent et al. and Brownlees et al.) the GJR models generally perform better than the GARCH specification. Thus, including a leverage effect leads to enhanced forecasting performance.

Table 3.GJR Maximum Likelihood Coefficients

0.022272	0.002015	0.044023	0.95558	-0.0102

Table 3 shows that the coefficients deviate from the original model. Var was then calculated using a confidence level of 1%. The desired outcome is for Var to be negative, as this indicates that the portofolio is more likely to be profitable. The estimated Var is **-1.550707**.

CODE

First function

```
autocorr_1<- function(y,l) {
  T<-length(y)
  mm<-mean(y)
  yy<- y-matrix(rep(mm,T),T)
  v<-matrix(yy[c((l+1):T)],T-l)
  v_l<- matrix(yy[1:(T-l)],T-l)
  g_l<-crossprod(v,v_l)/(T-l)
  g_0<-var(y)
  rho_l<- g_l/g_0
  w<- list(rho_l,g_l)
  return(w)
}


ols_1<-function(y,x) {
  #Inverse matrix of x'x
  z1<-solve(t(x)%*%x)
  b<- z1%*%(crossprod(x,y))
  e<- y-x%*%b
  T<-length(e)
  k<- ncol(x)
  ss<- c(crossprod(e,e)/(T-k))
  s1<-z1*ss
  std<-matrix(sqrt(diag(s1)),k)
```

```
  w<-list(b,std)
  return(w)}

likel_ar1<- function(theta,y){
  T<-length(y);
  lik<-0;
  for (l in 2:T) {
    lik<-lik +log(norm_pdf1(y[l],theta[1]+theta[2]*y[l-1],theta[3]))

  }
  lik1=-lik;
  return(lik1)
}

norm_pdf1<- function(x,m,ss) {
  #normal pdf
  f1<- ((2*pi*ss)^(-1/2))
  f2<- exp(-((x-m)^2)/(2*ss))
  f<-f1*f2;
  return(f)
}

log_norm_pdf1<- function(x,m,ss) {
  #normal pdf
  f1<-(-1/2)*log(ss);
```

```
  f2= -((x-m)^2)/(2*ss);
  f3= -(1/2)*log(2*pi);
  f<- f1+f2+f3;
  return(f)
}



likel_ar2<- function(theta,y){
  T<-length(y);
  lik<-0;
  for (l in 3:T) {
    lik<-lik +log(norm_pdf1(y[l],theta[1]+theta[2]*y[l-1]+theta[3]*y[l-2],theta[4]))

  }
  lik2=-lik;
  return(lik2)
}


likel_ar3<- function(theta,y){
  T<-length(y);
  lik<-0;
  for (l in 4:T) {
```

```
    lik<-lik +log(norm_pdf1(y[l],theta[1]+theta[2]*y[l-1]+theta[3]*y[l-2]+theta[4]*y[l-
3],theta[5]))


  }
  lik3=-lik;
  return(lik3)
}
```

Second Function

```
source("manual_functions.R")
likel_ar1_garch = function(theta, y){
  T = length(y);
  ss=matrix(0,T,1);
  ss[2]=var( y );
  lik=0;
  #cat(theta)
  for (l in 2:T)
  {
    er[l]= y[l]- theta[1]-theta[2]*y[l-1];
    #lik=lik+log( norm_pdf1( y[l],theta[1]+theta[2]*y[l-1], abs(ss[l]) ) )
    lik=lik+log_norm_pdf1( y[l],theta[1]+theta[2]*y[l-1], abs(ss[l]) )
    ss[l+1]=theta[3]+theta[4]*(er[l]^2)+theta[5]*ss[l];
  }
  lik1=-lik;
  return(lik1)
}
```

```
ar1arch1 <- function(yy) {
  T = length(yy);
  y=matrix( yy[2:T],T-1)
  xo=matrix(yy[1:T-1],T-1)
  c=matrix( rep(1,T-1),T-1)
  x1=cbind(c,xo)
  q1=ols_1(y,x1);
  z=matrix(unlist(q1[1]),2,1)
  er = y-x1%*%z;
  er2=er^2;
  T1 = length(er2);
  w=matrix( er2[2:T1],T1-1);
  wo=matrix(er2[1:T1-1],T1-1);
  wc=matrix( rep(1,T1-1),T1-1);
  w1=cbind(wc,wo);
  q2=ols_1(w,w1);
  z2=matrix(unlist(q2[1]),2,1)
  w=rbind(z,z2);
  return(w)
}


for_ar1_garch = function(theta, y,p){
  T = length(y);
```

```
  ss=matrix(0,T+1,1);
  m = matrix(0,T+1,1);
  ss[2]=var( y );
  lik=0;
  #cat(theta)
  for (l in 2:T)
  {
    er[l]= y[l]- theta[1]-theta[2]*y[l-1];
    m[l+1]= theta[1]+theta[2]*y[l];
    ss[l+1]=theta[3]+theta[4]*(er[l]^2)+theta[5]*ss[l];
  }
  var=m[T+1]+sqrt(ss[T+1])*qnorm(p)
  return(var)
}



likel_ar1_egarch=function(theta,y)
{
  T=length(y)

  ss=matrix(0,T,1);
  ss[2]=var(y)
  lik=0;

  for(l in 2:T)
```

```
 {
  er[l]=y[l]-theta[1]-theta[2]*y[l-1];

  lik=lik+log_norm_pdf1(y[l],theta[1]+theta[2]*y[l-1],abs(ss[l]))

  ss[l+1]=exp(theta[3]+theta[4]*(log(ss[l]))
              +theta[5]*(er[l]/sqrt(ss[l]))
              +theta[6]*((abs(er[l])/sqrt(ss[l]))-sqrt(2/pi)));

 }
 lik1=-lik;
```

