STAT0011: Decision and Risk

In-Course Assessment (ICA) 2024

Group 2. 2024-03-27

Contents

Individual contributions:

21006650

Developed ChatGPT prompts & output in Task 2

Constructed graphs for comparison and verification in Task 2

21021177

Developed ChatGPT prompts & output in Task 2

Analyze information and plan the report structure

20017991

Participated in the generation and modification of part of the Task 2 code.

Compared the accuracy and summerize reasons.

Write and finalize the report with review and edition.

20033987

Write Task1 code

Task 2 adjusted and check code for implementation in R Markdown

Task 2 adjusted layout and formatting of the R Markdown-generated document

21025328

Write and finalize the report with review and edition

21073015

Data processing

Write Task 1 code

Constructed graphs for comparison and verification in Task 1 and 2

Write and finalize Task 1 and Task 2 report, providing comments to each step and conclusion

- **TASK 1**

Our report consists of five parts.

1.Data Overview

2.Model Selection

3.Constructing Copula

4.Reintroducing AR-GARCH effects

5.Computing Vale-at-Risk

1. Data Overview

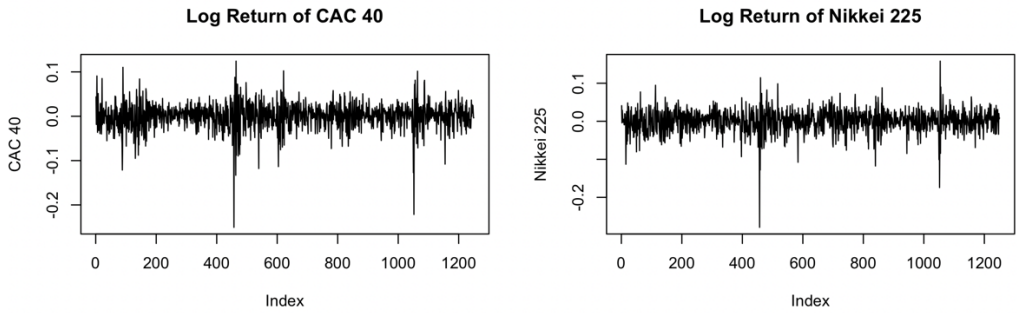
1.1 Data processing

In this task, we selected CAC40 and Nikkei 225 two stock price indicies to investigate. We downloaded their weekly adjusted data from yahoo.com and combined them into one file named as CAC40_Nk225.csv. Note that we deleted one row of N/A value artificially.

```
data <- read.csv("CAC40_Nk225.csv")
CAC40 <- data$CAC40
Nk225 <- data$Nikkei225
rt_CAC40 <- diff(log(CAC40))
rt_Nk225 <- diff(log(Nk225))
```

1.2 Data Overview

We would like to have a look at the data first.

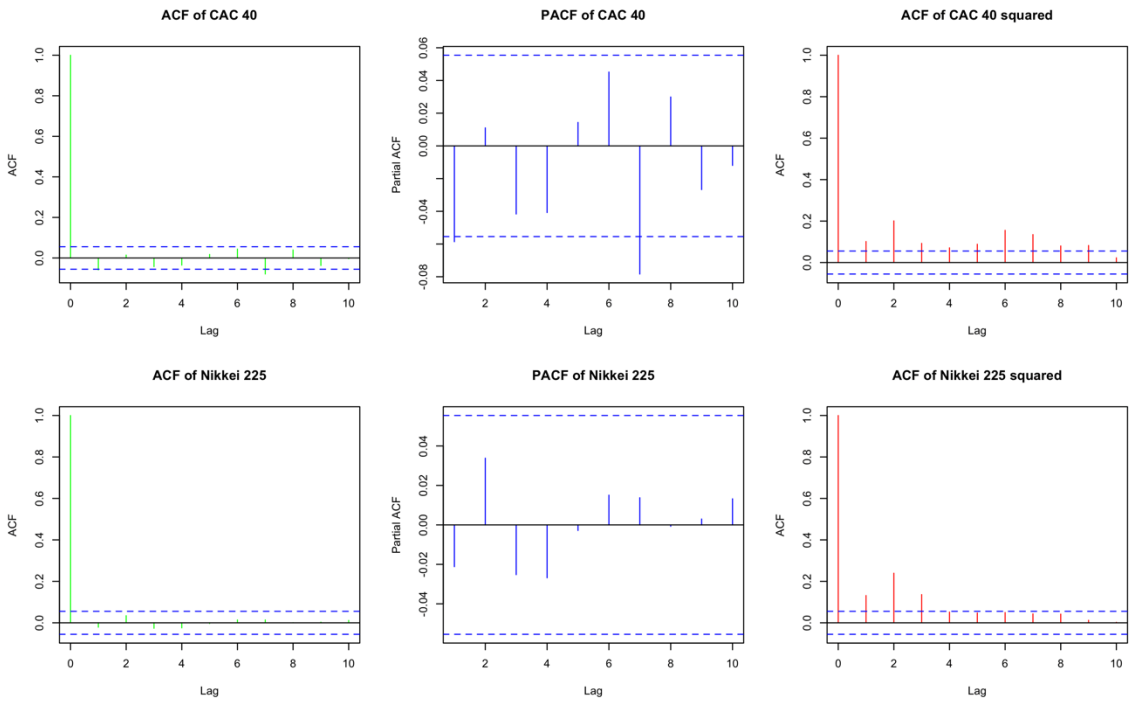


2. Model Selection

2.1 Draw time series plots

```
par(mfrow=c(2,3))
acf(rt_CAC40,lag.max= 10, col="green",main="ACF of CAC 40") # Observe t
he first 10 lags only.
```

```
pacf(rt_CAC40,lag.max= 10, col="blue",main="PACF of CAC 40")
acf(rt_CAC40^2,lag.max= 10, col="red",main="ACF of CAC 40 squared")
acf(rt_Nk225,lag.max= 10, col="green",main="ACF of Nikkei 225")
pacf(rt_Nk225,lag.max= 10, col="blue",main="PACF of Nikkei 225")
acf(rt_Nk225^2,lag.max= 10, col="red",main="ACF of Nikkei 225 squared")
```



It is observed that, for CAC 40 indices, all lags in the ACF plot lie within the significance level apart from the first lag. The tail off pattern is significant. Its pacf graph represents a cut off at lag 1. Therefore, an AR(1) model is suggested. In terms of Nikkei 225, the exponential decay in the acf graph is significant. However, we couldn't detect any cut off in the pacf graph, and the autocorrelation test above indicates an independence among data. No AR model is suggested. The acf plots of squared returns for both indices reveal some serial correlation among variances with time, which suggest GARCH models for their conditional variances.

