STAT 631 Project

Jack Cunningham (jgavc@tamu.edu)

12/10/24

The stock I chose for this project is the Bank of New York Mellon, ticker symbol BK.

Loading data, helper functions and libraries:

```
library(quantmod)
library(forecast)
library(rugarch)
load("garch.RData")
source("GARCH_plotFunctions.R")
source("GARCH_RFunctions.R")
```

Introduction and Stock Performance

The Bank of New York Mellon is a financial services company. It came to be after the Bank of New York, the oldest bank in the US and the largest custodian bank, and Mellon Financial Corporation, known for asset management, merged in 2007. It has many offerings and is less reliant on deal flow than an investment bank (say Goldman Sachs) and less reliant on interest rates than a consumer bank (say Citibank). A large portion of operations are providing technology solutions to asset managers and other banks.

Let's take a look at BK's performance over time:

```
plot(BK[,"BK.Adjusted"])
```

There are a few interesting events to point out.

First, the time series begins at the start of 2009 where the financial industry was working its way out of the great financial crisis, we see the stock struggle until the beginning of 2012 where it has a strong rally for a couple of years.

Second, in 2015 the company hits a snag after their accounting system failed. This particularly affected their mutual fund clients who were unable to provide NAVs to potential investors, and thus lost out on business. This issue persisted over two weeks and called into question the reliability of BK's systems.

After recovering in 2016 the stock continued climbing until 2018. The uncertainty about future Fed policy, an inverted yield curve and multiple poor earnings reports led to a drop of 28% from 2018 until the start of the pandemic.

During the pandemic, after a large immediate drop much like the rest of the market, the stock did very well. One reason may be that as financial companies were working remotely the demand for technology solutions (such as trading software) increased dramatically.

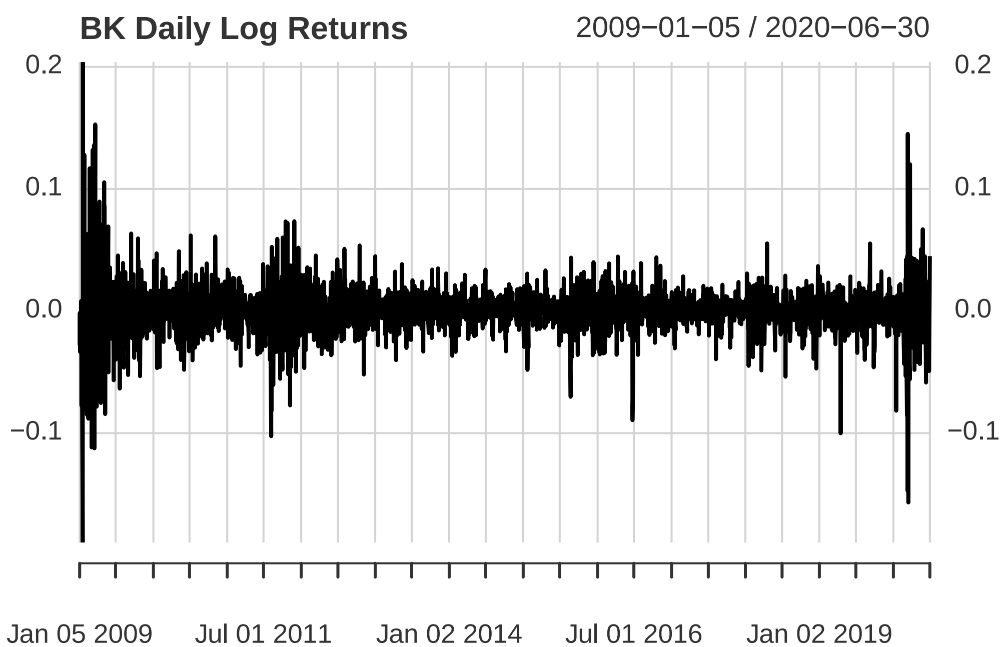
The stock struggled in 2022, due to quick rate hikes from the FED to combat inflation. Although not too much of BNY's business is retail banking, they still have large exposure to interest rates and the inability to predict FED policy and economic outcomes hurt much of the financial industry.

Currently the stock is on a strong run, similar to other banks. However an additional wrinkle helps explain BNY's performance. They have commanded investments to AI at a time where the market has rewarded companies for it. They were the first major bank to deploy an "AI Supercomputer" while collaborating with NVIDIA, they already have a history of providing technology solutions to other banks and being the largest custodian they have access to a lot of data.

Part 1)

Before fitting any ARMA models we should see if the daily log returns are stationary. We can do this with a time series plot.

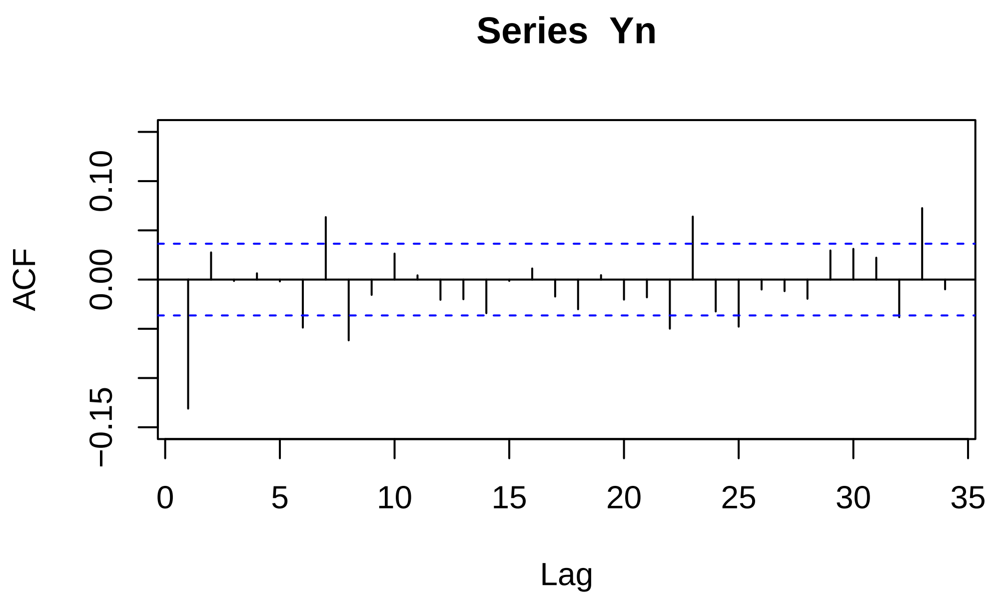
```
plot(Yn, grid.col = "lightgray", main = "BK Daily Log Returns")
```



In this plot we can see oscillation around 0%, reflecting mean reversion. This indicates that modeling this series as a stationary process is a reasonable approach.

