Xiaotian Zhu

1. Fix the random number generator
   Number is the length of return

```
rng default;
number = 599;
numpath = 8888;
```

2.

theta = [alpha0, alpha1, beta1];

```
theta = [0.0001,0.57,0.23];
Mdl = garch('Constant',theta(1),'ARCH',theta(2),'GARCH',theta(3));
[V,Y] = simulate(Mdl,number,'numpath',numpath);
```

3.set the initial value of thetas, then use garch to create a model
 Use simulate to simulate

```
theta0 = [0.0004,0.3,0.4];
func1 = @(theta)  -likelihood(theta(1),theta(2),theta(3),Y);
options = optimset('LargeScale','off','Display','off');
[thetahat,fval,exitflag,output,lambda,grad,H] =  fmincon(func1,theta0,[0,1,1],1,[],[],[0,0,0],[],[],options);
```

```
function func1 = likelihood(alpha0, beta1,alpha1,Y)
      sigma0 = (1/length(Y)*sum(Y.^2));
      sigma = sqrt(alpha0 + beta1*Y(1)^2 + alpha1*sigma0.^2);
      func1 = log(1/sqrt(2*pi)*sigma)*exp(-Y(1).^2/(2*sigma.^2));
      t = length(Y);

      for i = t
          sigma = sqrt(alpha0 + beta1*Y(i-1).^2 + alpha1*sigma.^2);
          func1 = func1 + log(1/(sqrt(2*pi)*sigma)*exp(-Y(i).^2/(2*sigma.^2)));
      end
end
```

4. Covariance matrix of MLE
   p-value, H0: theta_hat = theta

```
lambda_hat = (H/number)^(-1);
z1=  sqrt(number)*(thetahat(1) - theta(1))/sqrt(lambda_hat(1,1));
z2 = sqrt(number)*(thetahat(2) - theta(2))/sqrt(lambda_hat(2,2));
z3 = sqrt(number)*(thetahat(3) - theta(3))/sqrt(lambda_hat(3,3));
p1 = 2*(1 - normcdf(abs(z1)));
p2 = 2*(1 - normcdf(abs(z2)));
p3 = 2*(1 - normcdf(abs(z3)));
```

5. confidence interval

```
alpha = 0.05;
ub1 = (thetahat(1)) + norminv(1 - alpha/2) * sqrt(lambda_hat(1,1)/number);
lb1 = (thetahat(1)) - norminv(1 - alpha/2) * sqrt(lambda_hat(1,1)/number);
ub2 = (thetahat(2)) + norminv(1 - alpha/2) * sqrt(lambda_hat(2,2)/number);
lb2 = (thetahat(2)) - norminv(1 - alpha/2) * sqrt(lambda_hat(2,2)/number);
ub3 = (thetahat(3)) + norminv(1 - alpha/2) * sqrt(lambda_hat(3,3)/number);
lb3 = (thetahat(3)) - norminv(1 - alpha/2) * sqrt(lambda_hat(3,3)/number);
```

6.

```
fprintf("The confidence interval for alpha0 is : " , lb1, ub1);
fprintf("The confidence interval for alpha1 is : " , lb2, ub2);
fprintf("The confidence interval for beta1 is : " , lb3, ub3);
fprintf("The p-value for alpha0 is : ", p1);
fprintf("The p-value for alpha1 is : ", p2);
fprintf("The p-value for beta1 is : ", p3);
fprintf("The true theta is : ", theta(1),theta(2),theta(3));
fprintf("The estimate theta is : " , thetahat(1),thetahat(2),thetahat(3));
```

7.

```
sizeanalysis = zeros(numpath,3);

for i = 1:numpath
    func1 = @(theta) -likelihood(theta(1),theta(2),theta(3),Y);
    [thetahat,fval,exitflag,output,lambda,grad,H] = fmincon(func1,theta0,[0,1,1],1,[],[],[0,0,0],[],[],options)

     lambda_hat = (H/number)^(-1);
     z1=  sqrt(number)*(thetahat(1) - theta(1))/sqrt(lambda_hat(1,1));
     z2 = sqrt(number)*(thetahat(2) - theta(2))/sqrt(lambda_hat(2,2));
     z3 = sqrt(number)*(thetahat(3) - theta(3))/sqrt(lambda_hat(3,3));
     p1 = 2*(1 - normcdf(abs(z1)));
     p2 = 2*(1 - normcdf(abs(z2)));
     p3 = 2*(1 - normcdf(abs(z3)));

     sizeanalysis(i,:) = [p1 < 0.05, p2 < 0.05, p3 < 0.05];
end
```

```
sizeanalysis = sum(sizeanalysis)/numpath;

%print all results
fprint('The probability of rejecting null hypothesis for alpha0 is: %f\n', sizeanalysis(0));
fprint('The probability of rejecting null hypothesis for alpha1 is %f\n', sizeanalysis(1));
fprint('The probability of rejecting null hypothesis for beta1 is: %f\n', sizeanalysis(2));
```

8. power analysis

```
theta = [0.03,0.25,0.34];
Mdl = garch('Constant',theta(1),'ARCH',theta(2),'GARCH',theta(3));
[V,Y] = simulate(Mdl,number,'numpath',numpath);
theta0 = [0.05,0.31,0.29];
func1 = @(theta)  -likelihood(theta(1),theta(2),theta(3),Y);
options = optimset('LargeScale','off','Display','off');
[thetahat,fval,exitflag,output,lambda,grad,H] =  fmincon(func1,theta0,[0,1,1],1,[],[],[0,0,0],[],[],options);

%Covariance matrix of MLE
lambda_hat = (H/number)^(-1);

%p-value, H0: theta_hat = theta
z1=  sqrt(number)*(thetahat(1) - theta(1))/sqrt(lambda_hat(1,1));
z2 = sqrt(number)*(thetahat(2) - theta(2))/sqrt(lambda_hat(2,2));
z3 = sqrt(number)*(thetahat(3) - theta(3))/sqrt(lambda_hat(3,3));
p1 = 2*(1 - normcdf(abs(z1)));
p2 = 2*(1 - normcdf(abs(z2)));
p3 = 2*(1 - normcdf(abs(z3)));

%confidence interval
alpha = 0.05;
ub1 = (thetahat(1)) + norminv(1 - alpha/2) * sqrt(lambda_hat(1,1)/number);
lb1 = (thetahat(1)) - norminv(1 - alpha/2) * sqrt(lambda_hat(1,1)/number);
ub2 = (thetahat(2)) + norminv(1 - alpha/2) * sqrt(lambda_hat(2,2)/number);
lb2 = (thetahat(2)) - norminv(1 - alpha/2) * sqrt(lambda_hat(2,2)/number);
ub3 = (thetahat(3)) + norminv(1 - alpha/2) * sqrt(lambda_hat(3,3)/number);
lb3 = (thetahat(3)) - norminv(1 - alpha/2) * sqrt(lambda_hat(3,3)/number);
```