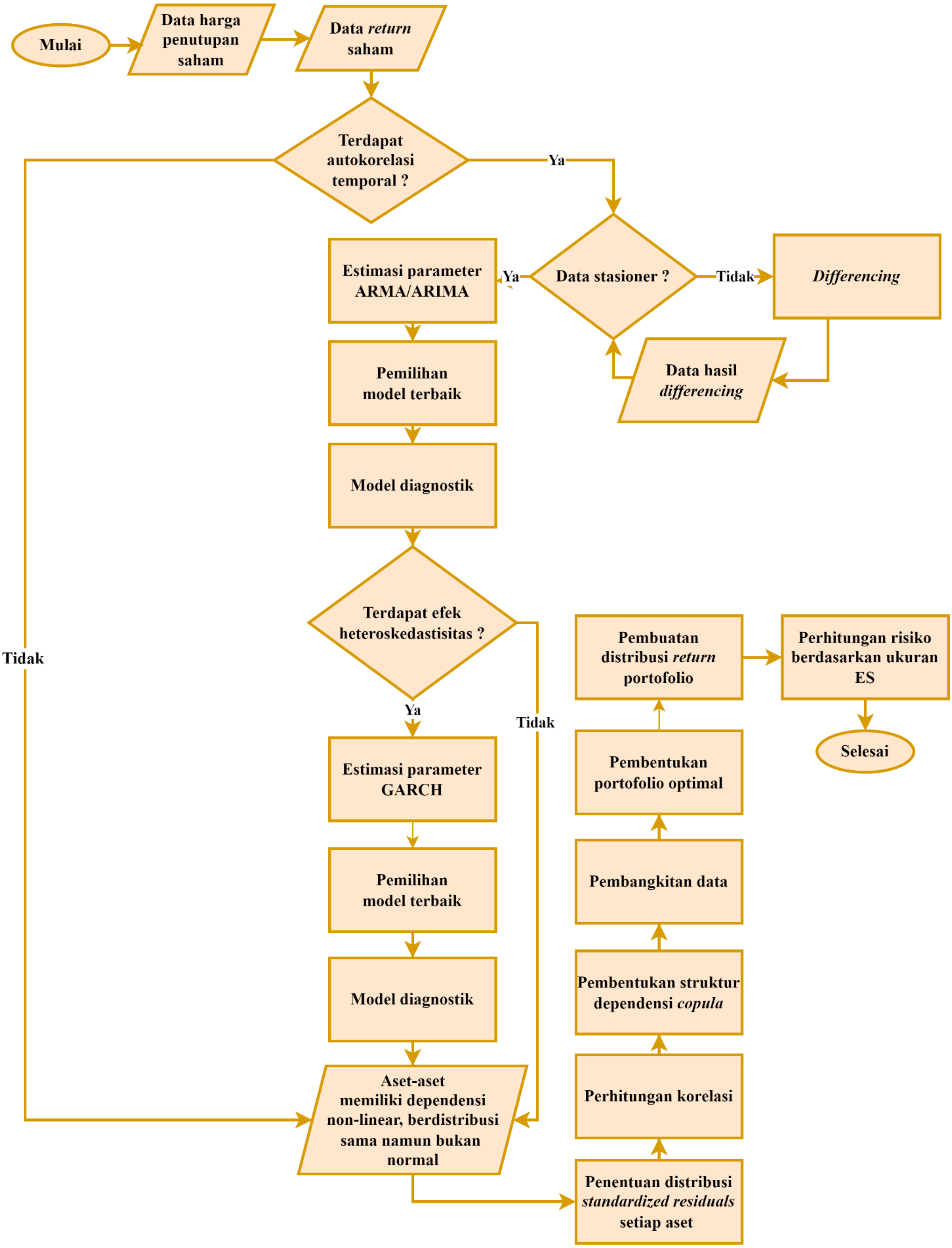
PENGUKURAN EXPECTED SHORTFALL PADA PORTOFOLIO OPTIMAL BERDASARKAN COPULA-ARMA-GARCH


Mulai
Data harga penutupan saham
Data *return* saham
Terdapat autokorelasi temporal ?
Ya
Tidak
Data stasioner ?
Tidak
Differencing
Data hasil *differencing*
Ya
Estimasi parameter ARMA/ARIMA
Pemilihan model terbaik
Model diagnostik
Terdapat efek heteroskedastisitas ?
Ya
Tidak
Estimasi parameter GARCH
Pemilihan model terbaik
Model diagnostik
Aset-aset memiliki dependensi non-linear, berdistribusi sama namun bukan normal
Penentuan distribusi *standardized residuals* setiap aset
Perhitungan korelasi
Pembentukan struktur dependensi *copula*
Pembangkitan data
Pembentukan portofolio optimal
Pembuatan distribusi *return* portofolio
Perhitungan risiko berdasarkan ukuran ES
Selesai

MENYIAPKAN LIBRARY YANG DIPERLUKAN

```
# Library #
library(quantmod)
library(lattice)
library(timeSeries)
library(rugarch)
library(tseries)
library(aTSA)
require(timeDate)
require(fBasics)
library(MASS)
library(rmutil)
library(goftest)
library(zoo)
library(copula)
library(mnormt)
library(fGarch)
library(QRM)
library(TSA)
library(forecast)


library("goft")
library("readr")
library("tsoutliers")
library("forecast")
library("tseries")
library("lmtest")
library("FitAR")
library("HAC")
library(readxl)
```

