FORDHAM | Gabelli School of Business

THE JESUIT UNIVERSITY OF NEW YORK

Financial Econometrics II Final Project

Project 1 (MLE Estimation of the EGARCH Model)

Due Date: March 3, 2020

Team Members: Gleb Dunaev, Eugene Lam, Qiuan Wu, Xiaotian Zhu

Methodology

In this project, we will be using EGARCH to model the conditional volatility for S&P500 index. Previous research shows that long term volatility is more volatile during the economic downturn. Bad news produces stronger effect than good news for the market, especially during the crisis. In the EGARCH model, the conditional variance is an asymmetric function of lagged disturbances. Note that we have to model the log-conditional volatility to ensure volatility is positive.

$$\begin{aligned} y_t &= \sigma_t \varepsilon_t \\ \ln(\sigma_t^2) &= \alpha_0 + \sum_{j=1}^{p} \alpha_j \ln(\sigma_{t-j}^2) + \sum_{i=1}^{q} \beta_i \left| \frac{y_{t-i}}{\sigma_{t-i}} \right| + \gamma_i \left(\frac{y_{t-i}}{\sigma_{t-i}} \right) \end{aligned}$$

The β parameter represents a magnitude effect or the symmetric effect of the model, the "GARCH" effect. α_j measures the persistence in conditional volatility irrespective of anything happening in the market. When α is relatively large, then volatility takes a long time to die out following a crisis in the market.

The γ parameter measures the asymmetry or the leverage effect, the parameter of importance so that the EGARCH model allows for testing of asymmetries. If $\gamma = 0$, then the model is symmetric. When $\gamma < 0$, then positive shocks (good news) generate less volatility than negative shocks (bad news). Our EGARCH (1, 1) model got a distinctive feature, i.e., conditional variance was modeled to capture the leverage effect of volatility.

Density Function

First, we will need to define the density function for EGARCH(p, q):

$$f(y_1, \dots, y_T; \theta) = \prod_{t=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_t} e^{-(y_t)^2/(2\sigma_t^2)}$$

MATLAB code for this function can be seen below:

```
function df = denFun(param, ySeries, pNum ,qNum)
    maximumOfpq = max(pNum,qNum);
    beta = param(pNum+2:end-1);
    alpha = param(2:pNum+1);
    gamma = param(length(param));
    sigm = zeros(length(ySeries),1);
    sigm(1:maximumOfpq) = std(ySeries);
    for j=(maximumOfpq+1):length(ySeries)
        sigm(j) = sqrt(exp(param(1)...
            + sum(log(sigm(j-pNum:j-1).*sigm(j-pNum:j-1)).*alpha)...
            + sum(abs(ySeries(j-qNum:j-1)./sigm(j-qNum:j-1)).*beta)...
            + gamma * ySeries(j-1)/sigm(j-1)));
    end
    df = 1/sqrt(2*pi)./sigm.*exp(-ySeries.^2./(2*(sigm.^2)));
end
```

Log-likelihood Function

Furthermore, we will need to take the log likelihood of the density function.

$$L(\theta) = \log f(y_1, ..., y_T; \theta) = \sum_{t=1}^{T} \left[-\frac{y_t^2}{2\sigma_t^2} - \log \left[\sqrt{2\pi}\sigma_t \right] \right]$$

MATLAB code for log likelihood function can be seen below:

```
function like = likelihood(funct, param, ySeries, pNum, qNum)
    like = -sum(log(funct(param, ySeries, pNum ,qNum)))/(length(ySeries));
end
```

Maximum Likelihood Estimation Function

Later, we also wrote an MLE function:

```
function [paramH, low_CI, up_CI, pVal] = MaxLikelihood(ySeries, paramInitial, signLev, hyp, pNum, qNum)
    [paramH,~,~,~,~,hessian] = fminunc(@(param) likelihood(@denFun, param, ySeries, pNum, qNum),paramInitial);
    A = length(paramH);
    N = length(ySeries);
    variance_mat = inv(hessian);
    sigm_estim = zeros(1,A);
    for j=1:A
        sigm_estim(j) = sqrt(variance_mat(j,j))/sqrt(N);
    end
    z_stat = (paramH-hyp)./sigm_estim;
    pVal = 2*normcdf(-abs(z_stat),0,1);
    L = abs(icdf('Normal',signLev/2,0,1));
    low_CI = hyp - L * sigm_estim;
    up_CI = hyp + L * sigm_estim;
end
```

Size and Power Analysis

First, to do the size analysis, we will simulate data from the EGARCH model which we estimated in the previous step. We set the initial values for the return and sigma as the random numbers from the normal distribution and we calculate the return as $y_t = \sigma_t \varepsilon_t$

