M2 Econometrics and Empirical Economics Track

Financial Econometrics, Homework 1

Lecturer:
Prof. Nour Meddhai

Student:
Jingwan LUO (Student Number 22007276)
Zhihan Zhang (Student Number 21613596)

Contents

Toulouse, March 1, 2022

1 QUESTION 1

1.1 SUMMARY STATISTICS

Figure 1 demonstrates a general glance on the data. Summary statistics is presented with the following table.

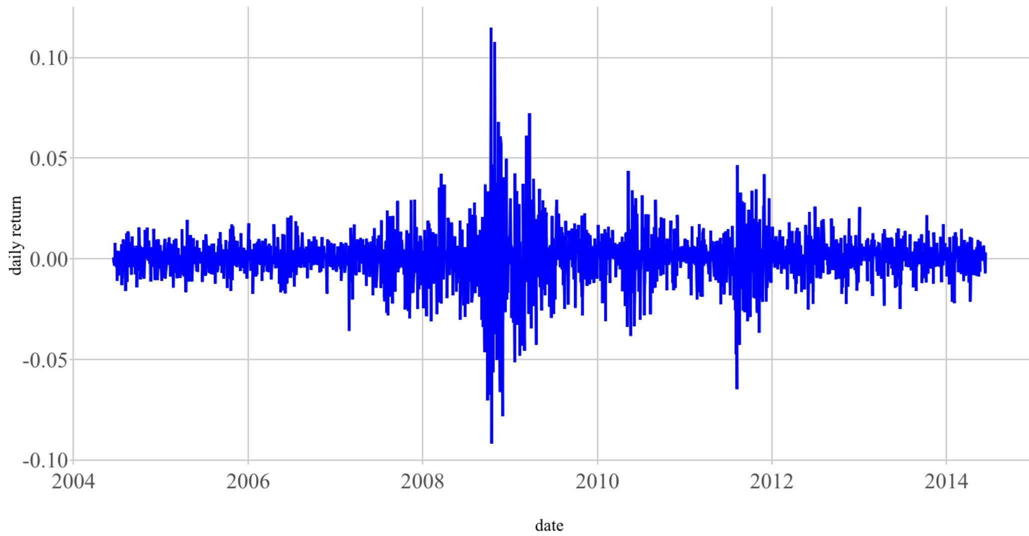
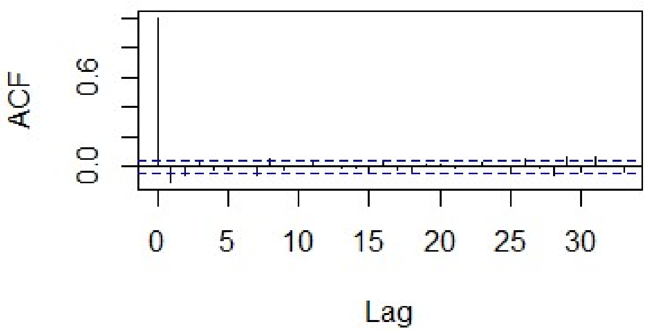


Figure 1: Data Glance on the Daily Return

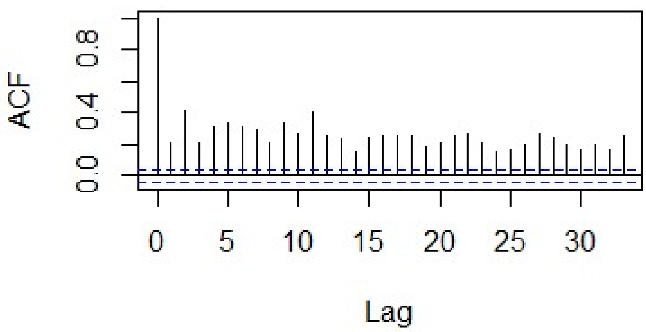
Mean	Stdev	Skewness	Kurtosis	Ex.Kurtosis
2.929×10^{-4}	1.588×10^{-4}	1.470×10^{-2}	13.6997	10.6997

Table 1: Summary Statistics of Daily Return

1.1.1 ACF plots



(a) ACF Plot of R_t



(b) ACF Plot of R_t^2

Figure 2: ACF PLOTS AT SEVERAL LAGS

COMMENTS: Figure 2 presents the ACF plots of R_t. From Figure 2 we observe that, though it may exhibit serial correlation after several lags, the link to the history decreases sharply, and this coincides with stylized facts of financial econometrics, that daily return has very little correlation.

1.2 Rolling Statistics

The following plots are for rolling statistics, on monthly window and yearly window, as depicted in Figure 3 and Figure 4. As expected, the rolling stats becomes more smooth as we increase the width of window. Rolling means and variance had reached the peak during the time of financial crisis, while the skewness and kurtosis also experience volatile fluctuation during the crisis.