We look for evidence of serial correlation. We can examine the auto-correlations individually to understand their structure.

```r
Acf(Yn, ylim = c(-.15,.15))
```



In this plot, test bounds for the null hypothesis that each autocorrelation is zero are drawn. The band is $\pm 1.96/\sqrt{n}$, so in this case $\pm 1.96/\sqrt{2892} = 0.03644$. There is strong autocorrelation at lag 1 and evidence of autocorrelation (although the effect is quite small) in lags 6,7 and 8.

We can also test for white noise, whether all auto correlations are equal to zero. That is, for the null hypothesis:

$$H_0 : \rho_1 = \cdots = \rho_K = 0$$

We do this using the Box-Pierce statistic with the Ljung and Box modification which corrects bias and improves the bias for our purposes. The statistic is:

$$Q(K) = n(n+2)\sum_{l=1}^{K} \frac{\hat{\rho_l}^2}{n-l}$$

This approximates the χ^2_k distribution.

The decision of K is important as it can affect the performance of the statistic, a method to help verify our results is to simply use multiple values of K and compare results.

```
Ks = 2 + 3*(0:5); pv = c()
for(i in Ks) pv = cbind(pv, Box.test(Yn, i , "L")$p.val)
colnames(pv) = paste("K = ", Ks); rownames(pv) = "BK"; round(pv,5)
```

```
   K =  2 K =  5 K =  8 K =  11 K =  14 K =  17
BK      0      0      0       0       0       0
```

We can soundly reject the null hypothesis that Bank of New York Mellon returns are white noise.

With the knowledge that this series can be modeled as a stationary process and evidence of autocorrelation a reasonable approach would be to use an ARMA model.

An ARMA(p, q) model is appropriate when a time series Y_t is defined as:

$$Y_t = c + \phi_1 Y_{t-1} + \cdots + \phi_p + \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q}, \quad c = \mu(1 - \phi_1 - \cdots - \phi_p).$$

This can also be written as:

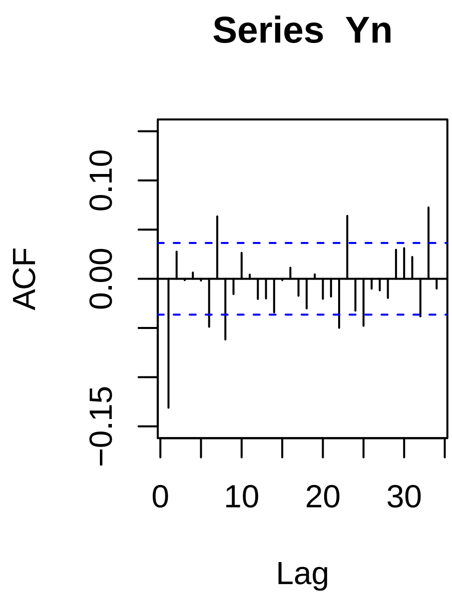
$$\phi(B) x_t = \theta(B) \epsilon_t$$

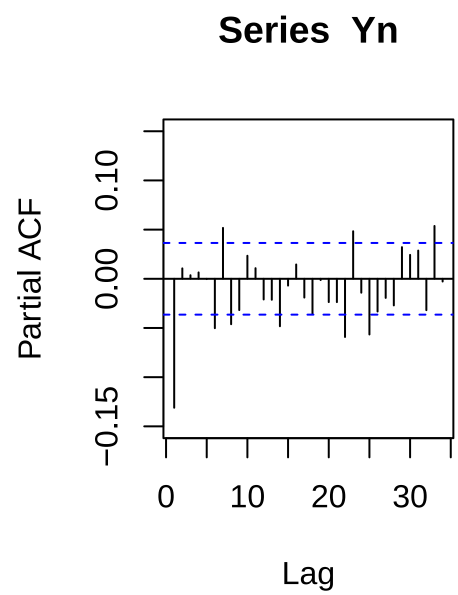
Key things we need in order to assure our series is stationary, causal and invertible, is that both the AR and MA polynomials have roots within the unit circle. And there should be no common factors between the two polynomials (if we had common factors it would add error to the model and hurt our ability to use it effectively).

With the concern of parameter redundancy in mind we do not rush to an auto selection tool, we instead start our model building leveraging auto correlations and partial auto correlations.

The auto-correlations can help guide our selection of p and the sample partial auto-correlations can help guide our selection of q.

```
par(mfrow = c(1,2))
Acf(Yn, ylim = c(-.15,.15))
Pacf(Yn, ylim = c(-.15,.15))
```





From these plots the most clear selection of p and q are 1 and 1 respectively. There is evidence of lags 6-8 of ACF and PACF having non-zero correlation but it is unlikely that one, the effect is significant as we are only touching correlations of about 0.05 and two, we are almost certain to run into parameter redundancy with p/q values being that high.

We continue by considering models with a maximum p of 1 and a maximum q of 1, these would be the reasonable set of models.

```
ps = 0:1; qs = 0:1
AIC = BIC = matrix(nrow = length(ps), ncol = length(qs))
rownames(AIC) = rownames(BIC) = paste0("p = ", ps)
colnames(AIC) = colnames(BIC) = paste0("q = ", qs)
for(i in ps){
  for(j in qs){
    if((i+j)<= 3){
      arma = Arima(Yn, order = c(i,0,j))
      AIC[i+1,j+1] = arma$aic; BIC[i+1,j+1] = arma$bic
    }
  }
}
AIC;
```

```
          q = 0      q = 1
```

```
p = 0 -14191.60 -14237.95
p = 1 -14239.82 -14238.11
```

```
BIC;
```

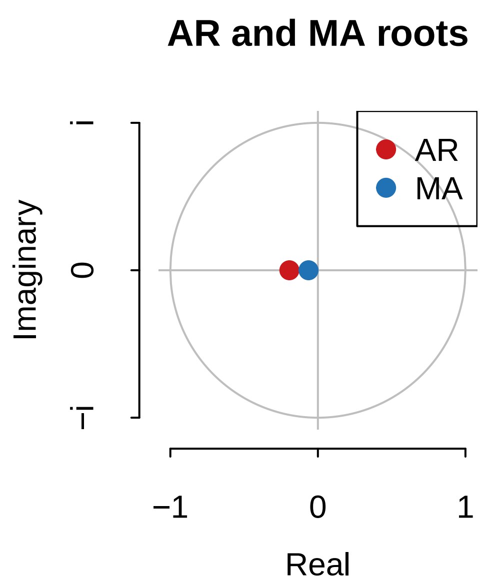
```
           q = 0     q = 1
p = 0 -14179.66 -14220.04
p = 1 -14221.91 -14214.23
```

The AIC criterion prefers the candidate models in the following order: AR(1), ARMA(1,1), MA(1), ARMA(0,0).

The BIC criterion prefers the candidate models in the following order: AR(1), MA(1), ARMA(1,1), ARMA(0,0).

Since ARMA(1,1) is a competitive model, we should check for parameter redundancy to see if it can be a good candidate when we introduce ARCH/GARCH effects.

```
arma_1_1 = Arima(Yn, order = c(i,0,j))
plot_roots(coef(arma_1_1))
```



Unfortunately, this model has roots that are pretty close together indicating parameter redundancy. As such, we will remove it from consideration.