2.2 Test for independency and normality

```
Box1 <- Box.test(rt_CAC40, lag = 10, type = c("Ljung-Box"), fitdf = 1)
Box2 <- Box.test(rt_Nk225, lag = 10, type = c("Ljung-Box"), fitdf = 1)
Autocorrelation=rbind(c(Box1$p.value,Box2$p.value))
rownames(Autocorrelation) <- c("p-value")
colnames(Autocorrelation) <- c("CAC40 Log Return","Nikkei225 Log Return
")
Autocorrelation
```

```
##         CAC40 Log Return Nikkei225 Log Return
## p-value      0.006342594             0.884528
```

```
##
## Title:
##  Jarque - Bera Normalality Test
##
## Test Results:
##   STATISTIC:
##     X-squared: 3402.3719
##   P VALUE:
##     Asymptotic p Value: < 2.2e-16

##
## Title:
##  Jarque - Bera Normalality Test
##
## Test Results:
##   STATISTIC:
##     X-squared: 3492.4335
##   P VALUE:
##     Asymptotic p Value: < 2.2e-16
```

The p-values of log-returns for CAC 40 and Nikkei 225 are 0.00634 and 0.8845 respectively. The p-value of CAC 40 lies below the significance level, which indicates an autocorrelation for CAC 40 data. On the other hand, Nikkei 225 stock prices are considered as independently distributed. This verifies our guess for models above.

The p-values of Jarque-Bera tests for normality are extremely small, indicating that our data does not follow a normal distribution.

2.3 Model construction

2.3.1 We would like to build an AR(1)+GARCH(p,q) model for CAC40

```
results <- data.frame(order = character(), AIC = numeric(), Distributio
n = character(), stringsAsFactors = FALSE) #Construct an empty data fra
me to store the results

distributions <- c("norm","snorm", "ged", "sged", "std", "sstd")
for (i in 1:3) {
  for (j in 1:3) {
    for (dist in distributions) {
      # Construct the formula for current GARCH model order
      formula <- as.formula(paste("~ arma(1,0) + garch(", i, ",", j, ")
", sep = ""))

      # Fit the GARCH model with the current order and distribution
      model <- garchFit(formula = formula, data = rt_CAC40, trace = F,
cond.dist = dist)

      # Extract AIC values
      aic_value <- model@fit$ics["AIC"]
```

```
      # Add the results to dataframe
      results <- rbind(results, data.frame(order = paste("garch(", i, ",
", j, ")", sep = ""),
                                           AIC = aic_value, Distributio
n = dist))


    }
  }
}

ordered_results_CAC40 <- results[order(results$AIC), ]# Order the resul
ts by AIC
best_model_CAC40 <- head(ordered_results_CAC40, 3)# Output the best 3 m
odels with the smallest values of AIC.
print(best_model_CAC40)

##            order       AIC Distribution
## AIC5  garch(1,1) -4.510153         sstd
## AIC35 garch(2,3) -4.509394         sstd
## AIC11 garch(1,2) -4.508898         sstd

rm(results, formula, model, aic_value) # Remove repeated data
```

By comparing AIC values, it is observed that AR(1)+GARCH(1,1) is the best model for CAC40, with the lowest AIC of -4.510153. This model follows the standardized skew Student-t distribution.

2.3.2 We would like to build a GARCH(p,q) model for Nikkei 225.

```
results <- data.frame(order = character(), AIC = numeric(), Distributio
n = character(), stringsAsFactors = FALSE) #Construct an empty data fra
me to store the results

distributions <- c("norm","snorm", "ged", "sged", "std", "sstd")
for (i in 1:3) {
  for (j in 1:3) {
    for (dist in distributions) {
      # Construct the formula for current GARCH model order
      formula <- as.formula(paste("~garch(", i, ",", j, ")", sep = ""))

      # Fit the GARCH model with current order and distribution
      model <- garchFit(formula = formula, data = rt_Nk225, trace = F,
cond.dist = dist)

      # Extract AIC values
      aic_value <- model@fit$ics["AIC"]

      # Add the results to dataframe
      results <- rbind(results, data.frame(order = paste("garch(", i, ",
", j, ")", sep = ""),
```

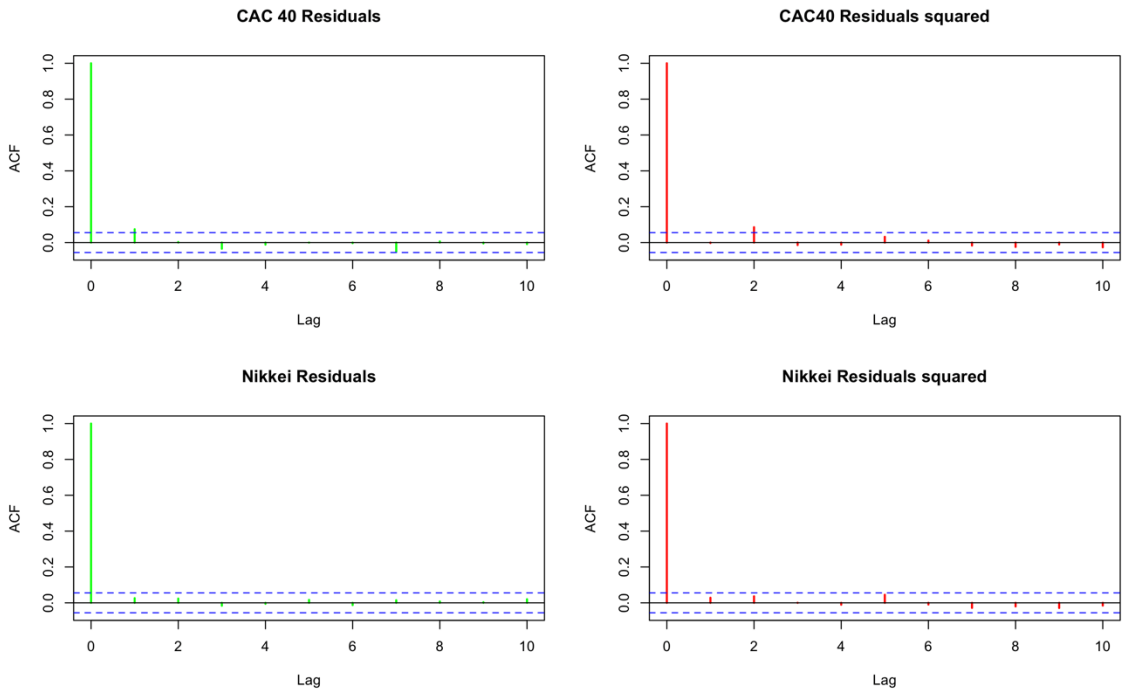
```
                                                AIC = aic_value, Distributio
n = dist))


    }
  }
}

ordered_results_Nk225 <- results[order(results$AIC), ]# Order the resul
ts by AIC
best_model_Nk225 <- head(ordered_results_Nk225, 3)# Output the best 3 m
odels with the smallest values of AIC.
print(best_model_Nk225)

##            order       AIC Distribution
## AIC5  garch(1,1) -4.363542         sstd
## AIC11 garch(1,2) -4.362535         sstd
## AIC23 garch(2,1) -4.362150         sstd

rm(results, formula, model, aic_value) # Remove repeated data
```