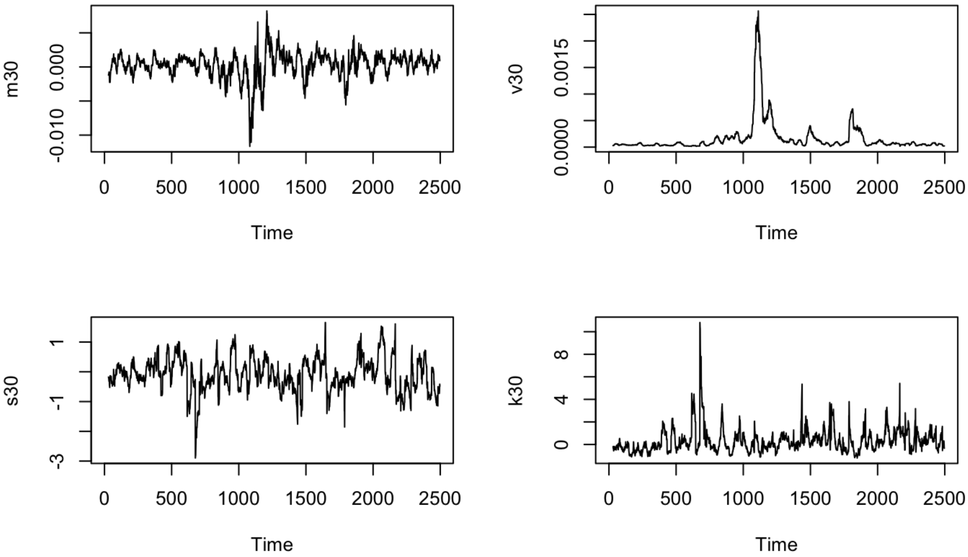


Figure 3: Rolling statistics, per month as window size

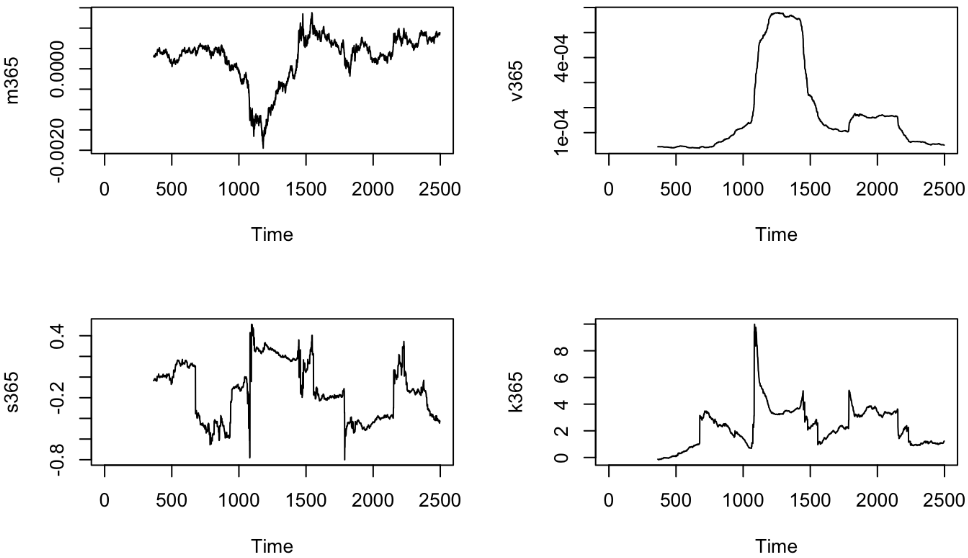


Figure 4: Rolling statistics, per year as window size

2 QUESTION 2

2.1 SIMULATION IN (A)

MODEL DESIGN 1:

$$
\begin{aligned}
R_t &= \mu_t + \sigma_t z_t, z_t \sim \mathcal{N}(0,1) \\
\sigma_t^2 &= \omega + \alpha R_{t-1}^2 + \beta \sigma_{t-1}^2 \\
\psi &= (0, 0.01, 0.05, 0.9, 0)
\end{aligned}
$$

MODEL DESIGN 2:

$$
\begin{aligned}
R_t &= \mu_t + \sigma_t z_t, z_t \sim \mathcal{N}(0,1) \\
\sigma_t^2 &= \omega + \alpha (R_{t-1}^2 - \theta) + \beta \sigma_{t-1}^2 \\
\psi &= (0, 0.01, 0.05, 0.6, 2)
\end{aligned}
$$

STEPS: We firstly generated R_t following each of the design randomly, next, with a manually written procedure Maximum Likelihood Estimation which outputs the value of negative log-likelihood, we fitted our randomly generated R_t (number of observations $T \in \{250, 1000\}$ for each design) to the procedure and take the summary statistics.

Table 2: Simulation Mean and S.D. for Question 2, (a), $\mathcal{N}(0,1)$

	μ	ω	α	β	θ
$T = 250$,	0.001009915	0.016204890	0.046723113	0.834294288	-0.018280030
First Design	(0.02528920)	(0.02463376)	(0.04843332)	(0.18671041)	(0.79702462)
$T = 1000$,	-0.001547458	0.015609792	0.051743484	0.865775960	-0.019761570
First Design	(0.01213386)	(0.02321683)	(0.02590004)	(0.12700134)	(0.20451365)
$T = 250$,	-0.002020656	0.059370650	0.065587170	0.649768662	1.101927476
Second Design	(0.04587505)	(0.07359194)	(0.05526415)	(0.22754040)	(0.78070127)
$T = 1000$,	-0.002720411	0.032902650	0.057309333	0.577428486	1.910531731
Second Design	(0.02252846)	(0.05257406)	(0.02203494)	(0.10541452)	(0.50444645)

COMMENT: One of the merits of Maximum Likelihood Estimation is its consistency, for this reason, we expect the estimated parameters to converge to the true ones as the sample increases.

1. (*Mean*) For each design, as number of observation T increases, the mean values gets closer to the actual value of ψ.

2. (*S.D.*) For each design, volatility decreases as T is larger.

3. (*Comparing Designs*) Between the two designs, cases where $T = 1000$, Second Design outperforms the other. Since the second design $\theta \neq 0$, it's common to see that model with richer design better fits the data.

2.2 SIMULATION IN (B)

For the similar procedure as in the last subsection, but for a change of assumption that, $z_t \sim T(10)$, $T(n) = X/\sqrt{Y/n}$, X and Y are independent, $X \sim N(0,1)$ and $Y \sim \chi^2(n)$. Resulted *mean* and *s.d.* of in Table 3.

COMMENT:

Table 3: Simulation Mean and S.D. for Question 2, (b), $T(10)$

	μ	ω	α	β	θ
$T = 250$,	0.005960188	0.026622023	0.044767418	0.819358086	0.084416940
First Design	(0.03549735)	(0.05310538)	(0.03247567)	(0.21776809)	(0.62654712)
$T = 1000$,	-0.001289811	0.012276515	0.050536106	0.887085632	-0.052696105
First Design	(0.008686317)	(0.013988828)	(0.019048437)	(0.048627354)	(0.188528254)
$T = 250$,	0.002743198	0.074870089	0.062043724	0.548560892	1.483816941
Second Design	(0.05412987)	(0.12663322)	(0.04053317)	(0.20756002)	(0.82771750)
$T = 1000$,	0.0009818054	0.0443180530	0.0578387458	0.5777400189	1.7667586130
Second Design	(0.02138454)	(0.07533654)	(0.01947793)	(0.10899446)	(0.48169716)

1. (*Comparing to a, i*): The same conclusions can be drawn similar to (a), mean converges to the true value while volatility shrinks as T increases, and with more complicated design the model fits better.
2. (*Comparing to a, ii*): In most cases, under the assumption that z_t is normal instead of student T's, simulated S.D. is smaller. One of the possible explanations is when assuming $T(10)$, the density has thicker tail.

2.3 Fit GARCH

2.3.1 Model Selection

We chose $ARMA(1,1)$ as the underlying mean process and select the orders of GARCH by Information Criteria. Following a previously written lag selection algorithm (the optimal lag choice is 2) and the summary statistics in Question 1, we tried the following choices of selection in Table 4, using 20% proportion of data of daily return as out-of-sample (so $\frac{N}{5} \approx 500$, the exact 4/5 of the testing sample started at index 1997.5, or 1998).

Table 4: Information Criteria: For Baseline GARCH with ARMA(1, 1)

	$Garch(1,1)$	$Garch(1,2)$	$Garch(2,1)$	$Garch(2,2)$
Akaike	-6.3521	-6.3506	-6.3894	**-6.3940**
Bayes	-6.3353	-6.3310	-6.3698	**-6.3716**
Shibata	-6.3522	-6.3507	-6.3894	-6.3941
Hannan-Quinn	-6.3460	-6.3434	-6.3822	-6.3858

Overall, AIC and BIC criteria suggests that GARCH(2, 2) may be suitable for our selection.

2.3.2 Test for the Leverage Effect

For capture leverage effect:

$$cov(R_t, R_{t+1}^2) < 0 \rightarrow cov(R_t, \sigma_t^2) < 0$$

Besides baseline GARCH, we can also consider two other famous GARCH-leverage model, TGARCH and EGARCH.