After this analysis we have three models in consideration, AR(1), MA(1) and ARMA(0,0) which is white noise. Given that both AIC and BIC agree on the AR(1) model, I feel comfortable deciding on AR(1) as our ARMA model without any further analysis.

```
AR_1 <- Arima(Yn, order = c(1,0,0))
AR_1
```

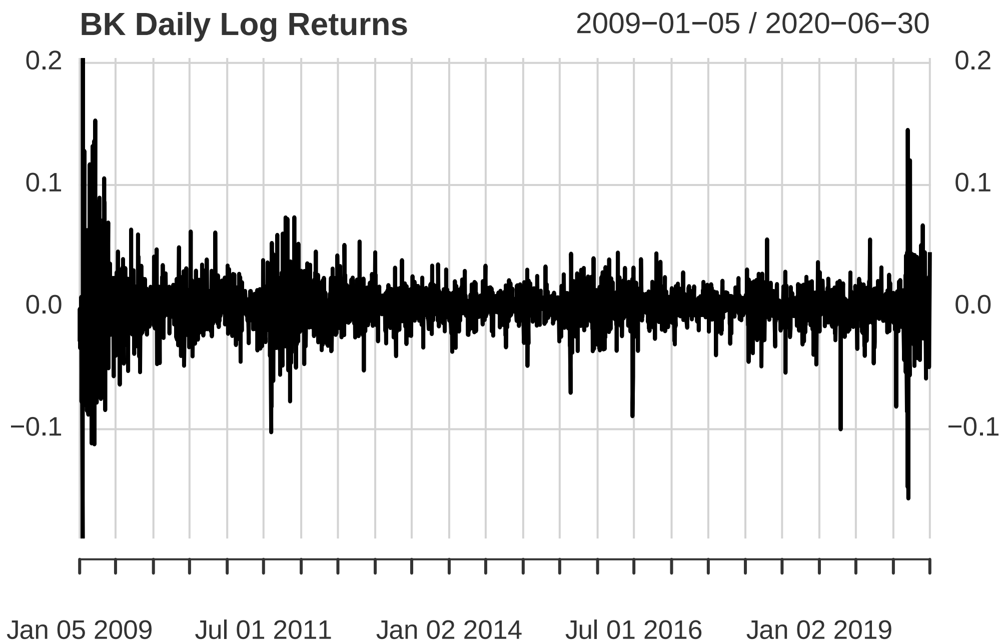
```
Series: Yn
ARIMA(1,0,0) with non-zero mean

Coefficients:
          ar1    mean
      -0.1313  2e-04
s.e.   0.0185  3e-04

sigma^2 = 0.0004251:  log likelihood = 7122.91
AIC=-14239.82   AICc=-14239.81   BIC=-14221.91
```

We can see that our coefficient ar1 is significant here.

Part 2)

One of the tell tale signs of an ARCH effect is clusters of high variance, indicating that although variance is constant conditional variance is not. We look at the daily log return time series plot again.

```
plot(Yn, grid.col = "lightgray", main = "BK Daily Log Returns")
```



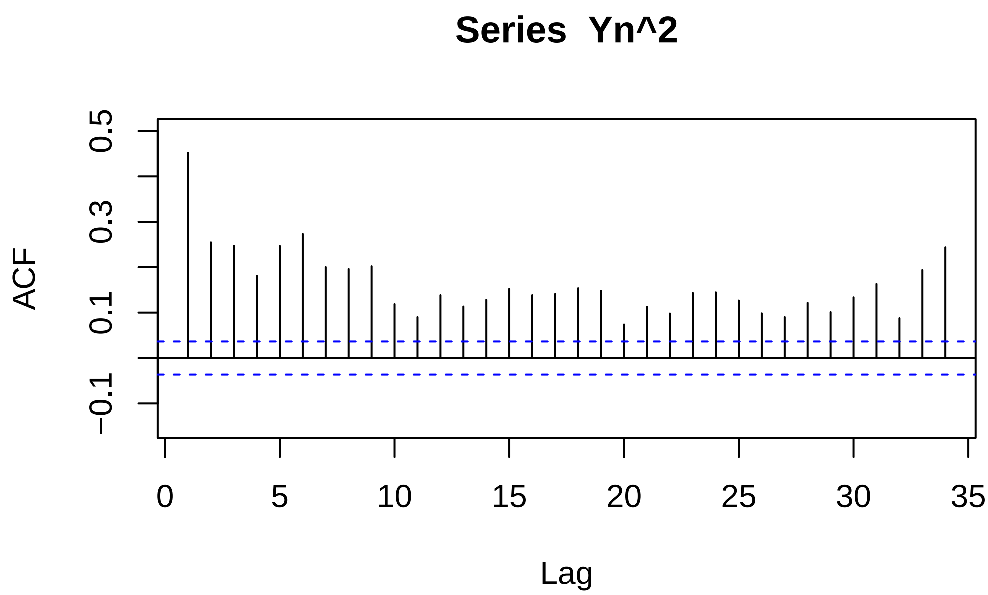
We can clearly see that large movements cluster together, most evident at the start of 2009 and at the start of COVID in March of 2020.

Due to this artifact, our ARMA model will struggle with provide us a prediction interval with suitable coverage in these shock times, it will be too narrow.

Ultimately we are most interested in the prediction invervals during times of shock, this is when our ability to measure risk is very important. This fact motives us to find a model which can improve our coverage rates during periods of high volatility.

We can verify that an ARCH effect in the data is apparent by looking at the ACF of the squared return data:

```
Acf(Yn^2, ylim = c(-.15,.5))
```



There is significance in every ACF, it is clear there is an ARCH effect.

We now proceed to fit an ARMA + GARCH model. To start we will carry the ARMA model from part 1 (AR(1)) forward to this model, and choose GARCH(1,1) due to its simplicity and general effectiveness.

```
spec = ugarchspec(mean.model = list(armaOrder = c(1, 0)),
                  variance.model = list(garchOrder = c(1,1)))
fit = ugarchfit(data = Yn, spec = spec)
showShort(fit)
```

```
*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*
GARCH Model : sGARCH(1,1)
Mean Model  : ARFIMA(1,0,0)
Distribution    : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
```

```
mu      0.000556    0.000253   2.1991 0.027873
ar1    -0.039570    0.020792  -1.9032 0.057018
omega   0.000011    0.000001  12.1830 0.000000
alpha1  0.129190    0.008839  14.6155 0.000000
beta1   0.841103    0.010910  77.0915 0.000000

LogLikelihood : 7808.079

Information Criteria
------------------------------------
  Akaike  Bayes Shibata Hannan-Quinn
 -5.3963 -5.386 -5.3963      -5.3926

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                    0.01859  0.8916
Lag[2*(p+q)+(p+q)-1][2]   0.90540  0.7951
Lag[4*(p+q)+(p+q)-1][5]   2.37688  0.6017
d.o.f=1
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.5682  0.4510
Lag[2*(p+q)+(p+q)-1][5]    1.4098  0.7621
Lag[4*(p+q)+(p+q)-1][9]    2.9068  0.7743
d.o.f=2

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]    0.7256 0.500 2.000  0.3943
ARCH Lag[5]    1.6265 1.440 1.667  0.5597
ARCH Lag[7]    2.8878 2.315 1.543  0.5350

Elapsed time : 0.489444
```