DATA PREPARATION

```
# Input Data Return #
setwd("D:/OneDrive - sworepublic/Local Disk D (Lenovo)/Data")
PANR = read_excel("Data Return.xlsx",sheet = 1)
AALI = read_excel("Data Return.xlsx",sheet = 4)
ICBP = read_excel("Data Return.xlsx",sheet = 3)

# Hanya mengambil kolom Return #
returnPANR = PANR[,2]
returnAALI = AALI[,2]
returnICBP = ICBP[,2]

# Konversi data sebagai numeric agar bisa diolah #
returnPANR = as.numeric(unlist(returnPANR))
returnAALI = as.numeric(unlist(returnAALI))
returnICBP = as.numeric(unlist(returnICBP))
```

DESKRIPSI DATA *RETURN* MASING-MASING ASET

```
# Ringkasan Data #
summary(returnPANR)
sd(returnPANR)
var(returnPANR)
distFreq_PANR = data.frame(table(returnPANR))
(modus_PANR = distFreq_PANR[order(distFreq_PANR$Freq, decreasing = TRUE),][1,])
skewness(returnPANR)
kurtosis(returnPANR)

summary(returnAALI)
sd(returnAALI)
var(returnAALI)
```

```
distFreq_AALI = data.frame(table(returnAALI))
(modus_AALI = distFreq_AALI[order(distFreq_AALI$Freq, decreasing = TRUE),][1,])
skewness(returnAALI)
kurtosis(returnAALI)

summary(returnICBP)
sd(returnICBP)
var(returnICBP)
distFreq_ICBP = data.frame(table(returnICBP))
(modus_ICBP = distFreq_ICBP[order(distFreq_ICBP$Freq, decreasing = TRUE),][1,])
skewness(returnICBP)
kurtosis(returnICBP)

# Histogram #
hist(x=returnPANR, main = "Probability Density Function" , xlab = "Return PANR", prob =
TRUE, breaks=10, xlim = c(-0.15, 0.4), ylim = c(0,35))
lines(density(returnPANR), col = 4, adjust=10)
range(returnPANR)

hist(returnAALI)
lines(density(returnAALI), col = 4, lwd = 5)
range(returnAALI)

hist(returnICBP)
lines(density(returnICBP), col = 4, lwd = 5)
range(returnICBP)

par(mfrow=c(1,3))
hist(returnPANR, freq = FALSE)
hist(returnAALI, freq = FALSE)
hist(returnICBP, freq = FALSE)

# Scatterplot #
par(mfrow=c(1,1))
plot(returnPANR, returnICBP)
plot(returnAALI, returnICBP)
plot(returnPANR, returnAALI)
```

PEMODELAN ARIMA *RETURN* ASET

```
returnICBP = ts(returnICBP)
returnPANR = ts(returnPANR)
returnAALI = ts(returnAALI)

par(mfrow=c(3,1))
par(mfrow=c(1,1))
plot(returnPANR, ylab = "Return PANR", main="Plot Deret Waktu PANR.JK")
plot(returnAALI, ylab = "Return AALI",main="Plot Deret Waktu AALI.JK")
plot(returnICBP, ylab = "Return ICBP",main="Plot Deret Waktu ICBP.JK")
```

A. PANR

1. Uji Autokorelasi Temporal

```
Box.test(returnPANR,lag = 21, type=c("Ljung-Box"))
```

2. Uji Stasioneritas Data

```
adf.test(returnPANR) #terbukti stasioner dalam rata2
```

3. Identifikasi Model Menggunakan Plot ACF dan PACF

```
par(mfrow=c(2,1))
acf(returnPANR,16,main="ACF Return PANR.JK")
pacf(returnPANR,16,main="PACF Return PANR.JK") # (2,2)
```

4. Pembentukan Model ARIMA

```
ARMA22PANR = arima(returnPANR,c(2,0,2))
ARMA21PANR = arima(returnPANR,c(2,0,1))
AR2PANR = arima(returnPANR,c(2,0,0))
ARMA12PANR = arima(returnPANR,c(1,0,2))
ARMA11PANR = arima(returnPANR,c(1,0,1))
AR1PANR = arima(returnPANR,c(1,0,0))
MA2PANR = arima(returnPANR,c(0,0,2))
MA1PANR = arima(returnPANR,c(0,0,1))
```

5. Uji Signifikansi Parameter

```
##UJI SIGNIFIKANSI PARAMETER
printstatarima=function(returnPANR,digits=4,se=T,...){
  if (length(returnPANR$coef)>0){
    cat("\nCoefficients:\n")
    coef=round(returnPANR$coef,digits=digits)
    if(se && nrow(returnPANR$var.coef)){
      ses=rep(0, length(coef))
      ses[returnPANR$mask] = round(sqrt(diag(returnPANR$var.coef)),digits=digits)
      coef=matrix(coef,1,dimnames=list(NULL, names(coef)))
      coef=rbind(coef,s.e.=ses)
      statt=coef[1,]/ses
      pval=2*pt(abs(statt),df=length(returnPANR$residuals)-1,lower.tail=F)
      coef=rbind(coef,t=round(statt,digits=digits),sign.=round(pval,digits=digits))
      coef=t(coef)
    }
    print.default(coef,print.gap=2)
  }
}

printstatarima(ARMA22PANR)#
printstatarima(ARMA21PANR)
printstatarima(AR2PANR)#
printstatarima(ARMA12PANR)
printstatarima(ARMA11PANR)#
printstatarima(AR1PANR)#
printstatarima(MA2PANR)#
printstatarima(MA1PANR)#
```