By comparing AIC values, it is observed that GARCH(1,1) is the best model for Nikkei, with the lowest AIC of -4.363542. This model follows the standardized skew Student-t distribution.

Rename best fit models for CAC 40 and Nikkei 225 as model_CAC40 and model_Nk225 respectively.

2.3.3 Check residuals.

```
# Check residuals.
res1 <- residuals(model_CAC40, standardize=TRUE)
res2 <- residuals(model_Nk225, standardize=TRUE)

par(mfrow=c(2,2))
acf(res1, main= "CAC 40 Residuals", lag.max= 10, col="green", lwd=2)
acf(res1^2, main= "CAC40 Residuals squared", lag.max= 10, col="red", lw
d=2)
acf(res2, main= "Nikkei Residuals", lag.max= 10, col="green", lwd=2)
acf(res2^2, main= "Nikkei Residuals squared", lag.max= 10, col="red", l
wd=2)

par(mfrow=c(1,1))
```



These graphs demonstrate ACF plots of residuals and residuals squared for two models. It is obvious that all plots appear a white noise pattern with lags lie within the significance level. This indicates that residuals appear randomly, and model_CAC40 and model_Nk225 are of good fit.

```
Box.test(res1, lag = 10, type = c("Ljung-Box"), fitdf = 1)

##
##  Box-Ljung test
##
## data:  res1
## X-squared = 12.077, df = 9, p-value = 0.209

Box.test(res1^2, lag = 10, type = c("Ljung-Box"), fitdf = 1)

##
##  Box-Ljung test
##
## data:  res1^2
## X-squared = 13.201, df = 9, p-value = 0.1537

Box.test(res2, lag = 10, type = c("Ljung-Box"), fitdf = 1)

##
##  Box-Ljung test
##
## data:  res2
## X-squared = 3.3653, df = 9, p-value = 0.948
```

```
Box.test(res2^2, lag = 10, type = c("Ljung-Box"), fitdf = 1)
```

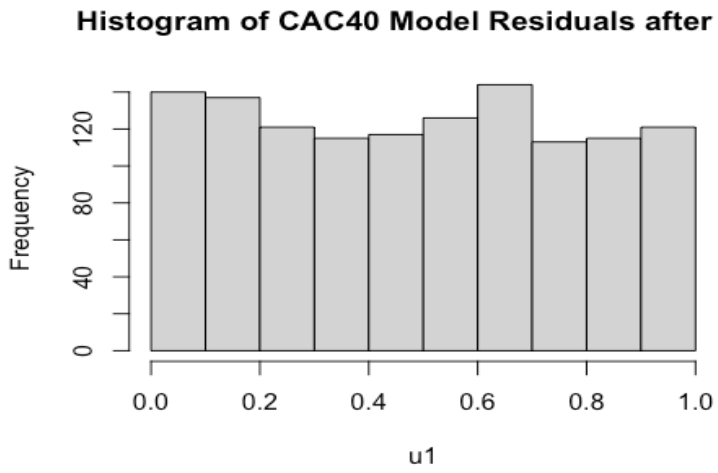
```
##
##  Box-Ljung test
##
## data:  res2^2
## X-squared = 8.3061, df = 9, p-value = 0.5036
```

The two Ljung-Box tests conduct p-values of 0.209 and 0.1537 for model_CAC40, and conduct p-values of 0.948 and 0.5036 for model_Nk225.All p-values are higher than 0.05. Hence, there is no evidence to reject the null hypothesis that residuals are independently distributed. model_CAC40 and model_Nk225 pass Ljung-Box tests.

3. Constructing Copula

3.1 Probability integral transformation (PIT)

```
# CAC 40
# Check histogram
u1<-psstd(res1, mean=0, sd=1, nu=coef(model_CAC40)["shape"], xi=coef(mo
del_CAC40)["skew"])[2:length(rt_CAC40)]
hist(u1,  main = "Histogram of CAC40 Model Residuals after PIT") # Fair
ly uniform.
```