Main Code

```
source('xfunctions.R')
T=5000
yy=matrix(0,T)
er=matrix(0,T)
ss=matrix(0,T)
a0=0.0078
phi=0.0290
c0=0.0013
c1=0.0365
c2=0.9598
for (l in 2:T)
{
  ss[l]=c0+c1*( er[l-1]^2 )+c2*(ss[l-1])
```

```
  yy[l]= a0+phi*yy[l-1] +rnorm(1,mean=0,sd=1)*sqrt(ss[l])
  er[l] = yy[l]-a0-phi*yy[l-1]
}
ss1=matrix( ss[101:T],T-100)
yy1=matrix(yy[101:T],T-100)
plot(yy1,type='l',col="maroon4", main="Simulated AR(1) data",ylab = 'time series' )

legend("bottomleft",lty=1,col=c("maroon4"),
       legend=c("AR(1) Data"),cex = 0.6)
plot(ss1,type='l',col="maroon",main="Simulated GARCH(1,1) data",ylab = 'time series')
legend("bottomleft",lty=1,col=c("maroon"),
       legend=c("Garch(1,1)"),cex = 0.6)

thet0=ar1arch1(yy1)
theta0=rbind(thet0,0);
theta0=c(thet0[1:2],var(yy),0,0);
test=likel_ar1_garch(theta0, yy1)
low=c(0.0,0.0,0.001,0.001,0)
h=c(1,1,1,1,1)
theta_opt<-optim(theta0,likel_ar1_garch, gr = NULL,y=yy1,
                 method = c("L-BFGS-B"),
                 lower = low, upper = h,
                 hessian = TRUE)
theta_opt$par
H =theta_opt$hessian
```

```
invH<- solve(H)

w=H%*%invH

std=sqrt(diag(invH))

# normal inverse

w=qnorm(1/100)

# For VaR use the theta_opt to compute the mu(T+1), ss(T+1) and the normal inverse

VaR=for_ar1_garch(theta_opt$par, yy1,1/100)

VaR

#theta0=c(a0,phi,0.01,0.01,0.01,0.8)

#theta0=c(0,0.1,var(yy1),0.1,0.1)


#egarch

theta2<-c(0,0,log(var(yy)),0,0,0)

test2=likel_ar1_egarch(theta2,yy1)

low2=c(-4,0, -10, -1, -1, 0.0)#(0,0...)

h2 = c(1,1, 10, 0.1, 1, 0.999)


theta_opt2<-optim(theta2,likel_ar1_egarch,gr = NULL,y=yy1,

          method = c("L-BFGS-B"),

          lower = low2, upper = h2,

          hessian = TRUE)

theta_opt2$par



VaR2=for_ar1_egarch(theta_opt2$par,yy1,1/100)
```

VaR2

```
#GJR-GARCH


library(rugarch)

egarchsnp.spec <-
ugarchspec(variance.model=list(model="gjrGARCH",garchOrder=c(1,1)),mean.model=list(armaOrder=c(0,0)))

egarchsnp.fit = ugarchfit(egarchsnp.spec,yy1)

coef(egarchsnp.fit)

library(cvar)

VaR(yy1, p=.99)
```