6. Model Terbaik

```
aic = as.matrix(AIC(ARMA22PANR, AR2PANR, ARMA11PANR, AR1PANR, MA2PANR, MA1PANR))
aic_best = aic[order(aic[,2]),] #ARMA22PANR
```

B. AALI

1. Uji autokorelasi temporal

```
Box.test(returnAALI,lag = 21, type=c("Ljung-Box"))
```

2. Uji Stasioneritas Data

```
adf.test(returnAALI)#terbukti stasioner dalam rata2
```

3. Identifikasi Model Menggunakan Plot ACF dan PACF

```
par(mfrow=c(2,1))
acf(returnAALI,16,main="ACF Return AALI.JK")
pacf(returnAALI,16,main="PACF Return AALI.JK")#(2,2)
```

4. Pembentukan Model ARIMA

```
ARMA33AALI = arima(returnAALI, c(3,0,3))
ARMA32AALI = arima(returnAALI, c(3,0,2))
ARMA31AALI = arima(returnAALI, c(3,0,1))
AR3AALI = arima(returnAALI, c(3,0,0))
ARMA23AALI = arima(returnAALI, c(2,0,3))
ARMA22AALI = arima(returnAALI,c(2,0,2))
ARMA21AALI = arima(returnAALI,c(2,0,1))
AR2AALI = arima(returnAALI,c(2,0,0))
ARMA13AALI = arima(returnAALI,c(1,0,3))
ARMA12AALI = arima(returnAALI,c(1,0,2))
ARMA11AALI = arima(returnAALI,c(1,0,1))
AR1AALI = arima(returnAALI,c(1,0,0))
MA3AALI = arima(returnAALI,c(0,0,3))
MA2AALI = arima(returnAALI,c(0,0,2))
MA1AALI = arima(returnAALI,c(0,0,1))
```

5. Signifikansi Parameter

```
printstatarima=function(returnAALI,digits=4,se=T,...){
  if (length(returnAALI$coef)>0){
    cat("\nCoefficients:\n")
    coef=round(returnAALI$coef,digits=digits)
    if(se && nrow(returnAALI$var.coef)){
      ses=rep(0, length(coef))
      ses[returnAALI$mask] = round(sqrt(diag(returnAALI$var.coef)),digits=digits)
      coef=matrix(coef,1,dimnames=list(NULL, names(coef)))
      coef=rbind(coef,s.e.=ses)
      statt=coef[1,]/ses
      pval=2*pt(abs(statt),df=length(returnAALI$residuals)-1,lower.tail=F)
      coef=rbind(coef,t=round(statt,digits=digits),sign.=round(pval,digits=digits))
      coef=t(coef)
    }
    print.default(coef,print.gap=2)
  }
}

printstatarima(ARMA33AALI)
printstatarima(ARMA32AALI)
printstatarima(ARMA31AALI)
printstatarima(AR3AALI)
printstatarima(ARMA23AALI)
printstatarima(ARMA22AALI)
printstatarima(ARMA21AALI)
printstatarima(AR2AALI)
printstatarima(ARMA13AALI)#
printstatarima(ARMA12AALI)
printstatarima(ARMA11AALI)#
printstatarima(AR1AALI)#
printstatarima(MA3AALI)
printstatarima(MA2AALI)
printstatarima(MA1AALI)#
```

6. Model Terbaik

```
AIC(ARMA13AALI, ARMA11AALI, AR1AALI,MA1AALI) #ARMA13AALI AIC =-4555.326
```

C. AALI

1. Uji Autokorelasi Temporal

```
Box.test(returnICBP,lag = 21, type=c("Ljung-Box"))
```

2. Uji Stasioneritas Data

```
adf.test(returnICBP)#terbukti stasioner dalam rata2
```

3. Identifikasi Model Menggunakan Plot ACF dan PACF

```
par(mfrow=c(2,1))
acf(returnICBP,16,main="ACF Return ICBP.JK")
pacf(returnICBP,16,main="PACF Return ICBP.JK") # (2,2)
```

4. Pembentukan Model ARIMA

```
ARMA22ICBP = arima(returnICBP,c(2,0,2))
ARMA21ICBP = arima(returnICBP,c(2,0,1))
AR2ICBP = arima(returnICBP,c(2,0,0))
ARMA12ICBP = arima(returnICBP,c(1,0,2))
ARMA11ICBP = arima(returnICBP,c(1,0,1))
AR1ICBP = arima(returnICBP,c(1,0,0))
MA2ICBP = arima(returnICBP,c(0,0,2))
MA1ICBP = arima(returnICBP,c(0,0,1))
```