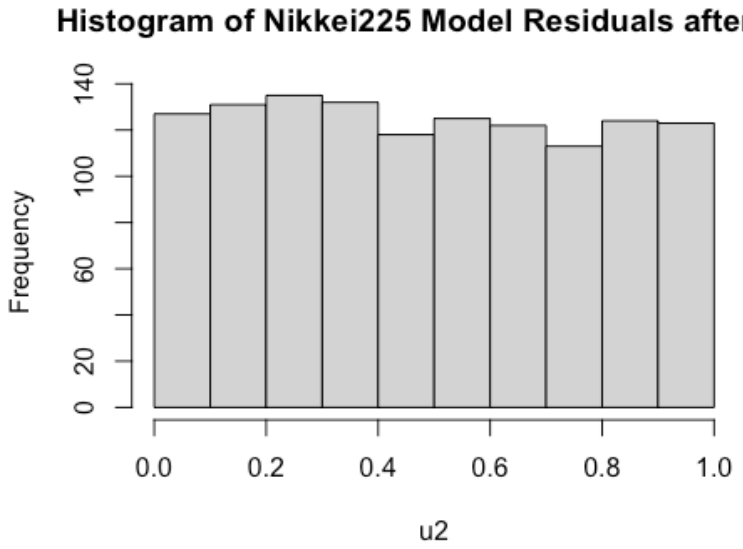

```
# Kolmogorov-Smirnov Test
KStest1<-LcKS(u1, cdf = "punif")
KStest1$p.value # Pass
```

```
## [1] 0.4062
```

```
# Anderson-Darling test
ADtest1<-ad.test(u1, null="punif")
ADtest1$p.value # Pass
```

```
## [1] 0.2151517
```

```
# Nikkei 225
# Check histogram
u2<-psstd(res2,mean=0, sd=1,nu=coef(model_Nk225)["shape"], xi=coef(mode
l_Nk225)["skew"])
hist(u2,  main = "Histogram of Nikkei225 Model Residuals after PIT") #
Fairly uniform.
```




```
# Kolmogorov-Smirnov Test
KStest2<-LcKS(u2, cdf = "punif")
KStest2$p.value # Pass

## [1] 0.4934

# Anderson-Darling test
ADtest2<-ad.test(u2, null="punif")
ADtest2$p.value # Pass

## [1] 0.4147453
```

This is a summary table for goodness of fit tests above.

```
##                KS        AD
## CAC40     0.4062 0.2151517
## Nikkei225 0.4934 0.4147453
```

We pass the test for uniformity for both transformed log-returns, so we can proceed to copula modelling.

3.2 Fit Copula

```
# After removing N/A values, we could find a difference in observations
 of 2 between
# two data. After conducting standard skew student-t distributions, the
re were
```

```
# still one difference in observation numbers between CAC40 and Neikkei
225.
# Therefore, we have to remove one observation from CAC40 in order to m
atch lengths
# of u1 and u2.
length(u1)

## [1] 1249

length(u2)

## [1] 1250

u2 <-u2[-1]
```

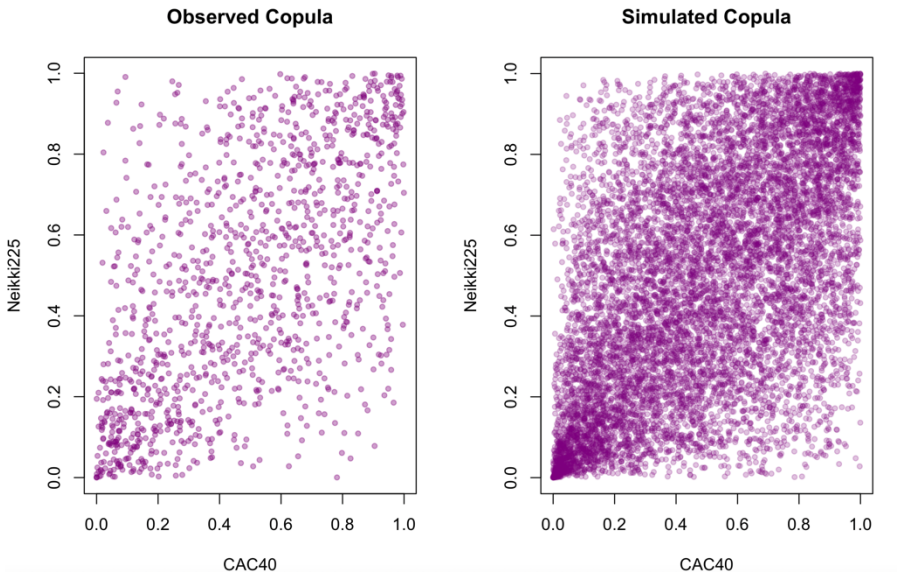
3.2.1 Determine Copula

```
# Using BiCopSelect function to fit various copulas to the dataset and
select
# the copula that provides the best fit based on the AIC criterion.
model=BiCopSelect(u1, u2, familyset=NA, selectioncrit="AIC", indeptest=
TRUE, level=0.05)
model

## Bivariate copula: Survival BB1 (par = 0.13, par2 = 1.54, tau = 0.39)
```

This is a survival BB1 copula.

3.2.2 Monte Carlo simulation

```
N=10000
set.seed(0123)
u_sim=BiCopSim(N, family= model$family, model$par,  model$par2)

par(mfrow=c(1,2))
plot(u1, u2 , xlab="CAC40",ylab="Neikki225", pch=20,
     col = rgb(128/255, 0/255, 128/255, 0.4)
     ,main="Observed Copula")
plot(u_sim[,1],u_sim[,2], xlab="CAC40",ylab="Neikki225", pch=20,
     col = rgb(128/255, 0/255, 128/255, 0.25)
     ,main="Simulated Copula")
```



The observed copula and the simulated copula demonstrate similar trends in their scatterplots. This means that our simulated copula is of good fit. There is a lower tail at the bottom left, and an upper tail tendency on the top right.

3.3 Invserse probability integral transformation (IPIT)

```
rt_CAC40_sim<-qsstd(u_sim[,1], mean=0, sd=1, nu=coef(model_CAC40)["shape"], xi=coef(model_CAC40)["skew"])
rt_Nk225_sim<-qsstd(u_sim[,2], mean=0, sd=1,nu=coef(model_Nk225)["shape"], xi=coef(model_Nk225)["skew"])
```

4. Reintroducing AR-GARCH effects

Re-introduce AR-GARCH effect for CAC40 and GARCH effect for Nikkei 225.