First we examine the significance of the AR(1) parameters. Mu is significant with a p-value of 0.02783. AR1 it is just barely insignificant at a $\alpha = 0.05$ threshold having a p-value of 0.057. Due to this we should consider the other candidate models, MA(1) and ARMA(0,0). I will also use AIC and BIC as a way to compare the models, so we can note that AIC is -5.3963 and BIC is -5.386.

```
spec2 = ugarchspec(mean.model = list(armaOrder = c(0, 1)),
                  variance.model = list(garchOrder = c(1,1)))
fit2 = ugarchfit(data = Yn, spec = spec2)
showShort(fit2)
```

```
*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*
GARCH Model : sGARCH(1,1)
Mean Model  : ARFIMA(0,0,1)
Distribution    : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000556    0.000252   2.2019 0.027671
ma1    -0.038971    0.020635  -1.8886 0.058948
omega   0.000011    0.000001  12.2016 0.000000
alpha1  0.129273    0.008844  14.6162 0.000000
beta1   0.841003    0.010916  77.0447 0.000000

LogLikelihood : 7808.052

Information Criteria
------------------------------------
  Akaike  Bayes Shibata Hannan-Quinn
 -5.3963 -5.386 -5.3963      -5.3926

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                    0.02663  0.8704
Lag[2*(p+q)+(p+q)-1][2]   1.00647  0.7360
Lag[4*(p+q)+(p+q)-1][5]   2.52863  0.5582
d.o.f=1
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
```

```
------------------------------------
                        statistic p-value
Lag[1]                     0.5624  0.4533
Lag[2*(p+q)+(p+q)-1][5]    1.4006  0.7643
Lag[4*(p+q)+(p+q)-1][9]    2.8980  0.7757
d.o.f=2

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]    0.7204 0.500 2.000  0.3960
ARCH Lag[5]    1.6217 1.440 1.667  0.5609
ARCH Lag[7]    2.8850 2.315 1.543  0.5355

Elapsed time : 0.494885
```

In this fit we use MA(1), mu is again significant with a p-value below 0.05. Coefficient MA1 is barely insignificant with a p-value of 0.058948. AIC is -5.3963 and BIC is -5.386.

```
spec3 = ugarchspec(mean.model = list(armaOrder = c(0, 0)),
                  variance.model = list(garchOrder = c(1,1)))
fit3 = ugarchfit(data = Yn, spec = spec3)
showShort(fit3)
```

```
*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*
GARCH Model : sGARCH(1,1)
Mean Model  : ARFIMA(0,0,0)
Distribution    : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000548    0.000261   2.0987 0.035845
omega   0.000011    0.000001  12.3354 0.000000
alpha1  0.132234    0.009073  14.5739 0.000000
beta1   0.837893    0.011148  75.1638 0.000000

LogLikelihood : 7806.285
```

```
Information Criteria
------------------------------------
                                    
  Akaike   Bayes Shibata Hannan-Quinn
 -5.3958 -5.3875 -5.3958      -5.3928

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                      3.862 0.04939
Lag[2*(p+q)+(p+q)-1][2]     4.829 0.04518
Lag[4*(p+q)+(p+q)-1][5]     6.318 0.07563
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.4978  0.4805
Lag[2*(p+q)+(p+q)-1][5]    1.3337  0.7807
Lag[4*(p+q)+(p+q)-1][9]    2.8356  0.7858
d.o.f=2

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]    0.7097 0.500 2.000  0.3995
ARCH Lag[5]    1.6249 1.440 1.667  0.5601
ARCH Lag[7]    2.8847 2.315 1.543  0.5356

Elapsed time : 0.7203739
```

In this fit we use the ARMA(0,0) model, one corresponding to white noise. We see that the mu coefficient remains significant. AIC is -5.3958, BIC is -5.3875. Lets now compare our three candidate models:

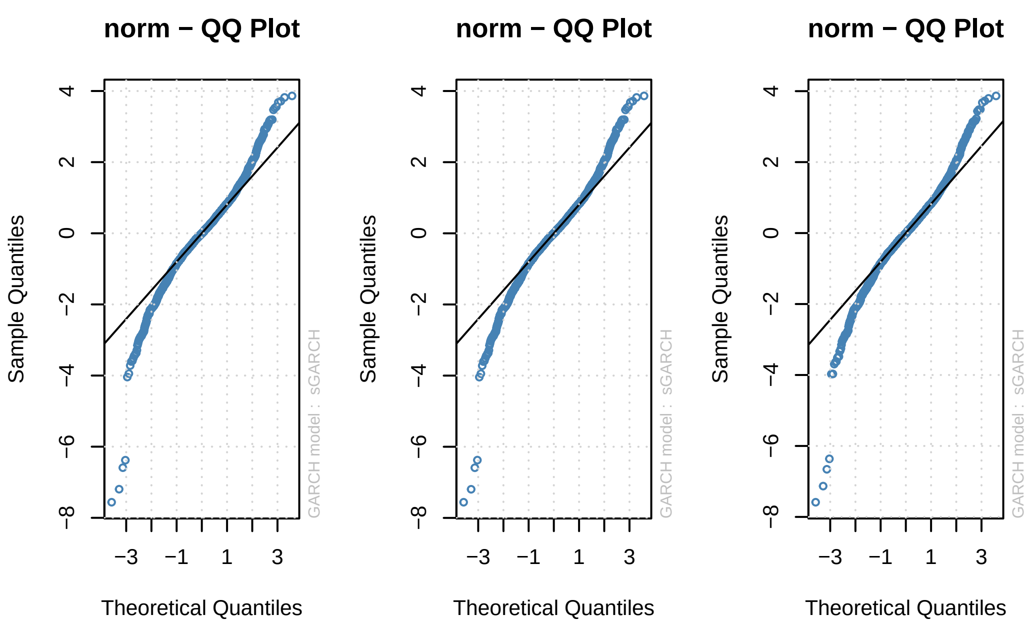
ARMA Model	AIC	BIC
AR(1)	-5.3963	-5.386
MA(1)	-5.3963	-5.386
ARMA(0,0)	-5.3958	-5.3875

We can see that AIC would select either the AR(1) or MA(1) model, they are tied, and BIC would select the ARMA(0,0) model. In this case I would lean on the principle of model parsimony for return data and select the ARMA(0,0) model. Neither the AR1 or MA1 coefficients were statistically significant and BIC, which values a models simplicity more than AIC, selected the ARMA(0,0) model.