We use the following Matlab function to simulate the data:

```
function [ySeries] = generateReturns(pNum,alpha,qNum,beta,gamma,size,const)
    maximumOfpq = max(pNum,qNum);
    ySeries = zeros(size+1000,1);
    sigm = zeros(size+1000,1);
    for j = 1:maximumOfpq
        ySeries(j) = normrnd(0,1);
        sigm(j) = normrnd(0,1);
    end
    for j = (maximumOfpq+1):(size+1000)
        sigm(j) = sqrt(exp(const...
            + sum(log(sigm(j-pNum:j-1).*sigm(j-pNum:j-1)).*alpha)...
            + sum(abs(ySeries(j-qNum:j-1)./sigm(j-qNum:j-1)).*beta)...
            + gamma * ySeries(j-1)/ sigm(j-1)));
        ySeries(j) = sigm(j)*normrnd(0,1);
    end
    ySeries = ySeries(1001:end);
end
```

Then we examine whether this particular model which we used to generate the data is rejected. In order to do that we identify the parameters for the simulated data using the MLE function.

The null hypothesis is that the newly obtained parameters are not statistically significant different from the true parameters of our model. We repeat this procedure 10000 times and calculate in how many iterations the null hypothesis was rejected. We would expect that the number of times when the null was rejected is not greater than the significance level of 5%.

Secondly, to do the power analysis, we need to simulate data from a GARCH model which is different from our model. Then we repeat the procedure we did in the size analysis, however, now we would expect that the number of times when the null was rejected is greater than the significance level of 5%.

Results

Estimated model coefficients

We downloaded the S&P 500 index for the past 5 years (3/2/2015 - 2/28/2019). In addition, we also calculated the log returns of S&P using it's closed prices. To generate the coefficients we used EGARCH(1, 1) model.

First, we identify the parameters for the constant, α, β and γ respectively which maximize the log-likelihood function.

```
Command Window
>> thetaH

thetaH =

   -0.8734    0.9248    0.1894   -0.2426

>>
```

In order to ensure that the coefficients are calculated correctly, we use the built-in Matlab EGARCH function and obtain the following results which match our results:

```
>> estimate(Mdl,R_t,'E0',R_t(1))

    EGARCH(1,1) Conditional Variance Model (Gaussian Distribution):

                    Value      StandardError    TStatistic      PValue
                   --------    -------------    ----------    ----------

    Constant       -0.72258       0.067167        -10.758      5.4357e-27
    GARCH{1}         0.9248      0.0068385         135.23               0
    ARCH{1}         0.18956       0.022265         8.5135      1.6874e-17
    Leverage{1}    -0.24248       0.015205        -15.948       2.937e-57
```

Hypothesis testing

We generate the p-values for each of the coefficients:

```
Command Window
>> p_value

p_value =

   1.0e-10 *

    0.0000         0    0.1065    0.0000

>>
```

We can see that the p-values for each of the coefficients are very close to zero, therefore, the null hypothesis that the coefficients are not statistically significant different from zero is rejected. We can also see that γ < 0 , therefore, positive shocks (good news) generate less volatility than negative shocks (bad news).

We obtain the following EGARCH (1, 1) equation which can be used for prediction:

$$\sigma_t{}^2 = \exp\left[-0.8734 + 0.9248\ln(\sigma_{t-1}{}^2) + 0.1894\left|\frac{y_{t-1}}{\sigma_{t-1}}\right| - 0.2426\left(\frac{y_{t-1}}{\sigma_{t-1}}\right)\right]$$

Size analysis

First, we simulate data from the EGARCH model which we estimated in the previous step. Then we examine whether this particular model which we used to generate the data is rejected. In order to do that we identify the parameters for the simulated data using the MLE function. The null hypothesis is that the newly obtained parameters are not statistically significant different from the true parameters of our model. We repeat this procedure 10000 times and calculate in how many iterations the null hypothesis was rejected. We would expect that the number of times when the null was rejected is not greater than the significance level of 5%.

Having run the size analysis, we obtain the following percentages of null hypothesis rejections for the constant, α, β and γ coefficients respectively and conclude that the model runs adequately.

```
p =

    0.0400
    0.0300
    0.0400
    0.0500

>>
```

Power analysis

We need to simulate data from a GARCH model which is different from our model. Then we repeat the procedure we did in the size analysis, however, now we would expect that the number of times when the null was rejected is greater than the significance level of 5%.

We generate the data using the following parameters [-0.7, 0.82, 0.1, -0.15] for the constant, α, β and γ coefficients respectively and test the null hypothesis that the optimized parameters for the simulated data is not significantly different from the true parameters of our initial model.