5. Signifikansi Parameter

```
printstatarima=function(returnICBP,digits=4,se=T,...){
  if (length(returnICBP$coef)>0){
    cat("\nCoefficients:\n")
    coef=round(returnICBP$coef,digits=digits)
    if(se && nrow(returnICBP$var.coef)){
      ses=rep(0, length(coef))
      ses[returnICBP$mask] = round(sqrt(diag(returnICBP$var.coef)),digits=digits)
      coef=matrix(coef,1,dimnames=list(NULL, names(coef)))
      coef=rbind(coef,s.e.=ses)
      statt=coef[1,]/ses
      pval=2*pt(abs(statt),df=length(returnICBP$residuals)-1,lower.tail=F)
      coef=rbind(coef,t=round(statt,digits=digits),sign.=round(pval,digits=digits))
      coef=t(coef)
    }
    print.default(coef,print.gap=2)
  }
}

printstatarima(ARMA22ICBP)
printstatarima(ARMA21ICBP)
printstatarima(AR2ICBP)
printstatarima(ARMA12ICBP)
printstatarima(ARMA11ICBP)#
printstatarima(AR1ICBP)
printstatarima(MA2ICBP)
printstatarima(MA1ICBP)
```

6. Model Terbaik

```
AIC(ARMA11ICBP) #ARMA11ICBP AIC =-5293.611
```

---

7. Uji Asumsi Residual

1) Pengujian Non Autokorelasi

```
resPANR=ARMA22PANR$residuals
wnPANR=Box.test(resPANR, type=c("Ljung-Box"))
wnPANR

resAALI=ARMA13AALI$residuals
wnAALI=Box.test(resAALI, type=c("Ljung-Box"))
wnAALI

resICBP=ARMA11ICBP$residuals
wnICBP=Box.test(resICBP, type=c("Ljung-Box"))
wnICBP

cat('H0: non autokorelasi') # ketiganya sudah non autokorelasi
```

2) Pengujian Normalitas

```
library("nortest")

nPANR=length(resPANR)
sdPANR=sd(resPANR)
meanPANR = mean(returnPANR)
resnPANR=rnorm(nPANR,meanPANR,sdPANR)
ks.test(resPANR,resnPANR)

nAALI=length(resAALI)
sdAALI=sd(resAALI)
meanAALI = mean(returnAALI)
resnAALI=rnorm(nAALI,meanAALI,sdAALI)
ks.test(resAALI,resnAALI)

nICBP=length(resICBP)
sdICBP=sd(resICBP)
meanICBP = mean(returnICBP)
resnICBP=rnorm(nICBP,meanICBP,sdICBP)
ks.test(resICBP,resnICBP)

cat('H0 : residual berdistribusi normal') # Ketiga residual tidak berdistribusi normal
```

3) Pengujian Homoskedastisitas

```
# Pakai Uji L-Jung Box (residual kuadrat)

Box.test(resPANR^2, type=c("Ljung-Box"))
Box.test(resAALI^2, type=c("Ljung-Box"))
Box.test(resICBP^2, type=c("Ljung-Box"))

# Pakai Uji Lagrange Multiplier

arch.test(ARMA22PANR, output =TRUE)
arch.test(ARMA13AALI, output = TRUE)
arch.test(ARMA11ICBP, output = TRUE)

cat('HO: Residual homogen') #Ketiga residual memiliki efek heteroskedastisitas


'''
library(writexl)
df_resPANR = data.frame(resPANR)
cat('write_xlsx(the dataframe name,"path to store the Excel file\\file name.xlsx")
')
write_xlsx(df_resPANR,"D:/OneDrive - sworepublic/Local Disk D (Lenovo)
/Data\\residualPANR.xlsx")

df_resSKLT = data.frame(resSKLT)
write_xlsx(df_resSKLT,"D:/OneDrive - sworepublic/Local Disk D (Lenovo)
/Data\\residualSKLT.xlsx")
```

```
'''
```

PEMODELAN ARCH/GARCH *RETURN* ASET

A. PANR

1. Identifikasi Model Menggunakan Plot ACF dan PACF

```
par(mfrow=c(2,1))
acf(resPANR^2, 16, main = "ACF of Squared ARMA Residuals on PANR.JK")
pacf(resPANR^2, 16, main = "PACF of Squared ARMA Residuals on PANR.JK") # 2,3
```

2. Pembentukan Model ARCH/GARCH

```
library(rugarch)

GARCH1PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(0,1)))
fitGARCH1PANR = ugarchfit(data=returnPANR, spec = GARCH1PANR,solver='hybrid')

GARCH2PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(0,2)))
fitGARCH2PANR = ugarchfit(data=returnPANR, spec = GARCH2PANR,solver='hybrid')

GARCH3PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(0,3)))
fitGARCH3PANR = ugarchfit(data=returnPANR, spec = GARCH3PANR,solver='hybrid')

ARCH1PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,0)))
fitARCH1PANR = ugarchfit(data=returnPANR, spec = ARCH1PANR,solver='hybrid')

ARCHGARCH11PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,1)))
fitARCHGARCH11PANR = ugarchfit(data=returnPANR, spec = ARCHGARCH11PANR,solver='hybrid')

ARCHGARCH12PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,2)))
fitARCHGARCH12PANR = ugarchfit(data=returnPANR, spec = ARCHGARCH12PANR,solver='hybrid')

ARCHGARCH13PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,3)))
fitARCHGARCH13PANR = ugarchfit(data=returnPANR, spec = ARCHGARCH13PANR,solver='hybrid')

ARCH2PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,0)))
fitARCH2PANR = ugarchfit(data=returnPANR, spec = ARCH2PANR,solver='hybrid')

ARCHGARCH21PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,1)))
fitARCHGARCH21PANR = ugarchfit(data=returnPANR, spec = ARCHGARCH21PANR,solver='hybrid')

ARCHGARCH22PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,2)))
fitARCHGARCH22PANR = ugarchfit(data=returnPANR, spec = ARCHGARCH22PANR,solver='hybrid')

ARCHGARCH23PANR = ugarchspec(mean.model = list(armaOrder=c(2,2), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,3)))
fitARCHGARCH23PANR = ugarchfit(data=returnPANR, spec = ARCHGARCH23PANR,solver='hybrid')
```