However, the ARMA(0,0) + GARCH(1,1) plot model struggles with correlation in the residuals. We can see this as it rejects the null hypothesis of no serial correlation in the Weighted Ljung-Box test. AR(1) + GARCH(1,1) and MA(1) + GARCH(1,1) don't have this issue, which motivates me to continue to dig into other specifications before deciding on the ARMA model that we will use.

So, lets look at the assumption of a normal distribution of errors for each model:

```
par(mfrow = c(1,3))
plot(fit, which = 9, main = "Norm QQ Plot AR(1)")
plot(fit2, which = 9, main = "Norm QQ Plot AR(1)")
plot(fit3, which = 9, main = "Norm QQ Plot AR(1)")
```



Clearly the conditional distribution is not normal. We get an idea of what distributions are better by fitting them and seeing their information criteria.

```
n = length(Yn)
```

```
at1 = residuals(fit)
et1 = residuals(fit, stand = T)
at2 = residuals(fit2)
et2 = residuals(fit2, stand = T)
at3 = residuals(fit3)
et3 = residuals(fit3, stand = T)


dists = c("std", "sstd", "ged","sged", "nig", "jsu")
#Model 1 fits
fits_1 = vector("list", 6)
for (i in 1:6) fits_1[[i]] = fitdist(dists[i], et1)

#Model 2 fits
fits_2 = vector("list", 6)
for (i in 1:6) fits_2[[i]] = fitdist(dists[i], et2)

#Model 3 fits
fits_3 = vector("list", 6)
for (i in 1:6) fits_3[[i]] = fitdist(dists[i], et3)


#Maximum Likelihood, parameters
ml_1 = c();ml_2 = c(); ml_3 = c();p = c();
for(i in 1:length(dists)) {
  ml_1[i] = -tail(fits_1[[i]]$values,1)
  ml_2[i] = -tail(fits_2[[i]]$values,1)
  ml_3[i] = -tail(fits_3[[i]]$values,1)
  p[i] = length(fits_1[[i]]$pars)
}
#AIC/BIC vectors
aic_1 = -2*ml_1 + 2*p; aic_2 = -2*ml_2 + 2*p; aic_3 = -2*ml_3 + 2*p; names(aic_1) = dists;
bic_1 = -2*ml_1 + log(n)*p;bic_2 = -2*ml_2 + log(n)*p;bic_3 = -2*ml_3 + log(n)*p; names(bi

cat("Model 1: \n")
```

```
Model 1:
```

```
rbind(AIC = aic_1, BIC = bic_1)
```

```
         std      sstd       ged      sged       nig       jsu
```

```
AIC 7937.582 7935.711 7950.265 7948.031 7932.666 7932.596
BIC 7955.491 7959.590 7968.174 7971.910 7956.545 7956.475
```

```
cat("Model 2: \n")
```

```
Model 2:
```

```
rbind(AIC = aic_2, BIC = bic_2)
```

```
         std     sstd      ged     sged      nig      jsu
AIC 7937.627 7935.756 7950.323 7948.144 7932.725 7932.648
BIC 7955.536 7959.635 7968.232 7972.023 7956.604 7956.527
```

```
cat("Model 3: \n")
```

```
Model 3:
```

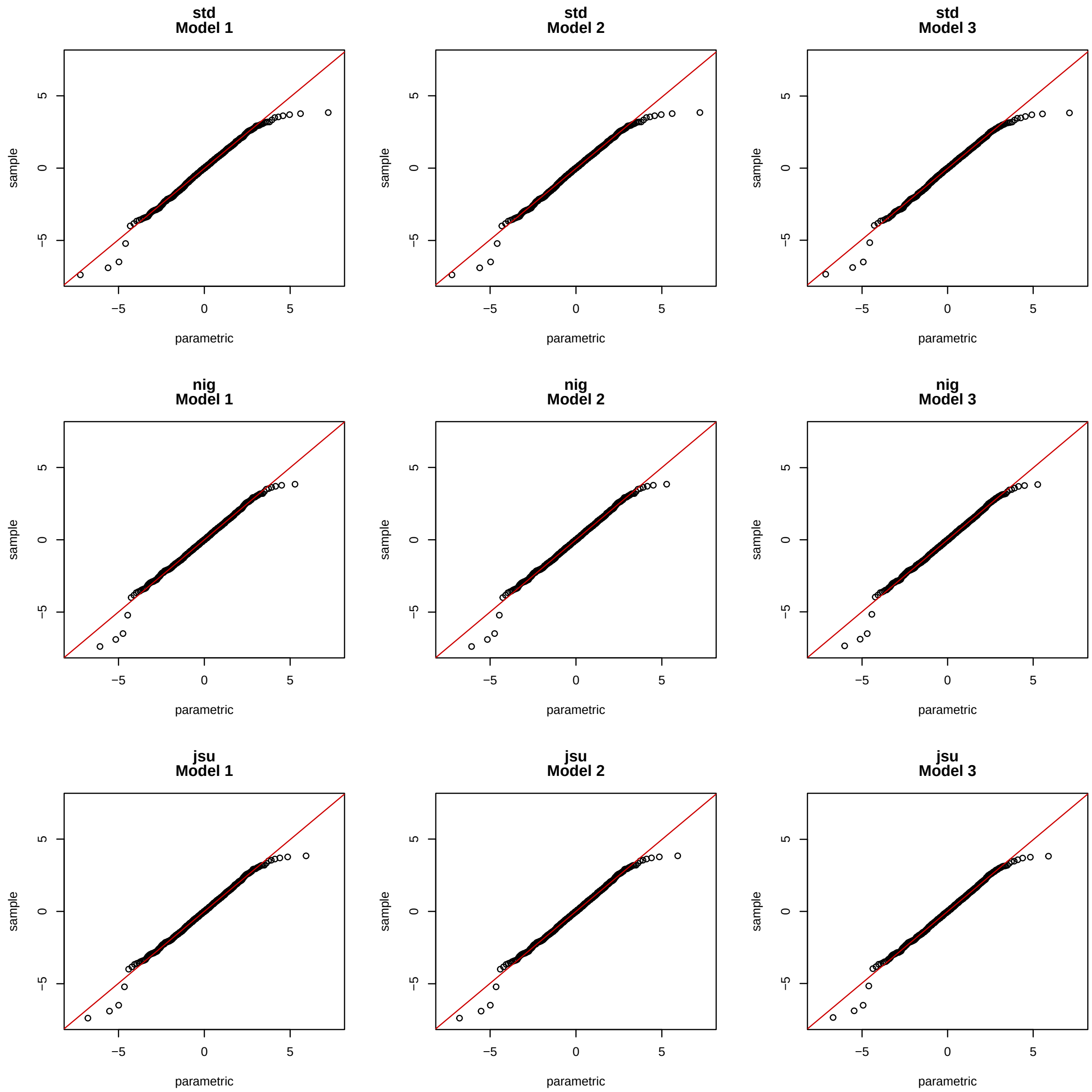
```
rbind(AIC = aic_3, BIC = bic_3)
```

```
         std     sstd      ged     sged      nig      jsu
AIC 7942.278 7940.912 7957.041 7955.307 7938.698 7938.226
BIC 7960.187 7964.790 7974.951 7979.186 7962.576 7962.105
```