TGARCH (Leverage effect η):

$$h_t = \omega + \sum_{i=1}^{p} \alpha_i \epsilon_{t-i}^2 + \sum_{i=1}^{p} \eta_i S_{t-i} \epsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j h_{t-i} \tag{2.3.1}$$

EGARCH (Asymmetric shock capture by the sign of γ):

$$\ln h_t = \omega + \sum_{i=1}^{p} \alpha_i z_{t-i} + \sum_{i=1}^{p} \gamma_i \left(|z_{t-i}| - E\left[|z_{t-i}|\right] \right) + \sum_{j=1}^{q} \beta_j \ln h_{t-i} \tag{2.3.2}$$

For testing, there are four choices of tests available for testing leverage effect as in package "rugarch".

1. Sign Bias Test.
2. Negative Size Bias Test.
3. Positive Size Bias Test.
4. Joint Effect.

Following the same procedure as choosing lags, we perform the same procedure of finding the best lags for TGARCH and for EGARCH. Afterwards we report the attached test results.
Recall the null is **"There is no leverage effect in the GARCH model"**.

Table 5: Leverage Effect: Test Result

	TGARCH(2,2)	eGARCH(2,1)
Sign Bias	2.109915**	0.00124
Negative Sign Bias	1.255556	0.33947
Positive Sign Bias	2.301515**	0.49286
Joint Effect	17.472865***	0.36204
AIC	-6.3940	-6.4041
BIC	-6.3716	-6.3789

From Table 5 we observed that for optimal choice of lags on TGARCH(2,2), the significance of the tests suggests that we reject the null of "there's no leverage effect in the GARCH Model".
Conversely, if we chose to use a EGARCH(2,1), the test results are not significant, so we cannot reject the null. (If we chose EGARCH with order (1,1) or (1,2), the test results for positive size bias test and joint effect is significant at 5 % level, however overall tests under model assumption of EGARCH don't really give significant results).

2.3.3 In-Out of Sample Diagnostics

For **in-sample** diagnostic checking, we report the three tests produced during the fit in Table 6.
For **out-sample** diagnostic checking, after forecasting using the model, we construct the residuals for box tests, at three lag choice of 2, 6, and 10, we cannot reject the null and therefore shows an acceptable fit to the testing sample, which is demonstrated in Table 7.

2.4 JP RiskMetrics and Realised Volatility

The comparison between JP risk metrics and realised volatility is illustrated in Figure 5.
The main conclusion on this comparison is that, for one, the prediction of out-of-sample variance of risk-metrics tend to be slightly higher than the realised volatility, while there exists positive shock around the beginning of 200^{th} and before the ending of 400^{th}. Moreover, the realised volatility is more extreme and turbulent than the prediction given by risk metrics.

Table 6: In-Sample Diagnostic Checking

	EGARCH(2,1)		TGARCH(2,2)	
Weighted Ljung-Box Test on Standardized Residuals				
H0 : No serial correlation, d.o.f=2	statistic	p-value	statistics	p-value
Lag[1]	0.1802	0.6712	0.007878	0.9293
Lag[2*(p+q)+(p+q)-1][5]	1.3975	0.9992	1.302066	0.9997
Lag[4*(p+q)+(p+q)-1][9]	3.4304	0.8152	3.338914	0.8328
Weighted Ljung-Box Test on Standardized Squared Residuals, d.o.f=3				
Lag[1]	0.3233	0.5696	9.847	0.001701
Lag[2*(p+q)+(p+q)-1][8]	1.4288	0.9354	15.872	0.005347
Lag[4*(p+q)+(p+q)-1][14]	3.1656	0.9459	23.563	0.002441
Weighted ARCH LM Tests				
ARCH Lag[4]	0.05445	0.8155	1.120	0.29000
ARCH Lag[6]	0.11486	0.9856	6.412	0.05920
ARCH Lag[8]	0.41109	0.9881	8.270	0.06191

Table 7: Out of the Sample: Box Tests

	TGARCH(2, 2)	eGARCH(2, 1)
p-value at lag 2	0.9316	0.7733
p-value at lag 6	0.1997	0.2394
p-value at lag 10	0.1953	0.2028

3 Forecast and best AR Model

The following picture shows the usage of package “FitAR” as a method to list information criteria as a reference for choosing optimal lag. Our choice of best AR model is $AR(12)$. A demonstration of prediction result is listed in Figure 6. We also plot Garch prediction with TGARCH (2, 2) design (distribution is generalized normal model), which is in Figure 7.

From the comparison, we observed that $ARMA$ prediction is more acceptable, while GARCH model usually gives extreme prediction.

4 Question 4

4.1 Definition:

For a given probability α (in our case, $\alpha = 5\%$. R_t is the logarithmic return and I_t is the information available at time t.