3. Signifikansi Parameter

```
fitGARCH1PANR # AIC = -3.6209, heteros
fitGARCH2PANR # AIC = -3.6237, heteros
fitGARCH3PANR
```

```
fitARCH1PANR # AIC = -3.8366, heteros
fitARCHGARCH11PANR # AIC =  -4.5218, homo
fitARCHGARCH12PANR # AIC = -4.5219 (terbaik) # homo --> mu tidak signifikan
fitARCHGARCH13PANR
fitARCH2PANR
fitARCHGARCH21PANR
fitARCHGARCH22PANR
fitARCHGARCH23PANR

# Mengambil Parameter yang Signifikan dari Model Terbaik
ar1PANR = fitARCHGARCH12PANR@model$pars[2,1]
ar2PANR =fitARCHGARCH12PANR@model$pars[3,1]
ma1PANR =fitARCHGARCH12PANR@model$pars[4,1]
ma2PANR =fitARCHGARCH12PANR@model$pars[5,1]
omegaPANR =fitARCHGARCH12PANR@model$pars[9,1]
alpha1PANR =fitARCHGARCH12PANR@model$pars[10,1]
beta1PANR =fitARCHGARCH12PANR@model$pars[11,1]
beta2PANR =  fitARCHGARCH12PANR@model$pars[12,1]
```

B. AALI

1. Identifikasi Model Menggunakan Plot ACF dan PACF

```
par(mfrow=c(2,1))
acf(resAALI^2, 16, main = "ACF of Squared ARMA Residuals on AALI.JK")
pacf(resAALI^2, 16, main = "PACF of Squared ARMA Residuals on AALI.JK") #2,3
```

2. Pembentukan Model ARCH/GARCH

```
GARCH1AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(0,1)))
fitGARCH1AALI = ugarchfit(data=returnAALI, spec = GARCH1AALI,solver='hybrid')

GARCH2AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(0,2)))
fitGARCH2AALI = ugarchfit(data=returnAALI, spec = GARCH2AALI,solver='hybrid')

GARCH3AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(0,3)))
fitGARCH3AALI = ugarchfit(data=returnAALI, spec = GARCH3AALI,solver='hybrid')

ARCH1AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,0)))
fitARCH1AALI = ugarchfit(data=returnAALI, spec = ARCH1AALI,solver='hybrid')

ARCHGARCH11AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,1)))
fitARCHGARCH11AALI = ugarchfit(data=returnAALI, spec = ARCHGARCH11AALI,solver='hybrid')

ARCHGARCH12AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,2)))
fitARCHGARCH12AALI = ugarchfit(data=returnAALI, spec = ARCHGARCH12AALI,solver='hybrid')

ARCHGARCH13AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,3)))
fitARCHGARCH13AALI = ugarchfit(data=returnAALI, spec = ARCHGARCH13AALI,solver='hybrid')

ARCH2AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,0)))
fitARCH2AALI = ugarchfit(data=returnAALI, spec = ARCH2AALI,solver='hybrid')

ARCHGARCH21AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,1)))
fitARCHGARCH21AALI = ugarchfit(data=returnAALI, spec = ARCHGARCH21AALI,solver='hybrid')
```

```
ARCHGARCH22AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,3)))
fitARCHGARCH22AALI = ugarchfit(data=returnAALI, spec = ARCHGARCH22AALI,solver='hybrid')

ARCHGARCH23AALI = ugarchspec(mean.model = list(armaOrder=c(1,3), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,3)))
fitARCHGARCH23AALI = ugarchfit(data=returnAALI, spec = ARCHGARCH23AALI,solver='hybrid')
```

3. Signifikansi Parameter

```
fitGARCH1AALI
fitGARCH2AALI # AIC = -4.5341, heteros
fitGARCH3AALI #  AIC =  -4.5322, masih heteros
fitARCH1AALI # AIC =  17.725, homos
fitARCHGARCH11AALI # AIC = -4.7100 (terbaik) # homos --> mu ga signif
fitARCHGARCH12AALI
fitARCHGARCH13AALI
fitARCH2AALI
fitARCHGARCH21AALI
fitARCHGARCH22AALI
fitARCHGARCH23AALI

# Mengambil Parameter yang Signifikan dari Model Terbaik
ar1AALI = fitARCHGARCH11AALI@model$pars[2,1]
ma1AALI = fitARCHGARCH11AALI@model$pars[3,1]
ma2AALI = fitARCHGARCH11AALI@model$pars[4,1]
ma3AALI = fitARCHGARCH11AALI@model$pars[5,1]
omegaAALI= fitARCHGARCH11AALI@model$pars[9,1]
alpha1AALI = fitARCHGARCH11AALI@model$pars[10,1]
beta1AALI = fitARCHGARCH11AALI@model$pars[11,1]
```