Our codes are based on formulae below:

$$AR(1) + GARCH(1,1):$$

$$\sigma_t^2 = \omega + \alpha \times u_{t-1}^2 + \beta \times \sigma_{t-1}^2$$

$$y_t = \mu + \alpha_1 \times y_{t-1} + \sigma_t \times \varepsilon_t$$

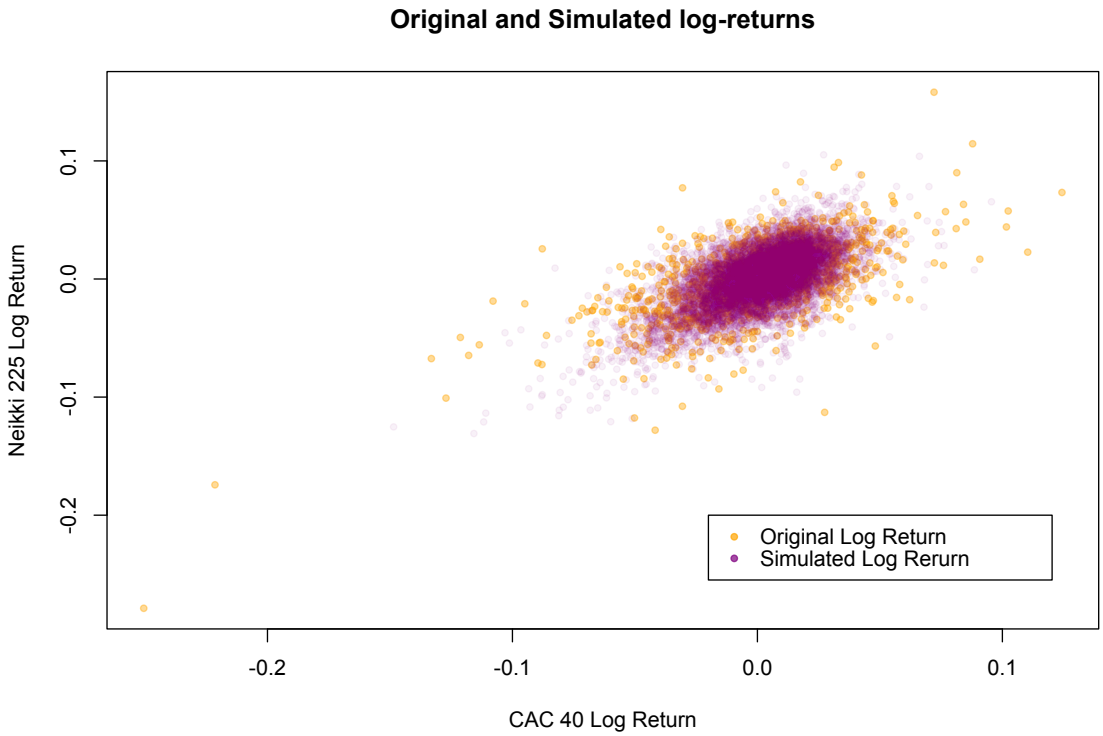
$$GARCH(1,1):$$

$$\sigma_t^2 = \omega + \alpha \times u_{t-1}^2 + \beta \times \sigma_{t-1}^2$$

$$y_t = \mu + \sigma_t \times \varepsilon_t$$

```
# Re-introduce AR-GARCH for CAC40
alpha1<-coef(model_CAC40)["ar1"]
mu<-coef(model_CAC40)["mu"]
alpha<-coef(model_CAC40)["alpha1"]
```

```
beta<-coef(model_CAC40)["beta1"]
omega <- coef(model_CAC40)["omega"]
res <-tail(model_CAC40@residuals,1)
epsilon <- rt_CAC40_sim
sigma<-tail(model_CAC40@sigma.t,1)

cond_var<- omega + alpha * res^2 + beta * sigma^2 # Conditional varianc
e at time t.
CAC40_last<- mu + alpha1*tail(rt_CAC40,1) + sqrt(cond_var)*epsilon

rm(alpha1,mu,alpha,beta,omega,res,epsilon,sigma,cond_var) # Remove repe
ated symbols

# Re-introduce GARCH for Nikkei 225
mu<-coef(model_Nk225)["mu"]
alpha<-coef(model_Nk225)["alpha1"]
beta<-coef(model_Nk225)["beta1"]
omega <- coef(model_Nk225)["omega"]
res<-tail(model_Nk225@residuals,1)
epsilon <- rt_Nk225_sim
sigma<-tail(model_Nk225@sigma.t,1)

cond_var<- omega + alpha * res^2 + beta * sigma^2
Nk225_last<- mu + sqrt(cond_var)*epsilon

rm(mu,alpha,beta,omega,res,epsilon,sigma,cond_var) # Remove repeated sy
mbols

plot(rt_CAC40, rt_Nk225, ylab="Neikki 225 Log Return",xlab="CAC 40 Log
Return",
     main="Original log-returns", pch=20,col = rgb(1, 165/255,0, 0.3))
points(CAC40_last, Nk225_last, pch = 20, col = rgb(128/225, 0, 128/255,
 0.07))
legend(-0.02,-0.2, pch = 20, col = rgb(c(1, 128/255), c(165/255,0), c(0
/255,128/255), 0.7),
       legend = c("Original Log Return","Simulated Log Rerurn"), cex =
1)
```



In this step, we used the formula for conditional variance in a GARCH model and the for log-returns in the AR model and the GARCH model to reintroduce autocorrelations for CAC 40 and Nikkei 225. The plot includes original log-returns and simulated log-returns, and represents a large extent of overlap. Hence, our models perform well.

5. Computing Vale-at-Risk

5.1 Compute portfolio log-returns

Here we construct an equally weighted potfolio for two indices.

```
port_sim <- matrix(0, nro= N, ncol = 1)
var_sim <- matrix(0, nrow = 1, ncol = 2)
port_sim=log(1+((exp(CAC40_last)-1)+(exp(Nk225_last)-1))*(1/2))
```

5.2 Compute Value-at-Risks

```
var_sim=quantile(port_sim,c(0.01,0.05))
var_sim
```

```
##          1%          5%
## -0.05414432 -0.03278675
```

Equally Weighted Portfolio VaR	
99% Value-at-Risk	**95% Value-at-Risk**
5.41%	3.28%

We could conclude from value-at-risks calculations that, there is 95% of confidence that this portfolio will not have a lager loss than 3.28% in one week, and a 99 percent of confidence that this portfolio will not have a larger loss than 5.41% in one week.