- The Value-at-Risk (VaR) is defined as the solution to

$$P(R_t \leq -VaR_t{}^{(\alpha)}|I_{t-1}) = \alpha \tag{4.1.1}$$

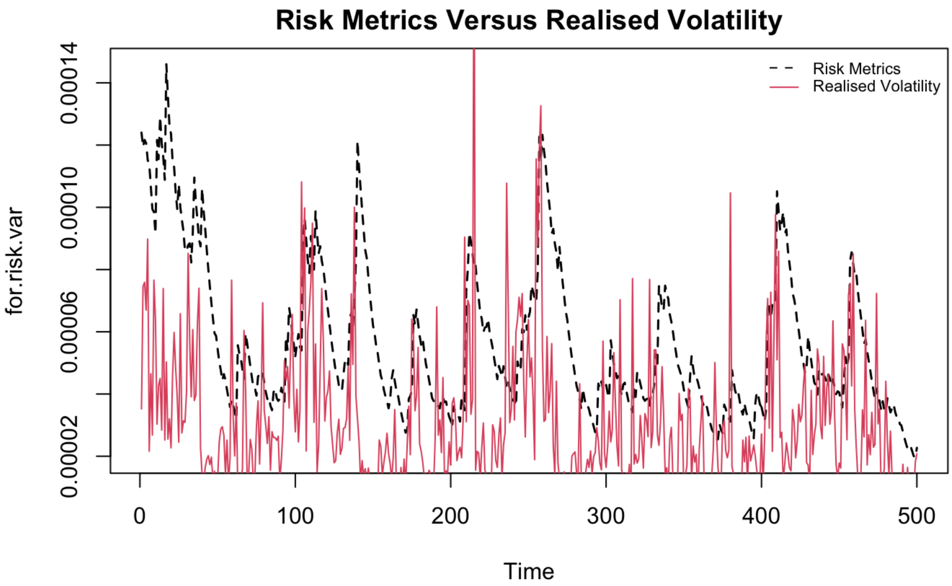


Figure 5: Risk Volatility versus Realised Volatility

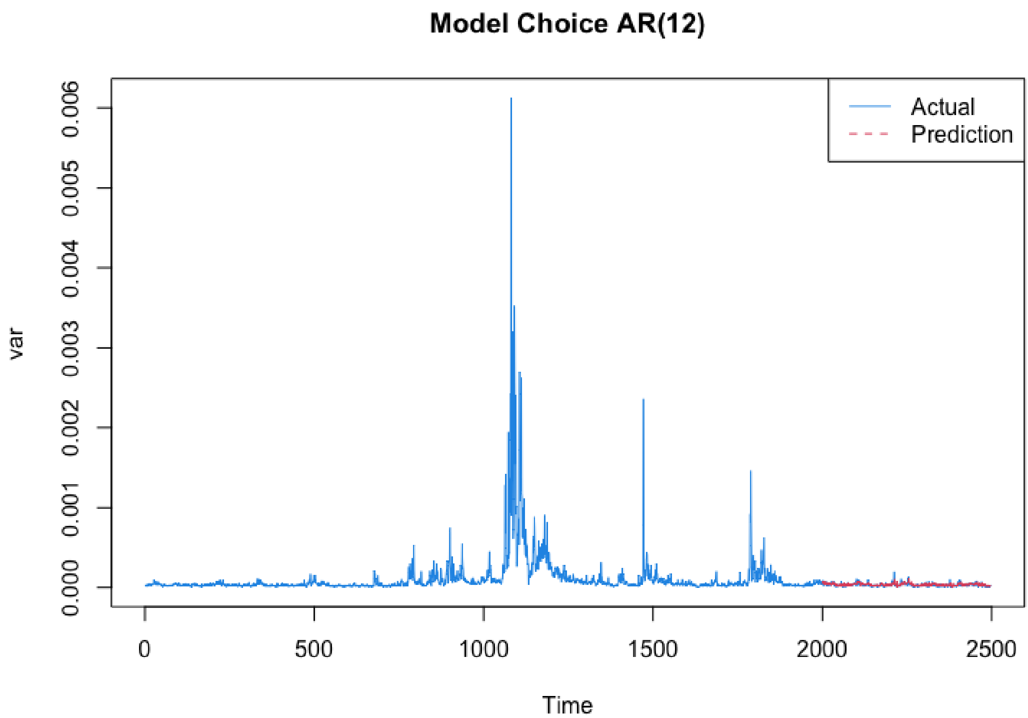


Figure 6: Prediction using model $AR(12)$

- The Expected Shortfall (ES) is defined by

$$ES_t{}^{\alpha} = -E_{t-1}[R_t | R_t < -VaR_t{}^{(\alpha)}] \tag{4.1.2}$$

4.2 Part 1: 2-step ahead VaR/ES

The three models are:
- GARCH(1,1) without leverage and with generalized error distribution
- GARCH(1,1) with leverage and generalized error distribution

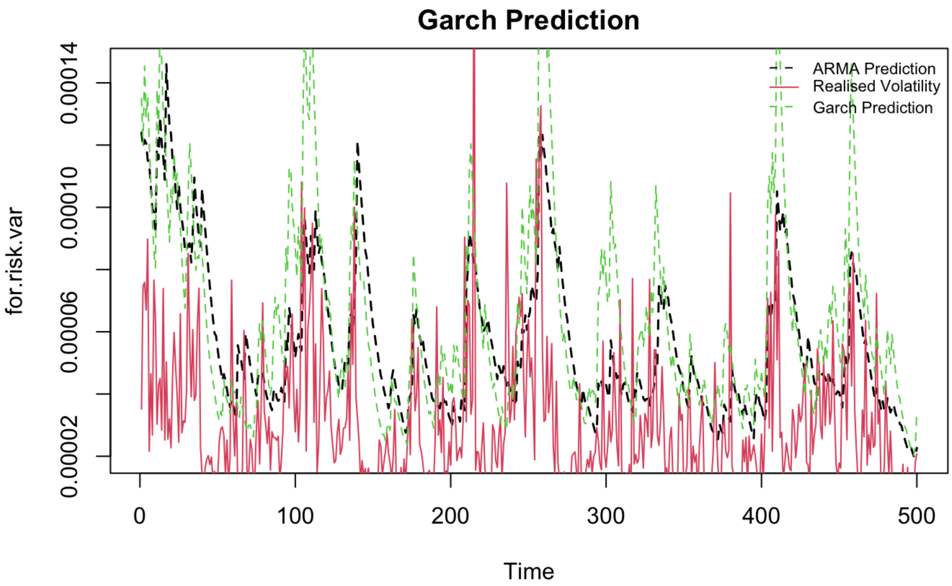


Figure 7: Garch and Realised Volatility Comparison

- GARCH(1,1) with normal distribution

We use the "garchFit" function in the "fGarch" package to get parameters that fit our data.

To calculate VaR, we first refer to R Codes - "Code VaR and ES" on moodle. The manual Garch model function has $h = 2$ and is simulated 100,000 times.
We characterize the forecast of $y_{t+1:t+h}$ and $X_{t:t+h}$, where

$$X_{t:t+h} = aP_t(\exp(\underbrace{y_{t+1} + \cdots + y_{t+h}}_{y_{t+1:t+h}}) - 1), h = 2$$

We computed the 5% conditional Value-at-risk (VaR) and the Expected Shortfall (ES) for each of the three models, and the results are shown in Table 8. VaR is $(-1) \times$ 5%-sample quantile of $y^{(s)}_{t+1:t+2}$ ($X^{(s)}_{t+1:t+2}$). And ES is $(-1)\times$ sample mean of $\left\{y^{(s)}_{t+1:t+2}\right\}$ ($\left\{X^{(s)}_{t+1:t+2}\right\}$). GARCH(1,1) without leverage in this table shows that there is a 5% probability $X_{t:t+h}$ becomes less than -11.51532.

Table 8: VaR Calculation for three models

	GARCH without leverage		GARCH with leverage		i.i.d Normal model	
	VaR	ES	VaR	ES	VaR	ES
Y	0.01158213	0.01522817	0.01346098	0.01731106	0.01226585	0.0159931
X	11.51532	15.10719	13.37079	17.1562	12.19093	15.86004

4.3 Part 2: Calculate VaR for the GARCH models chosen in the previous questions

The chosen models are:
- GARCH(2,2) without leverage with generalized error distribution
- EGARCH(2,1) (with leverage) - generalized error distribution
- GARCH(2,2) without leverage with normal distribution

- EGARCH(2,1) (with leverage) - normal distribution
- GARCH(2,2) without leverage with student distribution
- EGARCH(2,1) (with leverage) - student distribution

We use the "ugarchfit" function in the "rugarch" package to fit the data, then compute the VaR for each model.