AIC selects Johnson S_U followed by inverse Gaussian for all three models. BIC selects standardized t for all three models. Lets compare QQ-plots for these three distributions.

```
par(mfrow = c(3,3))
n = length(Yn)
q = ((1:n) -0.5)/n
q1 = quantile(et1, q)
q2 = quantile(et2, q)
q3 = quantile(et3, q)

for(i in c(1,5,6)){
  est_1 = fits_1[[i]]$pars
  est_2 = fits_2[[i]]$pars
  est_3 = fits_3[[i]]$pars
```

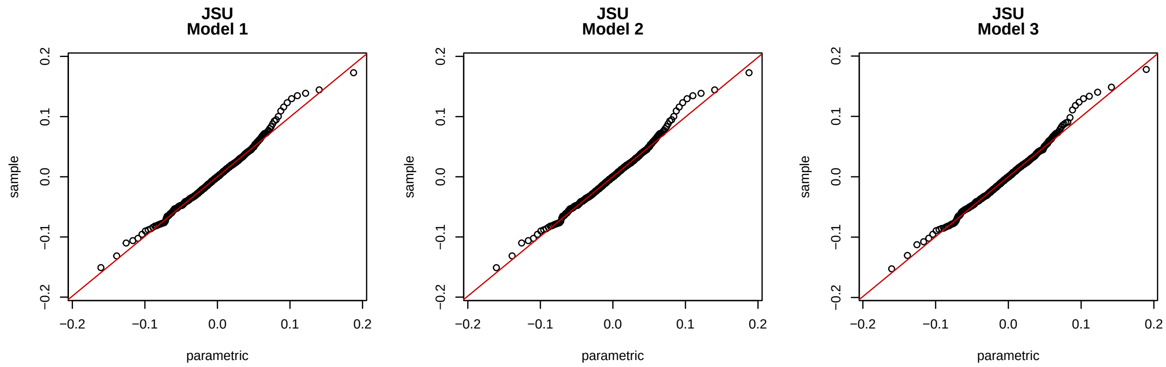
```
  qx_1 = qdist(dists[i], q, mu = est_1["mu"], sigma = est_1["sigma"], skew = est_1["skew"]
  qx_2 = qdist(dists[i], q, mu = est_2["mu"], sigma = est_2["sigma"], skew = est_2["skew"]
  qx_3 = qdist(dists[i], q, mu = est_3["mu"], sigma = est_3["sigma"], skew = est_3["skew"]

  plot(qx_1,q1, main = c(dists[i],"Model 1"), ylab = "sample", xlab = "parametric", xlim =
  abline(lsfit(qx_1,q1)$coef, col = "red3") ## reference line

  plot(qx_2,q2, main = c(dists[i], "Model 2"), ylab = "sample", xlab = "parametric", xlim
  abline(lsfit(qx_2,q2)$coef, col = "red3") ## reference line

  plot(qx_3,q3, main = c(dists[i],"Model 3"), ylab = "sample", xlab = "parametric", xlim =
  abline(lsfit(qx_3,q3)$coef, col = "red3") ## reference line


}
```



Out of these, the inverse Gaussian and Johnson S_U are the most competitive. I would choose Johnson S_U. We should also check that this distribution is suitable for the marginal distribution.

```
par(mfrow = c(1,3))
jsu_fit_1 = fitdist("jsu", at1)
jsu_fit_2 = fitdist("jsu", at2)
```

```
jsu_fit_3 = fitdist("jsu", at3)

q1 = quantile(at1, q)
q2 = quantile(at2, q)
q3 = quantile(at3, q)

est_1 = jsu_fit_1$pars
est_2 = jsu_fit_2$pars
est_3 = jsu_fit_3$pars

qx_1 = qdist("jsu", q, mu = est_1["mu"], sigma = est_1["sigma"],
             skew = est_1["skew"], shape = est_1["shape"])
qx_2 = qdist("jsu", q, mu = est_2["mu"], sigma = est_2["sigma"],
               skew = est_2["skew"], shape = est_2["shape"])
qx_3 = qdist("jsu", q, mu = est_3["mu"], sigma = est_3["sigma"],
             skew = est_3["skew"], shape = est_3["shape"])

plot(qx_1,q1, main = c("JSU","Model 1"), ylab = "sample", xlab = "parametric",
     xlim = c(min(at1), -min(at1)), ylim = c(min(at1), -min(at1)))
  abline(lsfit(qx_1,q1)$coef, col = "red3") ## reference line

plot(qx_2,q2, main = c("JSU", "Model 2"), ylab = "sample", xlab = "parametric",
     xlim = c(min(at2), -min(at2)), ylim = c(min(at2), -min(at2)))
  abline(lsfit(qx_2,q2)$coef, col = "red3") ## reference line

plot(qx_3,q3, main = c("JSU","Model 3"), ylab = "sample", xlab = "parametric",
     xlim = c(min(at3), -min(at3)), ylim = c(min(at3), -min(at3)))
  abline(lsfit(qx_3,q3)$coef, col = "red3") ## reference line
```