- **TASK 2**

In Task 2, we used the adjusted weekly stock prices of CAC 40 and Nikkei 225 same as Task 1. The following discussion will compare the logic and R codes from ourselves and ChatGPT, and identify potential strengths and limitations of ChatGPT for completing Task 1. Note that ChatGPT 4.0 was applied in this task. Our report for Task 2 consists of parts below:

1. Data Preparation

2. Model Selection

3. Constructing Copula and Transformation

4. Risk Analysis and VaR Estimation

5. Confirm the effectiveness

6. Effectiveness of ChatGPT

1. Data Preparation

1.1 Relevant R Packages

1.1.1 Here is a table of R packages used in Task 1 and in Task 2 respectively.

Task 1	Task 2
VineCopula	*copula*
goftest	*zoo*
KScorrect	*tseries*
fGarch	*rugarch*
fBasic	

1.2 Read data for Paris(CAC40) and Japan(Nikkei225) in csv

```
combined_data <- read.csv("CAC40_Nk225.csv")
```

In this process, we provided separate raw data sets of stock prices for ChatGPT. The initial code from ChatGPT cannot efficiently handle common data issues such as removing NA values. However, we combined two groups of data together later in this task and let ChatGPT to correct the code. Note that the combined dataset has been

removed the N/A values artificially. Therefore, ChatGPT cannot present its efficiency of removing missing values.

1.2.1 Compute Log-Return prices for CAC40(Paris) and Nikkei225(Japan)

```
CAC40_returns <- diff(log(combined_data$CAC40))
NIKKEI_returns <- diff(log(combined_data$Nikkei225))
```

However, this step doesn't provide an overview of data and therefore lacks the depth to identify some specific abnormality and independency of datasets.

2. Model Selection

In this step, ChatGPT skips data modelling and proceeds with copula modelling directly. If we correct this issue and inform ChatGPT to simulate data from a GARCH model. It will assume a standard GARCH(1,1) model without any model identification tests. To be more specific, ChatGPT simply conditions on variances of data and ignores the use of conditional mean. Whether to model conditional mean should be taken into account in previous step (an overview of data), and ChatGPT apparently misses this step. The dicision can significantly impact the model's efficacy in reflecting market behavior, particularly in terms of capturing volatility clustering and fat-tail phenomena. Such meticulous parameter adjustment is indispensable for the model to accurately mirror real-market dynamics and to respond aptly to extreme market events. Besides, although GARCH(1,1) is considered as the most used model for time-varying conditional variance, a test for indicators such as AIC and BIC will give more accuracy of model selection.

In addition, ChatGPT assumes skew student-t distribution for the model. This is the same selection of distribution as our Task 1, however, our selection is based on computing AIC values of models.

While it can suggest and implement a range of models, the selection is more formulaic and may fail to capture the details that we would carefully consider.

2.1 Fit a GARCH(1,1) model for CAC40 & Nikkei225

```
# Fit GARCH(1,1) models to each index's returns
spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder =
 c(1,1)),
                   mean.model = list(armaOrder = c(0,0), include.mean =
 TRUE),
                   distribution.model = "sstd")
garch_CAC40 <- ugarchfit(spec, CAC40_returns)
garch_nikkei <- ugarchfit(spec, NIKKEI_returns)
```

2.2 Residuals Processing

Considering the residuals of models, ChatGPT doesn't carry out any diagnostic tests for the fit of models. On the other hand, we draw autocorrelation function and partial

autocorrelation function plots for residuals, and check p-values of Ljung-Box tests to test model fits.

```
# Extract standardized residuals and conditional volatilities
std_resid_CAC40 <- residuals(garch_CAC40, standardize = TRUE)
std_resid_N225 <- residuals(garch_nikkei, standardize = TRUE)
```

3. Constructing Copula and Transformation

The following content has a large extent of difference between our logic in Task 1 and the logic from ChatGPT in terms of copula construction and the Monte Carlo simulation.

First of all, ChatGPT simulates future standardized residuals from the fitted GARCH models, using parameter sigma (from simulating univariate GARCH model) multiply by simulated normal random deviates.

3.1 Simulate future returns and standardized residuals

```
# Simulate future returns based on the GARCH model
n_sim <- 10000
sim_CAC40 <- ugarchsim(garch_CAC40, n.sim = n_sim, m.sim = 1)
sim_nikkei <- ugarchsim(garch_nikkei, n.sim = n_sim, m.sim = 1)

# Simulate future standardized residuals from the fitted GARCH models
sim_CAC40_resid <- as.vector(sigma(sim_CAC40) * rnorm(n_sim))
sim_nikkei_resid <- as.vector(sigma(sim_nikkei) * rnorm(n_sim))
```

3.2 Use the empirical CDF to transform sim_residuals

Secondly, it transformed the simulated residuals by the empirical cumulative distribution function.

```
# Combine the residuals and transform them using empirical CDF (pobs)
u_CAC40 <- pobs(sim_CAC40_resid)
u_nikkei <- pobs(sim_nikkei_resid)
u_combined <- cbind(u_CAC40, u_nikkei)
```

3.3 Fit copula

Thirdly, it creates a normal copula to the transformed residuals using function normalCopula() from package copula and fits the copula to simulated residuals by function fitCopula() with method = “ml”. However, the help page indicates that, for “ml”, data are assumed to be observations from the true underlying copula whose parameter is to be estimated. In task 1, we found that the rotated BB1 copula is the best copula model.

The third step may lead to a different result due to the different residual transformation method and a lack of precision wrong copula model selection.

The code from ChatGPT doesn't have the reintroducing GARCH correlation step either as a consequence of different transformation approach.

```
# Fit a copula to the transformed residuals
cop <- normalCopula(dim = 2)
fit_copula <- fitCopula(cop, u_combined, method = "ml")
```

3.4 Generate samples for portfolio

The 'copula_samples' simulates the joint behaviour of the standardized residuals derived from the GARCH models of log returns. This simulates potfolio returns based on the joint behaviour instead of generating simulated log-returns to compute portfolio VaR like what we did in Task 1.

```
# Generate samples from the copula
copula_samples <- rCopula(n_sim, copula = fit_copula@copula)
```

4. Risk Analysis and VaR Estimation

In the final step, ChatGPT uses generated copula samples to build an equally weighted portfolio while we use GARCH(1,1) and AR(1,1)+GARCH(1,1) formulae including inverse probability integral transformed simulated residuals in Task 1 to build portfolios. Regarding computing Vale-at-Risks, although ChatGPT's calculation bases on losses, its method is correct.