Table 9: VaR with different models

Date	GARCH-ged	EGARCH-ged	GARCH-norm
20040615	0.02010449	0.02003795	0.02034452
	0.02008342	0.02001189	0.02033961
	0.01883987	0.01881456	0.01914658
	0.01856293	0.01771977	0.01887087
	0.01787565	0.01698368	0.01818304
20040620	0.01758238	0.01639852	0.01788190
20140606	0.008167640	0.006333695	0.008839695
	0.007849355	0.006249936	0.008564810
	0.007225582	0.007276864	0.007881979
	0.007292521	0.007182925	0.007920576
	0.007042918	0.006921292	0.007585879
20140613	0.008030451	0.007469234	0.008443265

We can see that the VaR of nearly ten years ago is smaller than the VaR of nearly two decades ago.

Next, we derive the "hit sequence" of VaR violations, and apply it to the following three tests to show whether the models are correctly specified:

- First test: The unconditional distribution of H_t is Bernoulli(α)

Likelihood:

$$L(p) = \prod_{t=1}^{T} (1-\pi)^{1-H_t} \pi^{H_t} = (1-\pi)^{T_0} \pi^{T_1} \tag{4.3.1}$$

where T_0 and T_1 are the numbers of zeros and ones in the sample. The unconditional MLE estimator of π is $\hat{\pi} = \frac{T_1}{T}$

The log-likelihood ratio test under the null is given by:

$$LR_{uc} = 2[\log(L(\hat{\pi})) - \log(L(\alpha))] \sim \chi^2(1) \tag{4.3.2}$$

- Second test: The H_t are i.i.d. and Bernoulli(α)

$$\mathit{\Pi} = \begin{bmatrix} 1-\pi_{01} & \pi_{01} \\ 1-\pi_{11} & \pi_{11} \end{bmatrix}$$

where $\pi_{11} = P\left[H_t = 1 \mid H_{t-1} = 1\right]$ and $P\left[H_t = 1 \mid H_{t-1} = 0\right] = \pi_{01}$. The Likelihood of the Markov Chain is:

$$L(\mathit{\Pi}) = (1-\pi_{01})^{T_{00}} \pi_{01}^{T_{01}} (1-\pi_{11})^{T_{10}} \pi_{11}^{T_{11}}$$

And the test under the null is:

$$LR_{cc} = 2[\log L(\hat{\Pi}) - \log L(\alpha)] = LR_{uc} + LR_{\text{ind}} \sim \chi^2(2) \tag{4.3.3}$$

- Third test: Add explanatory variables to explain the violations of VaR
Consider a vector of explanatory variables x_{t-1} and we do the regression and test $b_0 = \alpha$ and $b_1 = 0$

$$H_t = b_0 + x'_{t-1} b_1 + e_{t+1} \tag{4.3.4}$$

In our model, we can test the coefficients of two variables: BV and RJ.

Table 10 summarizes the results of several models in three tests. The first test shows that hypothesis null is rejected in all models. This means that all models cannot satisfy the unconditional distribution of hit sequence is Bernoulli(α). But GARCH(2,2) without leverage and with normal distribution will not be rejected at the 5% level, which means it better. In the second test GARCH(2,2) without leverage and with normal distribution is the only one that matches hypothesis null. The EGARCH model performed particularly poorly. In the third test, all models except EGARCH(2,1)-student satisfy the assumptions of $b_0 = \alpha$ and $b_{RJ} = 0$, but none of the models satisfy $b_{BV} = 0$. On the whole, the normal model is better correctly specified than other distributions. And the GARCH model without leverage performs better than with leverage among the three tests.

Table 10: Three tests for checking the specified models

	GARCH(2,2) (Generalized)	EGARCH(2,1) (Generalized)	GARCH(2,2) (Normal)	EGARCH(2,1) (Normal)	GARCH(2,2) (Student)	EGARCH(2,1) (Student)
First test	The unconditional distribution of H_t is Bernoulli(α)					
	5.419716**	10.68146***	2.952694*	8.08541***	11.24058***	16.18001***
Second test	The H_t are i.i.d and Bernoulli (α)					
	5.607868*	12.21366***	2.981906	8.543059**	11.28649***	16.52184***
Third test	Add explanatory variables to explain the violations of VaR					
H_0: $b_0 = \alpha$	0.4695256	1.5825768	0.0615136	1.2947266	1.4906543	2.35943482**
H_0: $b_{BV} = 0$	1.89385936*	2.52394493**	2.08404598**	2.86419051***	1.5334637	2.27278261**
H_0: $b_{RJ} = 0$	1.5220942	0.1730916	1.3251361	-0.06013348	1.119016	-0.2200258