C. ICBP

1. Identifikasi Model Menggunakan Plot ACF dan PACF

```
par(mfrow=c(2,1))
acf(resICBP^2, 16, main = "ACF of Squared ARMA Residuals on ICBP.JK")
pacf(resICBP^2, 16, main = "PACF of Squared ARMA Residuals on ICBP.JK") #2,3
```

2. Pembentukan Model ARCH/GARCH

```
GARCH1ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(0,1)))
fitGARCH1ICBP = ugarchfit(data=returnICBP, spec = GARCH1ICBP,solver='hybrid')

GARCH2ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(0,2)))
fitGARCH2ICBP = ugarchfit(data=returnICBP, spec = GARCH2ICBP,solver='hybrid')

GARCH3ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(0,3)))
fitGARCH3ICBP = ugarchfit(data=returnICBP, spec = GARCH3ICBP,solver='hybrid')

ARCH1ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,0)))
fitARCH1ICBP = ugarchfit(data=returnICBP, spec = ARCH1ICBP,solver='hybrid')

ARCHGARCH11ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,1)))
fitARCHGARCH11ICBP = ugarchfit(data=returnICBP, spec = ARCHGARCH11ICBP,solver='hybrid')

ARCHGARCH12ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,2)))
fitARCHGARCH12ICBP = ugarchfit(data=returnICBP, spec = ARCHGARCH12ICBP,solver='hybrid')
```

```
ARCHGARCH13ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(1,3)))
fitARCHGARCH13ICBP = ugarchfit(data=returnICBP, spec = ARCHGARCH13ICBP,solver='hybrid')

ARCH2ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,0)))
fitARCH2ICBP = ugarchfit(data=returnICBP, spec = ARCH2ICBP,solver='hybrid')

ARCHGARCH21ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,1)))
fitARCHGARCH21ICBP = ugarchfit(data=returnICBP, spec = ARCHGARCH21ICBP,solver='hybrid')

ARCHGARCH22ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,2)))
fitARCHGARCH22ICBP = ugarchfit(data=returnICBP, spec = ARCHGARCH22ICBP,solver='hybrid')

ARCHGARCH23ICBP = ugarchspec(mean.model = list(armaOrder=c(1,1), include.mean = T),
variance.model = list(model="sGARCH",garchOrder=c(2,3)))
fitARCHGARCH23ICBP = ugarchfit(data=returnICBP, spec = ARCHGARCH23ICBP,solver='hybrid')
```

3. Signifikansi Parameter

```
fitGARCH1ICBP #AIC =  -5.2707, heteros
fitGARCH2ICBP #AIC =  -5.2686, heteros
fitGARCH3ICBP #AIC =  -5.2666, homos
fitARCH1ICBP # AIC =  -5.3595, heteros
fitARCHGARCH11ICBP # AIC =   -5.4166, homos
fitARCHGARCH12ICBP
fitARCHGARCH13ICBP
fitARCH2ICBP # AIC =  -5.4210 (terbaik)#, homos --> mu ga signif
fitARCHGARCH21ICBP
fitARCHGARCH22ICBP
fitARCHGARCH23ICBP

# Mengambil Parameter yang Signifikan dari Model Terbaik
ar1ICBP = fitARCH2ICBP@model$pars[2,1]
ma1ICBP = fitARCH2ICBP@model$pars[3,1]
omegaICBP = fitARCH2ICBP@model$pars[7,1]
alpha1ICBP = fitARCH2ICBP@model$pars[8,1]
alpha2ICBP = fitARCH2ICBP@model$pars[9,1]
```

MODEL ARMA-ARCH/GARH *RETURN* ASET

```
cat(' Model Terbaik :
      1. PANR : (1,2)
      2. AALI : (1,1)
      3. ICBP : (2,0))
```

1. Mengambil Nilai *Return* dan Sigma Setiap Aset pada Periode Tertentu

```
SeriesFcPANR=(ugarchforecast(fitARCHGARCH12PANR)@forecast$seriesFor)[1]
sigmaFcPANR=(ugarchforecast(fitARCHGARCH12PANR)@forecast$sigmaFor)[1]

SeriesFcAALI=(ugarchforecast(fitARCHGARCH11AALI)@forecast$seriesFor)[1]
sigmaFcAALI=(ugarchforecast(fitARCHGARCH11AALI)@forecast$sigmaFor)[1]

SeriesFcICBP=(ugarchforecast(fitARCH2ICBP)@forecast$seriesFor)[1]
sigmaFcICBP=(ugarchforecast(fitARCH2ICBP)@forecast$sigmaFor)[1]
```