4. 1 Compute portfolio

```
# Assuming equal weights for CAC 40 and Nikkei 225 in the portfolio
portfolio_returns_sim <- rowMeans(copula_samples) # Simplified portfoli
o returns from copula samples

# Conceptual adjustment for calculating losses
portfolio_losses = 1 - (1 + portfolio_returns_sim) # Reflecting losses
from an initial investment of 1
```

4.2 VaR Estimation

```
# Calculate Value-at-Risk (VaR) based on losses
VaR_99 <- quantile(portfolio_losses, 0.99) # Calculating as negative lo
sses
```

```
VaR_95 <- quantile(portfolio_losses, 0.95) # Calculating as negative lo
sses

# Print VaR results
print(paste("99% VaR:", VaR_99))

## [1] "99% VaR: -0.0755063582786452"

print(paste("95% VaR:", VaR_95))

## [1] "95% VaR: -0.159726939074298"
```

4.2.1 The 95% and 99% VaR are shown below together with the values obtained in Task 1.

Value at Risk			
Task 1 generated from our own code		**Task 2 generated from ChatGPT code**	
95%	99%	95%	99%
3.28%	5.41%	15.90%	7.36%

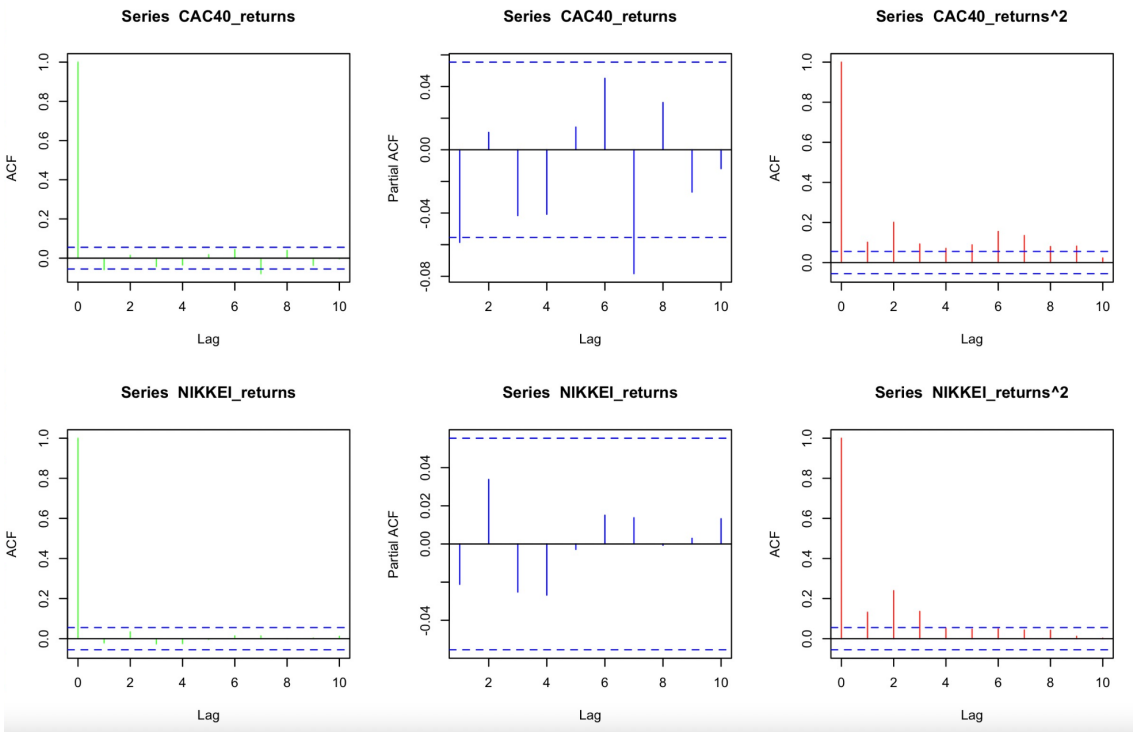
Note that Value at Risk represents the expected maximum loss over a time horizon within a confidence interval such as 95% and 99%. For either a normal distributed or a skew student-t distributed portfolio returns, as the confidence interval increases, the more losses will accumulate, the higher value at risk values will have. The result generated from ChatGPT represents a higher VaR value of 95% confidence interval than the value of 99% confidence interval. This is impossible in the real world and therefore, ChatGPT doesn't complete this task correctly.

5. Confirm the effectiveness

In order to verify the effectiveness of codes generated from ChatGPT, we have the following codes from us to illustrate this with some key plots.

5.1 Draw time series plots

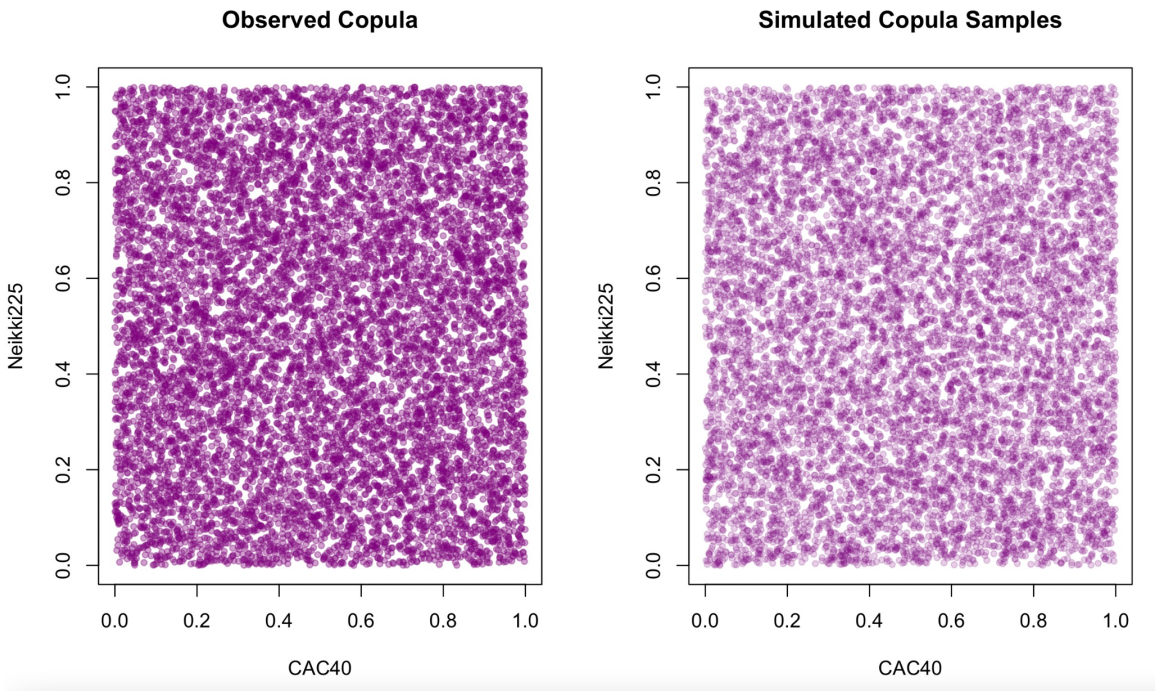
```
par(mfrow=c(2,3))
acf(CAC40_returns,lag.max= 10, col="green")
pacf(CAC40_returns,lag.max= 10, col="blue")
acf(CAC40_returns^2,lag.max= 10, col="red")
acf(NIKKEI_returns,lag.max= 10, col="green")
pacf(NIKKEI_returns,lag.max= 10, col="blue")
acf(NIKKEI_returns^2,lag.max= 10, col="red")

par(mfrow=c(1,1))
```