p<0.1 ; **p<0.05 ; ***p<0.01

5 Code Annex

```
# ========================= #
# Question 1, Summary Stats #
# ========================= #
library(tidyverse)
library(kableExtra)
library(patchwork)
library(moments)
library(anytime)
library(hrbrthemes)
library(psych)
library(bayesforecast)
library(ggplot2)
library(plotly)
library(PerformanceAnalytics)
library(zoo)
library(FitAR)
library(fGarch)
library(data.table) # To transpose dataframe

spy <- read.delim("~/Desktop/spy.txt", header=FALSE)
spy$V1 <- anydate(spy$V1)

names(spy)[1] <- "date"
names(spy)[2] <- "DR"
names(spy)[3] <- "RV"

plot_rd <- spy %>%
  ggplot(aes(x=date, y=DR)) +
    geom_area(fill="blue", alpha=0.5) +
    geom_line(color="blue") +
    ylab("daily␣return") +
    theme_ipsum_es()
plot_rd <- ggplotly(plot_rd)
plot_rd

spy %>%
  summarise(Caption="Summary␣Stats␣for␣Daily␣Return",
            MeanDR=mean(spy$DR),
            VarDR=var(spy$DR),
            N=n(),
            SkeDR=moments::skewness(spy$DR),
            KurDR=moments::kurtosis(spy$DR)-3)

diff_dr <- spy$DR-mean(spy$DR)
n <- length(diff_dr)
diff_dr_cub <- diff_dr^3
s <- sum(diff_dr_cub)
skewness_dr <- (var(spy$DR)^(3/2)*n)^(-1)*s
skewness_dr
diff_df_4 <- diff_dr^4
k <- sum(diff_df_4)
kurtosis_dr <- (var(spy$DR)^(2)*n)^(-1)*k-3
```

```r
kurtosis_dr

ggacf(spy$DR) + theme_bw()

spy$DRsq <- spy$DR^2
plot_rd_sq <- spy %>%
  ggplot(aes(x=date, y=DRsq)) +
    geom_area(fill="yellow", alpha=0.5) +
    geom_line(color="yellow") +
    ylab("daily return Squred") +
    theme_ipsum_es()
plot_rd_sq <- ggplotly(plot_rd_sq)
plot_rd_sq

ggacf(spy$DRsq)+theme_bw()

m30 <- apply.rolling(as.ts(spy$DR), 30, FUN="mean")
v30 <- apply.rolling(as.ts(spy$DR), 30, FUN="var")
s30 <- apply.rolling(as.ts(spy$DR), 30, FUN="skewness")
k30 <- apply.rolling(as.ts(spy$DR), 30, FUN="kurtosis")
par(mfrow=c(2,2))
plot.ts(m30)
plot.ts(v30)
plot.ts(s30)
plot.ts(k30)

m120 <- apply.rolling(as.ts(spy$DR), 120, FUN="mean")
v120 <- apply.rolling(as.ts(spy$DR), 120, FUN="var")
s120 <- apply.rolling(as.ts(spy$DR), 120, FUN="skewness")
k120 <- apply.rolling(as.ts(spy$DR), 120, FUN="kurtosis")
par(mfrow=c(2,2))
plot.ts(m120)
plot.ts(v120)
plot.ts(s120)
plot.ts(k120)

m365 <- apply.rolling(as.ts(spy$DR), 365, FUN="mean")
v365 <- apply.rolling(as.ts(spy$DR), 365, FUN="var")
s365 <- apply.rolling(as.ts(spy$DR), 365, FUN="skewness")
k365 <- apply.rolling(as.ts(spy$DR), 365, FUN="kurtosis")
par(mfrow=c(2,2))
plot.ts(m365)
plot.ts(v365)
plot.ts(s365)
plot.ts(k365)


# ======================= #
# Question 2, (a) and (b) #
# ======================= #

# ============= Method 1 of MLE ================ #

R_t2 <- numeric()
condvar2 <- numeric()
```

```
omega2 <- 0.01
alpha2 <- 0.05
beta2 <- 0.6
psi16 <- 2
condvar2[1] <- (omega2+alpha2*psi16^2) / (1-alpha2-beta2)

observation_generation <- function(n){
  for (i in 1:n){
    R_t2[i] <- rnorm(1,0,condvar2[i])
    condvar2[i+1] <- omega2 + alpha2*(R_t2[i]-psi16)^2 + beta2*condvar2[i]
  }
  return(R_t2[250:n])
}

# MLE calculations that takes 2 arguments, psi and y
myfunction2 <- function(psi,y){

  mu = psi[1]
  omega = exp(psi[2]) # exp so that omega>0
  alpha = exp(psi[3])
  beta = exp(psi[4]) # alpha + beta  < 1
  theta = exp(psi[5])

  if (alpha + beta > 1){
    LT = -Inf  # in order to put alpha + beta < 1
  } else {

  n = length(y) # sample size

  ell = numeric(n) # vector of size n
  condvar = numeric(n) # vector containing sigma_t^2
  condvar[1] = (omega+alpha*theta^2)/(1-alpha-beta)

  for (t in 2:n){
    ell[t] = dnorm(y[t], mean = 0, sd=sqrt(condvar[t-1]),log=T)
    condvar[t] = omega + alpha * (y[t-1]-theta)^2 + beta * condvar[t-1]
  }
  LT = sum(ell)
  }
  return(-LT)
}

psi1 = c(0, log(0.01), log(0.05), log(0.6), log(2))


# Replications, 100 times first to try
RR <- data.frame(replicate(n=5, optim(par=psi1, myfunction2,
                                      y=observation_generation(1000),
                                      control=list(maxit=9000000))$par))
#n_seq <- 1:2
#colnames(RR) <- paste("n", n_seq, sep = "_")
#rownames(RR) <- c("mu", "omega", "alpha", "beta", "theta")
#RR <- transpose(RR)

# ============ Method 2 ============ #
```

```r
Garch_DG <- function (n, mu, omega, alpha, beta, theta) {
  burn = 250
  n = n + burn

  z = d.g(n)

  h = rep(0, n)
  r = rep(0, n)

  for (t in 2:n) {
    h[t] = omega + alpha * r[t-1]^2 + beta * h[t-1]
    r[t] = mu + z[t] * sqrt(h[t])
  }
  return(r[burn:n])

}
# Run the following to check the generation
# r1 <- Garch_DG(1000, 0, 0.01, 0.05, 0.9, 0)

garchInit = function(repli) {
  Mean = mean(repli)
  Var = var(repli)
  S = 1e-6
  psi1 = c(mu = Mean, omega = 0.1*Var, alpha = 0.1, beta = 0.9, theta = 0)
  lowerBounds = c(mu = -10*abs(Mean),
                  omega = S^2, alpha = S, beta = S, theta = -3)
  upperBounds = c(mu = 10*abs(Mean),
                  omega = 100*Var, alpha = 1-S, beta = 1-S, theta = 3)
  cbind(psi1=psi1, lowerBounds=lowerBounds, upperBounds=upperBounds)
}

# garchInit(r1)
# garchInit(r1)[,1]

Garch_NLL = function(psi, repli){
  mu = psi[1]; omega = psi[2]; alpha = psi[3]; beta = psi[4]; theta = psi[5]
  z = (repli - mu)
  Mean = mean((z^2))

  e = omega + alpha*c(Mean, (z[-length(repli)] - theta)^2)
  h = timeSeries::filter(e, beta, "r", init = Mean)
  hh = sqrt(abs(h))
  llh = -sum(log(g.d(z, hh)))
}

Garch_Optim = function(repli, psi1, lowerBounds, upperBounds){
  fit = nlminb(start = psi1, objective = Garch_NLL,
               lower = lowerBounds,
               upper = upperBounds,
               control = list(trace=10),
               repli = repli)
}

# Change below for different number of replications
```

```
R <- 10

# Change below for different number of observations
O <- 100

# Change below for different designs
# Design Indicator D, D=1 for first and otherwise the second
D <- 2

# Change below for different distribution of zt
# Indicator for dist. of zt (~D(0,1)/~T(10), therefore "norm" or "std")
dist = "norm"

# Distribution for later MLEs
Garch_Dist <- dist

# Change below for different choice of variance process
vp <- 2

# parameter subcases
if (D == 1){
  mu = 0
  omega = 0.01
  alpha = 0.05