2. Pengujian Normalitas

```
resPANR_2=fitARCHGARCH12PANR@fit$residuals
```

```
resAALI_2=fitARCHGARCH11AALI@fit$residuals
resICBP_2=fitARCH2ICBP@fit$residuals

nPANR_2=length(resPANR_2)
meanPANR_2=mean(resPANR_2)
sdPANR_2=sd(resPANR_2)
resnPANR_2=rnorm(nPANR,meanPANR_2,sdPANR_2)
ks.test(resPANR_2,resnPANR_2)

nAALI_2=length(resAALI_2)
meanAALI_2=mean(resAALI_2)
sdAALI_2=sd(resAALI_2)
resnAALI_2=rnorm(nAALI,meanAALI_2,sdAALI_2)
ks.test(resAALI_2,resnAALI_2)

nICBP_2=length(resICBP_2)
meanICBP_2 = mean(resICBP_2)
sdICBP_2=sd(resICBP_2)
resnICBP_2=rnorm(nICBP,meanICBP_2,sdICBP_2)
ks.test(resICBP_2,resnICBP_2)

cat('H0 : residual berdistribusi normal') # Ketiga residual tidak berdistribusi normal

'''
library(writexl)
df_resPANR_2 = data.frame(resPANR_2)
cat('
write_xlsx(the dataframe name,"path to store the Excel file\\file name.xlsx")
')
write_xlsx(df_resPANR_2,"D:/OneDrive - sworepublic/Local Disk D
(Lenovo)/Data\\residualPANR ARIMA-GARCH.xlsx")

df_resAALI_2 = data.frame(resAALI_2)
write_xlsx(df_resSKLT_2,"D:/OneDrive - sworepublic/Local Disk D
(Lenovo)/Data\\residualAALI ARIMA-GARCH.xlsx")

df_resICBP_2 = data.frame(resICBP_2)
write_xlsx(df_resICBP_2,"D:/OneDrive - sworepublic/Local Disk D
(Lenovo)/Data\\residualICBP ARIMA-GARCH.xlsx")
'''
```

3. Mengambil Residual, Sigma, dan *Standardized Residual*

```
##
# Epsilon (residual)
resPANR_2 = fitARCHGARCH12PANR@fit$residuals
resAALI_2 = fitARCHGARCH11AALI@fit$residuals
resICBP_2 = fitARCH2ICBP@fit$residuals

# Sigma
sigmaPANR = fitARCHGARCH12PANR@fit$sigma
sigmaAALI = fitARCHGARCH11AALI@fit$sigma
sigmaICBP = fitARCH2ICBP@fit$sigma


# Z (standardized residual)
ZPANR = fitARCHGARCH12PANR@fit$z
ZAALI = fitARCHGARCH11AALI@fit$z
ZICBP = fitARCH2ICBP@fit$z
ZAset = cbind(ZPANR, ZAALI, ZICBP)

'''
library(writexl)
df_ZPANR = data.frame(ZPANR)
cat('
write_xlsx(the dataframe name,"path to store the Excel file\\file name.xlsx")
')
```

```
write_xlsx(df_ZPANR,"D:/OneDrive - sworepublic/Local Disk D (Lenovo)/Data\\Z PANR ARIMA-
GARCH.xlsx")

df_ZAALI = data.frame(ZAALI)
write_xlsx(df_ZAALI,"D:/OneDrive - sworepublic/Local Disk D (Lenovo)/Data\\Z AALI ARIMA-
GARCH.xlsx")

df_ZICBP = data.frame(ZICBP)
write_xlsx(df_ZICBP,"D:/OneDrive - sworepublic/Local Disk D (Lenovo)/Data\\Z ICBP ARIMA-
GARCH.xlsx")

'''
```

4. Membuat Plot Perbandingan Distribusi Empiris, Laplace, dan Normal

```
# Parameter Distribusi Normal
meanPANR = as.matrix(unlist(fitdistr(ZPANR, "normal")))[1]
stdPANR = as.matrix(unlist(fitdistr(ZPANR, "normal")))[2]

meanAALI = as.matrix(unlist(fitdistr(ZAALI, "normal")))[1]
stdAALI = as.matrix(unlist(fitdistr(ZAALI, "normal")))[2]

meanICBP = as.matrix(unlist(fitdistr(ZICBP, "normal")))[1]
stdICBP = as.matrix(unlist(fitdistr(ZICBP, "normal")))[2]

# Parameter Distribusi Laplace
thetaLaplacePANR = median(ZPANR)
bPANR = mean(abs(ZPANR-thetaLaplacePANR))

thetaLaplaceAALI = median(ZAALI)
bAALI = mean(abs(ZAALI-thetaLaplaceAALI))

thetaLaplaceICBP = median(ZICBP)
bICBP = mean(abs(ZICBP-thetaLaplaceICBP))
```

1) Plot Distribusi *Standardized Residual*

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

plot(density(ZPANR), main = "Distribution of PANR.JK", col = "blue",
     xlab = "Standardized Residual of PANR", ylim = c(0.0, 0.9), lwd = 2)
curve(dlaplace(x, thetaLaplacePANR,bPANR , log = FALSE), from=-6, to=6, add = T,
col="red", lwd = 2)
curve(dnorm(x, meanPANR ,stdPANR , log = FALSE), from=-6, to=6, add = T, col="green",
lwd = 2)
legend(x="topright", legend=c("Empiris", "Laplace","Normal"), col = c("blue", "red",
"green"), lty = 1, bty = "n")


plot(density(ZAALI), main = "Distribution of AALI.JK", col = "blue",
     xlab = "Standardized Residual of AALI", ylim = c(0.0, 0.9), lwd = 2)
curve(dlaplace(x, thetaLaplaceAALI,bAALI , log = FALSE), from=-6, to=6, add = T,
col="red", lwd = 2)
curve(dnorm(x,meanAALI     ,stdAALI , log = FALSE), from=-6, to=6, add = T, col="green",
lwd = 2)
legend(x="topright", legend=c("Empiris", "Laplace","Normal"), col = c("blue", "red",
"green"), lty = 1, bty = "n")