The JSU distribution does a good job with the AR(1) residuals, as such we will use it in our model.

```
spec = ugarchspec(mean.model = list(armaOrder = c(1, 0)),
                  variance.model = list(garchOrder = c(1,1)),
                  distribution.model = "jsu")
fit = ugarchfit(data = Yn, spec = spec)
showShort(fit)
```

```
*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*
GARCH Model : sGARCH(1,1)
Mean Model  : ARFIMA(1,0,0)
Distribution    : jsu

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000537    0.000244   2.1951 0.028154
ar1    -0.061098    0.018335  -3.3323 0.000861
omega   0.000005    0.000003   1.8895 0.058828
alpha1  0.092458    0.016639   5.5568 0.000000
beta1   0.896508    0.018569  48.2789 0.000000
skew   -0.109903    0.058352  -1.8835 0.059638
shape   1.500526    0.078627  19.0840 0.000000

LogLikelihood : 7957.499

Information Criteria
------------------------------------
  Akaike   Bayes Shibata Hannan-Quinn
 -5.4983 -5.4838 -5.4983      -5.4931

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.9433  0.3314
Lag[2*(p+q)+(p+q)-1][2]    1.7076  0.3310
Lag[4*(p+q)+(p+q)-1][5]    3.0736  0.4130
d.o.f=1
```

```
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                      3.078 0.07936
Lag[2*(p+q)+(p+q)-1][5]     4.171 0.23345
Lag[4*(p+q)+(p+q)-1][9]     5.504 0.35924
d.o.f=2

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]     1.336 0.500 2.000  0.2478
ARCH Lag[5]     1.757 1.440 1.667  0.5276
ARCH Lag[7]     2.831 2.315 1.543  0.5461

Elapsed time : 1.109964
```

```
spec2 = ugarchspec(mean.model = list(armaOrder = c(0, 1)),
                  variance.model = list(garchOrder = c(1,1)),
                  distribution.model = "jsu")
fit2 = ugarchfit(data = Yn, spec = spec2)
showShort(fit2)
```

```
*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*
GARCH Model : sGARCH(1,1)
Mean Model  : ARFIMA(0,0,1)
Distribution    : jsu

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000537    0.000244   2.2014 0.027706
ma1    -0.060081    0.018124  -3.3150 0.000916
omega   0.000005    0.000003   1.8821 0.059818
alpha1  0.092682    0.016730   5.5399 0.000000
beta1   0.896340    0.018671  48.0077 0.000000
```

```
skew   -0.110106    0.058394  -1.8856 0.059353
shape   1.500550    0.078596  19.0921 0.000000

LogLikelihood : 7957.399

Information Criteria
------------------------------------
  Akaike   Bayes Shibata Hannan-Quinn
 -5.4982 -5.4838 -5.4982       -5.493

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                      0.863  0.3529
Lag[2*(p+q)+(p+q)-1][2]     1.845  0.2708
Lag[4*(p+q)+(p+q)-1][5]     3.333  0.3524
d.o.f=1
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                      3.028 0.08182
Lag[2*(p+q)+(p+q)-1][5]     4.112 0.24045
Lag[4*(p+q)+(p+q)-1][9]     5.448 0.36669
d.o.f=2

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]     1.313 0.500 2.000  0.2518
ARCH Lag[5]     1.735 1.440 1.667  0.5327
ARCH Lag[7]     2.817 2.315 1.543  0.5487

Elapsed time : 0.8751259
```

```r
spec3 = ugarchspec(mean.model = list(armaOrder = c(0, 0)),
                  variance.model = list(garchOrder = c(1,1)),
                  distribution.model = "jsu")
fit3 = ugarchfit(data = Yn, spec = spec3)
showShort(fit3)
```

```
*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*
GARCH Model : sGARCH(1,1)
Mean Model  : ARFIMA(0,0,0)
Distribution    : jsu

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000548    0.000257   2.1284 0.033303
omega   0.000005    0.000003   2.0347 0.041878
alpha1  0.096591    0.016264   5.9391 0.000000
beta1   0.891569    0.018114  49.2189 0.000000
skew   -0.100606    0.058959  -1.7064 0.087938
shape   1.520295    0.080737  18.8303 0.000000

LogLikelihood : 7952.009

Information Criteria
------------------------------------
  Akaike   Bayes Shibata Hannan-Quinn
 -5.4952 -5.4828 -5.4952      -5.4907

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                      3.760 0.05251
Lag[2*(p+q)+(p+q)-1][2]     4.657 0.05023
Lag[4*(p+q)+(p+q)-1][5]     6.063 0.08694
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                      2.594  0.1073
Lag[2*(p+q)+(p+q)-1][5]     3.656  0.3001
Lag[4*(p+q)+(p+q)-1][9]     5.003  0.4290
d.o.f=2

Weighted ARCH LM Tests
```

```
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]     1.241 0.500 2.000  0.2653
ARCH Lag[5]     1.674 1.440 1.667  0.5479
ARCH Lag[7]     2.762 2.315 1.543  0.5595

Elapsed time : 0.6963708
```

Now, a big difference can be seen as the AR1 coefficient of the AR(1) + GARCH(1,1) model and the MA1 coefficient of the MA(1) + GARCH(1,1) model are now highly significant. Lets compare the AIC and BIC for the three fits:

ARMA Model	AIC	BIC
AR(1)	-5.4983	-5.4838
MA(1)	-5.4982	-5.4838
ARMA(0,0)	-5.4952	-5.4828

After specifying the distribution of residuals, both the AIC and BIC select the AR(1) + GARCH(1,1) model. Additionally the AR(1) + GARCH(1,1) model does not have correlation in residuals or squared residuals as it fails to reject the null hypothesis in both Weighted Ljung-Box Tests, and ARCH effects are well handled as can be seen by the weighted ARCH LM tests all having high p-values. We finally select AR(1) as our ARMA portion of the ARMA + GARCH model. Before concluding on AR(1) + GARCH(1,1) we should check up to GARCH(2,2) to see if fit is improved.

```r
ic = c(); ind = 0
fits = vector("list", 6)
for(p in 1:2) {
  for(q in 0:2){
    ind = ind + 1
    spec = ugarchspec(mean.model = list(armaOrder = c(1,0)),
                      variance.model = list(garchOrder = c(p,q)),
                      distribution.model = "jsu")
    garch = ugarchfit(data = Yn, spec = spec)
    ic = cbind(ic, infocriteria(garch))
    fits[[ind]] = garch
    names(fits)[ind] = paste0("garch", p, q)
  }
}
colnames(ic) = names(fits);ic
```

```
              garch10   garch11   garch12   garch20   garch21   garch22
Akaike       -5.382210 -5.498271 -5.497718 -5.426321 -5.497456 -5.497026
Bayes        -5.369825 -5.483821 -5.481204 -5.411872 -5.480943 -5.478448
Shibata      -5.382219 -5.498282 -5.497733 -5.426333 -5.497472 -5.497046
Hannan-Quinn -5.377747 -5.493063 -5.491767 -5.421114 -5.491505 -5.490331
```