  beta = 0.9
  theta = 0
}

if (D == 2){
  mu = 0
  omega = 0.01
  alpha = 0.05
  beta = 0.6
  theta = 2
}

# distribution of zt, subcases
if (Garch_Dist == "norm"){
  d.g = function(n){
    rnorm(n,0,1)
  }
}

if (Garch_Dist == "std"){
  d.g = function(n){
    rt(n, df=10)
  }
}

# variances process subcases
if (vp == 1){
  v_eq = function(omega, alpha, Mean, z, repli){
    omega + alpha*c(Mean, (z[-length(repli)])^2)
  }
}
```

```
if (vp == 2){
  v_eq = function(omega, alpha, Mean, z, repli, theta){
    omega + alpha*c(Mean, (z[-length(repli)] - theta)^2)
  }
}

if (Garch_Dist == "norm"){
  g.d = function(z, hh){
    dnorm(x = z/hh)/hh
    }
}

if (Garch_Dist == "std"){
  g.d = function(z, hh){
    dt(x=z/hh, df=10)/hh
    }
}

mat = matrix(0, nrow = R, ncol = 5)

for (i in 1:R){


  r = Garch_DG(n = n,
               mu = mu, omega = omega, alpha = alpha,
               beta = beta, theta = theta)
  r = ts(r)

  RR1 = garchInit(r)
  psi1 = RR1[,1]
  lowerBounds = RR1[,2]
  upperBounds = RR1[,3]

  fit = Garch_Optim(r, psi1, lowerBounds, upperBounds)

  mat[i,] = fit$par
}


mest = colMeans(mat)
sdst = colSds(mat)

mest
sdst


# ================================== #
# 2.c Fit GARCH                      #
# 2.c.1 Model Selection              #
# 2.c.2 Leverage Effect              #
#                                    #
# Garch_Model_Selection Function:    #
#                                    #
# With ARMA(1,1) Assumption;         #
# Input takes a choice of garch.model #
```

```
# among baseline, TGARCH and EGARCH    #
# and garch.order (1:2, 1:2)           #
# =================================== #

Garch_Model_Selection <- function(garch.model, garch.order){
  mean.model = c(1,1);
  dist.choice = "norm";
  R = spy$DR;
  out.sample = length(R)/5;
  if (garch.model == "GARCH" | garch.model == "TGARCH") {
    spec.selection <- ugarchspec(variance.model = list(model = "fGARCH",
                                                       garchOrder=garch.order,
                                                       submodel = "GARCH"),
                                 mean.model = list(include.mean=TRUE,
                                                   armaOrder=mean.model),
                                 distribution.model = dist.choice)
    fit.outcome <- ugarchfit(spec = spec.selection,
                             data = R, out.sample = out.sample)
  }
  else if (garch.model == "eGARCH"){
    spec.selection = ugarchspec(variance.model = list(model = garch.model,
                                                      garchOrder=garch.order),
                                mean.model = list(include.mean=TRUE,
                                                  armaOrder=mean.model),
                                distribution.model = dist.choice)
    fit.outcome = ugarchfit(spec = spec.selection,
                            data = R, out.sample = out.sample)
  }
  return(fit.outcome)
}
Garch_Model_Selection("GARCH", c(1,1))
Garch_Model_Selection("GARCH", c(1,2))
Garch_Model_Selection("GARCH", c(2,1))
Garch_Model_Selection("GARCH", c(2,2))

# Testing for Leverage Effect, start using TGARCH and eGARCH
# Based on the previous selection result we chose GARCH(2,2)

Garch_Model_Selection("TGARCH", c(1,1))
Garch_Model_Selection("TGARCH", c(1,2))
Garch_Model_Selection("TGARCH", c(2,1))
Garch_Model_Selection("TGARCH", c(2,2))

# For TGARCH choose order (2,2)

Garch_Model_Selection("eGARCH", c(1,1))
Garch_Model_Selection("eGARCH", c(1,2))
Garch_Model_Selection("eGARCH", c(2,1))
Garch_Model_Selection("eGARCH", c(2,2))

# For eGARCH choose order (2,1)

Garch_Model_Selection("TGARCH", c(2,2))
Garch_Model_Selection("eGARCH", c(2,1))
```

```r
# ============================== #
# Out of the Sample Diagnostics #
# ============================== #

model.t <- Garch_Model_Selection("TGARCH",
                        c(2, 2))
model.e <- Garch_Model_Selection("eGARCH",
                        c(2, 1))
model.t1 <- ugarchforecast(model.t,
                           data=NULL,
                           n.head=1,
                           n.roll=499,
                           out.sample=500)
model.t2 <- ugarchforecast(model.e,
                           data=NULL,
                           n.head=1,
                           n.roll=499,
                           out.sample=500)
length(spy$DR)/5

y = spy$DR[1998:length(spy$DR)]
yhat1 = model.t1@forecast$seriesFor[1,]
epsilon1 = y - yhat1
sigma.hat1 = model.t1@forecast$sigmaFor[1,]

z1 = epsilon1 /sigma.hat1

yhat2 = model.t2@forecast$seriesFor[1,]
epsilon2 = y - yhat2
sigma.hat2 = model.t2@forecast$sigmaFor[1,]

z2 = epsilon2 /sigma.hat2

Box.test(z1, lag=2, type = "Ljung-Box")
Box.test(z1, lag=6, type = "Ljung-Box")
Box.test(z1, lag=10, type = "Ljung-Box")

Box.test(z2, lag=2, type = "Ljung-Box")
Box.test(z2, lag=6, type = "Ljung-Box")
Box.test(z2, lag=10, type = "Ljung-Box")

# ============ #
# Risk Metrics #
# ============ #

rm.spec = ugarchspec(mean.model=list(armaOrder=c(1,1), include.mean=TRUE),
                              variance.model=list(model="TGARCH"),
                              distribution.model = "ged",
                              fixed.pars = list(omega = 0))
rm.model = ugarchfit(spec = rm.spec, data = spy$DR, out.sample = 500)
rm.pred = ugarchforecast(rm.model,
                                data=NULL,
                                n.head=1,
                                n.roll = 499,
                                out.sample = 500)
```

```
rm.pred = as.ts(rm.pred@forecast$sigmaFor[1,])^2
rv.comp = as.ts(dat$RV[1998:2497])

plot(rm.pred, col="1", lty= 2, lwd = 1.5, main="Risk␣Metrics␣Versus␣Realised␣Volatility")
lines(rv.comp, col="2", lty=1.5, lwd = 1)
l1 <- as.expression("Risk␣Metrics")
l2 <- as.expression("Realised␣Volatility")
legend("topright",
       c(l1, l2),
       col=c(1, 2),
       lty=c(2, 1.5))

# ================================== #
# Question 3: Best AR and Comparison #
# ================================== #

library(FitAR)
library(forecast)


selection <- SelectModel(as.ts(spy$RV[1:1997]), lag.max = 14, ARModel = "AR",