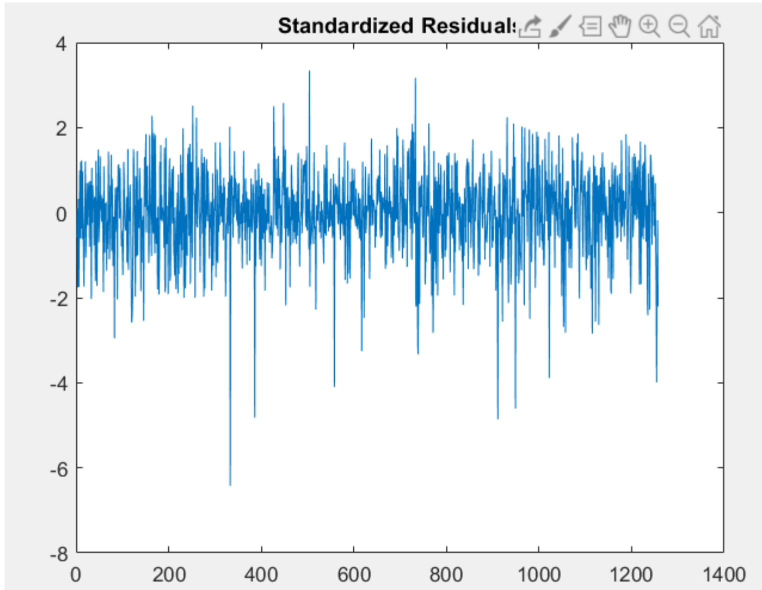
Having run the power analysis, we obtain the following percentages of null hypothesis rejections for the constant, α, β and γ coefficients respectively and conclude that the model runs adequately.

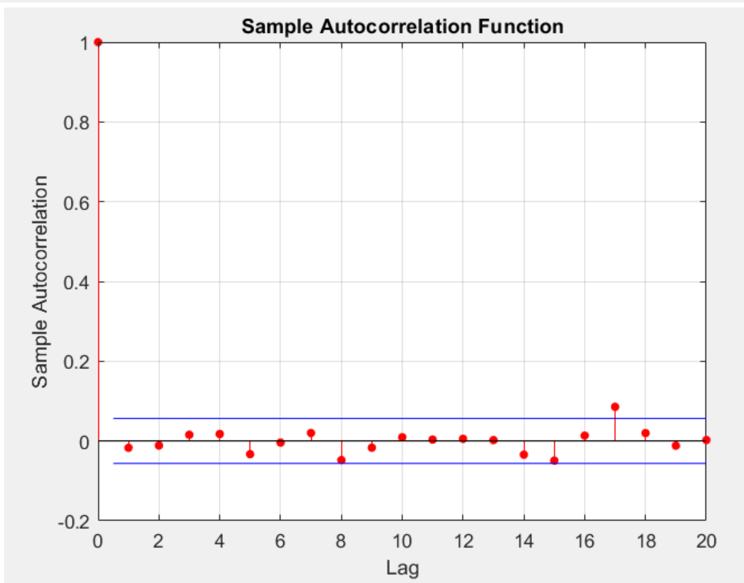
```
p =

    0.3000
    0.6900
    0.4000
    0.6300

>>
```

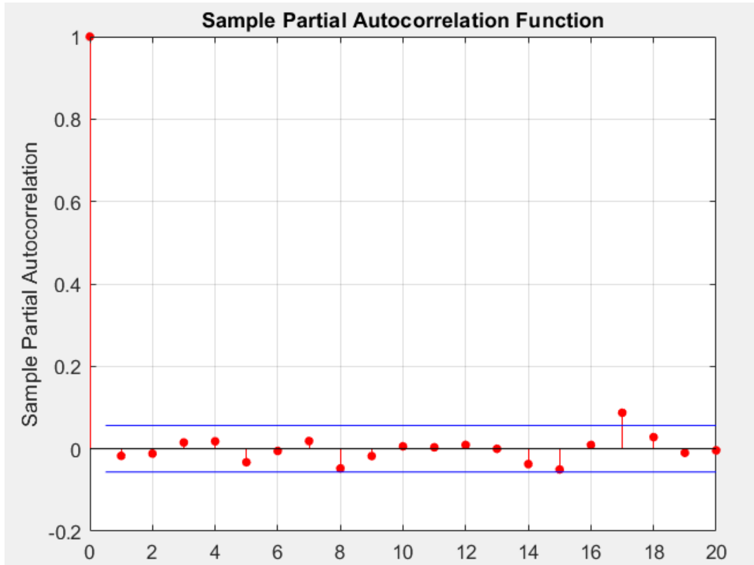
Post Estimation

In order to obtain residual, we need to fit EGARCH(1,1) and find the conditional variance.

```
Mdl = egarch('Offset',NaN','GARCHLags',1,'ARCHLags',1,'LeverageLags',1);
EstMdl = estimate(Mdl,R_t);
v = infer(EstMdl,R_t);
res = (R_t-EstMdl.Offset)./sqrt(v);
plot(res)
title('Standardized Residuals')
autocorr(res)
parcorr(res)
```


Standardized Residuals
4
2
0
-2
-4
-6
-8
0
200
400
600
800
1000
1200
1400


Sample Autocorrelation Function
Sample Autocorrelation
1
0.8
0.6
0.4
0.2
0
-0.2
0
2
4
6
8
10
12
14
16
18
20
Lag


Sample Partial Autocorrelation Function
Sample Partial Autocorrelation
1
0.8
0.6
0.4
0.2
0
-0.2
0
2
4
6
8
10
12
14
16
18
20