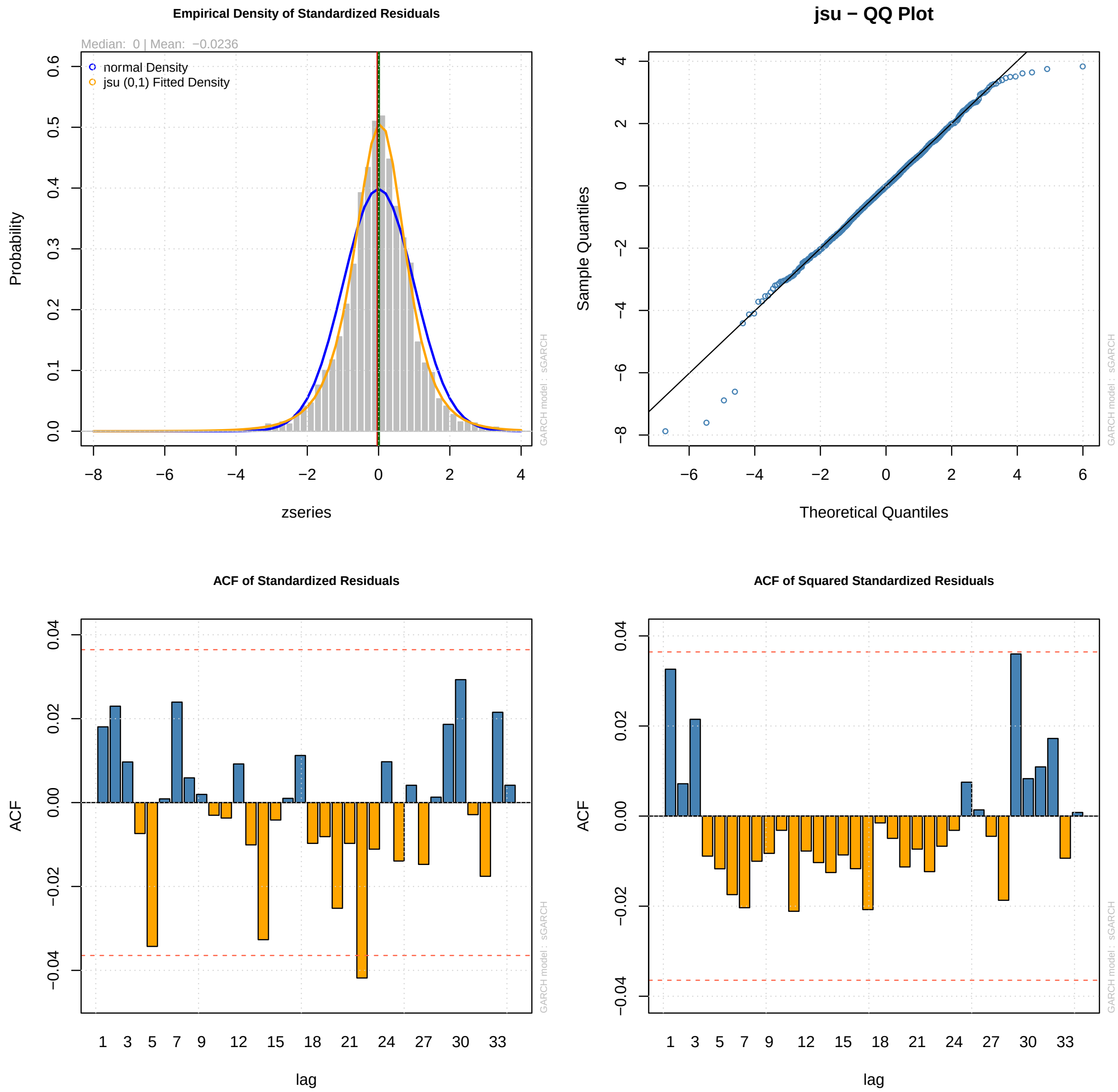
```r
cat("Model Selected: "); apply(ic, 1, function(u) colnames(ic)[which.min(u)])
```

```
Model Selected:
```

```
      Akaike        Bayes      Shibata Hannan-Quinn
   "garch11"    "garch11"    "garch11"    "garch11"
```

All information criteria choose GARCH(1,1) as our GARCH component. I feel confident choosing AR(1) + GARCH(1,1) with Johnson S_U conditional error distribution as the final model for the Bank of New York Mellon daily return series. The key plots are below:

```r
par(mfrow = c(2,2))
plot(fit, which = 8)
plot(fit, which = 9)
plot(fit, which = 10)
plot(fit, which = 11)
```



We can see that JSU distribution fits the density function very well. Also, as discussed. the ARCH effect has been handled. We no longer see significance in individual ACFs or squared ACFs.

Now we need to validate the model with rolling forecasts.

```
n = dim(Yn)[1]; n0 = dim(Yt)[1]; n.fore = n0 - n
dates = data.frame(data = c("Total data", "Sample", "Forecast"),
```

```
                     from = paste(time(Yt)[c(1,1,n+1)]), to = paste(time(Yt)[c(n0,n,n0)]),
                     Size = c(n0,n,n.fore));dates
```

```
        data       from         to Size
1 Total data 2009-01-05 2024-12-06 4009
2     Sample 2009-01-05 2020-06-30 2892
3   Forecast 2020-07-01 2024-12-06 1117
```

Fitting the full model:

```
spec.11 = ugarchspec(mean.model = list(armaOrder = c(1,0)),
                     variance.model = list(garchOrder = c(1,1)), distribution.model = "jsu
fit.11 = ugarchfit(data = Yt, spec = spec.11, out.sample = n.fore)
```

Checking coverage rate:

```
fore = ugarchforecast(fit.11, n.roll = n.fore - 1)
cat("AR(1) + GARCH(1,1)"); rate(fore)
```

```
AR(1) + GARCH(1,1)



      One-Step Rolling Forecast
----------------------------------
       coverage below PI beyond PI
95% PI   0.9543   0.0242    0.0215
90% PI   0.9069   0.0483    0.0448
```

A coverage rate of 0.9543 is close to the desired coverage of 0.95. We also see the asymmetry property that has been shown by studies.

We compare this to the AR(1) model, which had constant variance. We refit using rugarch's function which allows us to specify the distribution model.

```
spec.0 = arfimaspec(mean.model = list(armaOrder = c(1,0)),
                    distribution.model = "jsu")
fit.0 = arfimafit(data = Yt, spec = spec.0, out.sample = n.fore)
fore.0 = arfimaforecast(fit.0, n.roll = n.fore - 1)
showShort0(fit.0)
```

```
*---------------------------------*
*          ARFIMA Model Fit        *
*---------------------------------*
Mean Model  : ARFIMA(1,0,0)
Distribution    : jsu

Optimal Parameters
------------------------------------
       Estimate  Std. Error  t value Pr(>|t|)
mu     0.000068    0.000338  0.19954 0.841842
ar1   -0.099807    0.018286 -5.45816 0.000000
sigma  0.020675    0.000743 27.83807 0.000000
skew  -0.083968    0.037122 -2.26193 0.023702
shape  1.022606    0.036072 28.34881 0.000000

LogLikelihood : 7674.344

Information Criteria
------------------------------------
  Akaike   Bayes Shibata Hannan-Quinn
 -5.3038 -5.2935 -5.3038      -5.3001

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                      2.690 0.10096
Lag[2*(p+q)+(p+q)-1][2]     3.010 0.03449
Lag[4*(p+q)+(p+q)-1][5]     3.275 0.36537

H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                      532.6       0
Lag[2*(p+q)+(p+q)-1][2]     619.2       0
Lag[4*(p+q)+(p+q)-1][5]     870.5       0

ARCH LM Tests
------------------------------------
             Statistic DoF P-Value
ARCH Lag[2]      544.9   2       0
```

```
ARCH Lag[5]       679.4   5        0
ARCH Lag[10]      735.9  10        0
```

The coverage rate for the AR(1) model is as follows:

```r
cat("AR(1) + i.i.d JSU Noise"); rate0(fore.0)
```

```
AR(1) + i.i.d JSU Noise


      One-Step Rolling Forecast
----------------------------------
       coverage below PI beyond PI
95% PI   0.9731   0.0125    0.0143
90% PI   0.9221   0.0403    0.0376
```

The AR(1) model has significant over coverage.

```r
plot_PI(fore.0, fore)
```


95% PI
sGARCH
i.i.d.
Time Series
observed
forecast
Outside PI
sGARCH
i.i.d.
Daily Return
0.10
0.05
0.00
−0.05
−0.10
2021
2022
2023
2024