The second plot at the top suggests an AR(1) model for the conditional mean of CAC 40 log-returns, while this has not been taken into account by ChatGPT.

5.2 Draw copula comparation plots

```
par(mfrow=c(1,2))
plot(u_CAC40, u_nikkei , xlab="CAC40",ylab="Neikki225", pch=20,
     col = rgb(128/255, 0/255, 128/255, 0.4),main="Observed Copula")
plot(copula_samples[,1],copula_samples[,2], xlab="CAC40",ylab="Neikki22
5", pch=20,
     col = rgb(128/255, 0/255, 128/255, 0.2),main="Simulated Copula Sam
ples")
```




```
par(mfrow=c(1,1))
```

There is no tendency in both scatterplots. As a result, we can conclude that the copula generated from ChatGPT is not fitted although points lie within the [0,1] interval.

6. Effectiveness of ChatGPT

6.1 Strengths:

ChatGPT offers rapid setup and can guide less experienced users through standard data preparation steps.

ChatGPT can deal with various functions such as ugarchspec(), and provide approaches that we haven't considered before such as the empirical cumulative distribution function. In this way, users will have more choices to solve the problem.

ChatGPT used GARCH(1,1) model to construct models for data. This is the commonly used model for data with volatile variances with time such as stock prices.

ChatGPT can provide a baseline for further analysis and save time.

ChatGPT simplifies the introduction to risk analysis concepts and can quickly generate code for basic VaR calculations.

6.2 Limitations:

Subsequent attempts to execute code generated by ChatGPT revealed issues in the data preparation phase such as the AI's handling of missing data. Despite efforts to correct this through feedback, the resulting data management was suboptimal, necessitating manual intervention and corrections by our team.

The provided codes are frequently inaccurate and error-ridden. It is often necessary to execute multiple runs in R Studio personally to identify these errors and send these issues back to ChatGPT for correction.

ChatGPT lacks the deep understanding of specific data, which may lead to potentially missing critical preprocessing steps required for optimal model performance. For example, it doesn't test for independency and normality of data. As a consequence, ChatGPT doesn't include the AR model for conditional mean of CAC 40. This leads to a less precision in data modelling.

ChatGPT fails to apply incorporate essential methodology such as the statistical model for time series data unless we correct this artificially.

ChatGPT doesn't offer the detailed evaluation needed to refine models and check residuals, potentially overlooking subtleties that affect model validity or interpretation.

ChatGPT lacks the depth to handle complex risk modeling problems, which requires artificial intervention to ensure accuracy and relevance.

The code generated from ChatGPT does not include creating plots for visual illustration.

The culmination of these issues was the generation of a positive VaR output at the first attempt, contradicting the established understanding that VaR, in the context of task1,

should manifest as a negative value. This discrepancy prompted a re-evaluation of the analytical steps undertaken by ChatGPT.

ChatGPT cannot recognize the mistake of conducting an incorrect sign of VaR. If you point out this issue, it may initially attempt to resolve it by merely altering the value of VAR to positive. Instead, it is more effective to insist that ChatGPT revisit and amend the coding approach rather than simply adjusting the variable's sign.

6.3 Conclusion:

Overall, ChatGPT 4.0 can respond to data and tagets of Task 1 rapidly, while the codes generated by it have revealed numerous problems and the inappropriate models and defective steps will lead to an incorrect result of Value at Risk.

In our task 2 report, we explore the process and outcomes of utilizing ChatGPT 4.0 to tackle Task 1, where ChatGPT offers a comprehensive framework as guidance. However, this framework is flawed, particularly evident in the decision to bypass the GARCH model treatment of time series data directly in favor of utilizing the copula model. This oversight, coupled with the absence of fitness tests at each step, potentially introduces significant errors. Moreover, the lack of application of the probabilistic integral transform (PIT) within the copula theory and the failure to reintroduce GARCH/AR effects further contribute to inaccuracies. The methodologies employed in Task 1, such as the Monte Carlo simulation approach and the method of transforming residuals using the empirical cumulative distribution function (ECDF), diverge from those suggested by ChatGPT, leading to erroneously calculated Value at Risk (VaR) values.

Upon comparing the results derived from manual methods, it becomes evident that while ChatGPT exhibits proficiency in data preparation, model recommendation, and simulation tasks, its effectiveness is markedly limited when faced with the unique challenges of financial data and the intricacies of model selection and adjustment. This underscores the crucial role of human oversight in the selection of financial models, interpretation of results, and the making of strategic decisions.

In summary, using ChatGPT does offer convenience when completing complex financial analysis tasks, particularly in providing a structure or framework for the project and guidance in completing specific detailed steps. ChatGPT's ability to produce valuable content demonstrates its strengths when given detailed guidance rather than being expected to complete the entire project in one go. However, for key aspects such as model selection, parameter tuning and in-depth analysis, human expertise is also required to ensure the accuracy and usefulness of the results. Therefore, in real-world financial analysis, ChatGPT is best used as an auxiliary tool to assist analysts in conducting more in-depth and accurate analyses, and should not be fully relied upon for decision-making.