                         Criterion = "AIC", Best = 6); selection


fit.1 <- arima(spy$RV, order = c(12,0,0))
fit.2 <- Arima(as.ts(spy$RV), model = fit1)
fc <- window(fitted(fit.2), start = 1998)


plot(as.ts(spy$RV), col = 4, main = "Model␣Choice␣AR(12)",
     ylab = "var", lty = 1)
lines(fc, col = 2, lty = 1)
legend("topright", c("Actual", "Prediction"), col=c(4,2),
       lty=c(1,2))




# ======================== #
# Question 4, Value-at-risk #
# ======================== #
#1. Three models and VaR when h=2. Similar way as that on moodle.

library(fGarch)

myfit = garchFit(~garch(1,1), data=R, leverage=NULL, cond.dist="ged")
#myfit = garchFit(~garch(1,1), data=R, leverage=TRUE, cond.dist="ged")
#myfit = garchFit(~garch(1,1), data=R, cond.dist="norm")

results = myfit@fit
results = results$coef

cc = results[1]
omega = results[2]
alpha = results[3]
```

```
beta = results[4]


condvol = myfit@sigma.t # conditional volatility (sqrt(variance))

plot.ts(condvol)


# Simulation


S= 100000

yt = R[n]
sigmat = sqrt(omega + alpha * (yt-cc)^2 + beta * condvol[n]^2) # cond.vol for y_{t+1}

yforecast = numeric(S)
Xforecast = numeric(S)

set.seed(1230910)
for (s in 1:S){
  z = rnorm(n=2)
  y1= cc + sigmat * z[1]

  sigmatplusone = sqrt(omega + alpha * (sigmat * z[1])^2 + beta * sigmat^2)
  y2 = cc + sigmatplusone * z[2]
  yforecast[s] = y1 + y2
  Xforecast[s] = 100 * 10 * (exp(y1+y2)-1)
}

hist(yforecast,20)
hist(Xforecast,30)


# Compute VaR of level 5%

q005 = quantile(yforecast, 0.05)
VaR_y = - q005

q005X = quantile(Xforecast, 0.05)
VaR_X = -q005X


# Compute ES of level 5%

subsety = yforecast[yforecast<q005]
length(subsety) #50 because 1000 * 0.05 = 50
hist(subsety,20)
ES_y = - mean(subsety)

subsetX = Xforecast[Xforecast < q005X]
length(subsetX) #5000 because 100000 * 0.05
hist(subsetX,20)
ES_X = - mean(subsetX)
```

```r
# Results

c(VaR_y, ES_y)

c(VaR_X, ES_X)


# 2. VaR & Violations of the VaR

# Preparation of Functions and Specifications

spec1<-ugarchspec(variance.model = list(model = "fGARCH",
                                        garchOrder=c(2,2),
                                        submodel = "GARCH"),
                  distribution.model = "ged")


spec2<-ugarchspec(variance.model = list(model = "eGARCH",
                                        garchOrder=c(2,1),
                                        submodel = "GARCH"),
                  distribution.model = "ged")


spec3<-ugarchspec(variance.model = list(model = "fGARCH",
                                        garchOrder=c(2,2),
                                        submodel = "GARCH"),
                  distribution.model = "norm")


spec4<-ugarchspec(variance.model = list(model = "eGARCH",
                                        garchOrder=c(2,1),
                                        submodel = "GARCH"),
                  distribution.model = "norm")

spec5<-ugarchspec(variance.model = list(model = "fGARCH",
                                        garchOrder=c(2,2),
                                        submodel = "GARCH"),
                  distribution.model = "std")


spec6<-ugarchspec(variance.model = list(model = "eGARCH",
                                        garchOrder=c(2,1),
                                        submodel = "GARCH"),
                  distribution.model = "std")


hitseq <- function(myfit, alpha = 0.05) {
  VaR = -quantile(myfit, alpha)
  # Hit Sequence
  n = length(R[1:2497])
  hit = rep(0, n)
  for (i in 1:2497){
    if(R[i] < -VaR[i]){
      hit[i] = 1
    }
```

```
  }

  return(hit)
}

#Test 1: Uncondtional Ber(alpha) ####

test1uc = function(hit, alpha = 0.05){
  n = length(hit)
  T1 = length(which(hit==1))
  T0 = n - T1
  pi = T1 / n
  Lp = (1-pi)^T0 * pi^T1
  La = (1-alpha)^T0 * alpha^T1

  #LR unconditional
  LRuc = 2*(log(Lp) - log(La))
  LRuc
}

#Test 2: Independence ####

test2ind = function(hit, alpha = 0.05){
  # T00
  n = length(hit)
  T00 = 0
  for (i in 1: (n-1)){
    if (hit[i] == 0 & hit[i+1] == 0){
      T00 = T00 + 1
    }
  }
  # T01
  T01 = 0
  for (i in 1: (n-1)){
    if (hit[i] == 0 & hit[i+1] == 1){
      T01 = T01 + 1
    }
  }
  # T10
  T10 = 0
  for (i in 1: (n-1)){
    if (hit[i] == 1 & hit[i+1] == 0){
      T10 = T10 + 1
    }
  }
  # T11
  T11 = 0
  for (i in 1: (n-1)){
    if (hit[i] == 1 & hit[i+1] == 1){
      T11 = T11 + 1
    }
  }

  # Likelihood of pi
```

```
  pi.01 = T01 / (T00 + T01)
  pi.11 = T11 / (T10 + T11)
  pi.00 = 1 - pi.01
  pi.10 = 1 - pi.11

  L.pi2 = (1 - pi.01)^T00 * pi.01^T01 * (1-pi.11)^T10 * pi.11^T11
  L.alpha2 = (1 - alpha)^T00 * alpha^T01 * (1-alpha)^T10 * alpha^T11
  LRcc = 2 * ( log(L.pi2) - log(L.alpha2) )
  LRcc
}

#Test 3: Add explanatory variables###
ttest <- function(reg, coefnum, val){
  tstat <- (coef(summary(reg))[coefnum,1]-val)/coef(summary(reg))[coefnum,2]
  pvalue <- 2 * pt(abs(tstat), reg$df.residual, lower.tail = FALSE)
  return(c(tstat, pvalue))
}

library(lmtest)
test3add = function(hit, set) {
  n = length(hit)
  T = length(data[,4])-1

  test3 = lm(hit[2:length(hit)] ~ data[,4][1:2496] + data[,5][1:2496])

  print("b0=alpha")
  print(ttest(test3, 1, 0.05))
  print("b1␣=␣0")
  print(ttest(test3, 2, 0))
  print("b2␣=␣0")
  print(ttest(test3, 3, 0))
}


myfit1<-ugarchfit(spec=spec1, data = R)
myfit2<-ugarchfit(spec=spec2, data = R)
myfit3<-ugarchfit(spec=spec3, data = R)
myfit4<-ugarchfit(spec=spec4, data = R)
myfit5<-ugarchfit(spec=spec5, data = R)
myfit6<-ugarchfit(spec=spec6, data = R)


#Calculate VaR
VaR1 = -quantile(myfit1, 0.05)
VaR2 = -quantile(myfit2, 0.05)
VaR3 = -quantile(myfit3, 0.05)
VARTABLE<-cbind.data.frame(VaR1,VaR2,VaR3)
head(VARTABLE)
tail(VARTABLE)

#Hit Sequence
hit1<-hitseq(myfit1,0.05)
hit2<-hitseq(myfit2,0.05)
hit3<-hitseq(myfit3,0.05)
hit4<-hitseq(myfit4,0.05)
```

```r
hit5<-hitseq(myfit5,0.05)
hit6<-hitseq(myfit6,0.05)

#Test 1
test1uc(hit1)
test1uc(hit2)
test1uc(hit3)
test1uc(hit4)
test1uc(hit5)
test1uc(hit6)

#Test 2
test2ind(hit1)
test2ind(hit2)
test2ind(hit3)
test2ind(hit4)
test2ind(hit5)
test2ind(hit6)

#Test 3
test3add(hit1)
test3add(hit2)
test3add(hit3)
test3add(hit4)
test3add(hit5)
test3add(hit6)
```