plot(density(ZICBP), main = "Distribution of ICBP.JK", col = "blue",
     xlab = "Standardized Residual of ICBP", ylim = c(0.0, 0.9), lwd = 2)
curve(dlaplace(x, thetaLaplaceICBP,bICBP , log = FALSE), from=-6, to=6, add = T,
col="red", lwd = 2)
curve(dnorm(x,meanICBP     ,stdICBP , log = FALSE), from=-6, to=6, add = T, col="green",
lwd = 2)
legend(x="topright", legend=c("Empiris", "Laplace","Normal"), col = c("blue", "red",
"green"), lty = 1, bty = "n")
```

2) Plot Distribusi Kumulatif *Standardized Residual*

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

plot(ecdf(ZPANR), main = "Distribusi Kumulatif PANR", col = "blue",
     xlab = "Standardized Residual PANR", ylim = c(0.0, 1), lwd = 2)
curve(plaplace(x, thetaLaplacePANR,bPANR ), from=-5, to=3, add = T, col="red", lwd = 2)
curve(pnorm(x,meanPANR ,stdPANR , log = FALSE), from=-5, to=3, add = T, col="green", lwd
= 2)
legend(x="right", legend=c("Empiris", "Laplace", "Normal"), col = c("blue", "red",
"green"), lty = 1, bty = "n")


plot(ecdf(ZAALI), main = "Distribusi Kumulatif AALI", col = "blue",
     xlab = "Standardized Residual AALI", ylim = c(0.0, 1), lwd = 2)
curve(plaplace(x, thetaLaplaceAALI,bAALI ), from=-5, to=3, add = T, col="red", lwd = 2)
curve(pnorm(x,meanAALI ,stdAALI , log = FALSE), from=-5, to=3, add = T, col="green", lwd
= 2)
legend(x="right", legend=c("Empiris", "Laplace", "Normal"), col = c("blue", "red",
"green"), lty = 1, bty = "n")


plot(ecdf(ZICBP), main = "Distribusi Kumulatif ICBP", col = "blue",
     xlab = "Standardized Residual ICBP", ylim = c(0.0, 1), lwd = 2)
curve(plaplace(x, thetaLaplaceICBP,bICBP ), from=-5, to=3, add = T, col="red", lwd = 2)
curve(pnorm(x,meanICBP ,stdICBP , log = FALSE), from=-5, to=3, add = T, col="green", lwd
= 2)
legend(x="right", legend=c("Empiris", "Laplace", "Normal"), col = c("blue", "red",
"green"), lty = 1, bty = "n")
```

5. Perhitungan Koefisien Korelasi Tau Kendall

```
corAset = cor(ZAset, method = "kendall")
corMatrix = matrix(corAset, nrow= 3, ncol=3)
paramCor = corAset[upper.tri(corAset)]

# Uji signifikansi koefisien korelasi
cor.test(unifPANR,unifAALI,alternative="two.sided",method="kendall")# signif
cor.test(ZPANR, ZICBP,alternative="two.sided",method="kendall")# signif
cor.test(ZAALI,ZICBP,alternative="two.sided",method="kendall") # signif
```

ANALISIS *COPULA*

1. Transformasi *Copule* ke *Uniform*

```
# Transformasi Copula ke Uniform
unifPANR = plaplace(ZPANR, thetaLaplacePANR,bPANR)
unifAALI = plaplace(ZAALI, thetaLaplaceAALI,bAALI)
unifICBP = plaplace(ZICBP, thetaLaplaceICBP,bICBP)
unifAset = cbind(unifPANR, unifAALI, unifICBP)

### Manual ###
unifPANR2=c()
for (i in 1:1004){
  if (ZPANR[i]<thetaLaplacePANR) {
    unifPANR2[i]=0.5*exp((ZPANR[i]-thetaLaplacePANR)/bPANR)
  } else {
    unifPANR2[i]=1-0.5*exp(-((ZPANR[i]-thetaLaplacePANR)/bPANR))
  }
}
unifPANR[1]
unifPANR2[1] # hasil sudah sama
```

2. D-Vine Copula

```
library("CDVineCopulaConditional")
library("VineCopula")
```

1) Menentukan urutan/*order copula*

```
corAset[1,2]+corAset[1,3]#PANR-AALI + PANR-ICBP (Ketiga)
corAset[2,1]+corAset[2,3]#AALI-PANR + AALI-ICBP (Terbesar)
corAset[3,1]+corAset[3,2]#ICBP-PANR + ICBP-AALI (Kedua)

library(HAC)
print(corAset)
sort(paramCor, decreasing = TRUE) #1. AALI-ICBP, PANR-AALI, PANR-ICBP
```

2) Menentukan D-Vine *Copula* terbaik

```
unifAset_sort= matrix(c(unifAset[,3],unifAset[,2],unifAset[,1]),1004,3)
RVM = CDVineCondFit(unifAset_sort, familyset = c(3:5) , rotations = FALSE,
                    treecrit = "AIC", Nx = 1, type = "DVine", method = "itau",
selectioncrit = "AIC")
summary(RVM)

# cek nilai parameter copula based on rumus itau
(2*corMatrix[3,2])/(1-corMatrix[3,2]) # nilainya sudah sama dengan D-Vine otomatis
(2*corMatrix[1,2])/(1-corMatrix[1,2])  # nilainya sudah sama dengan D-Vine otomatis
```

3) Plot D-Vine pada Setiap Pohon

```
plot(RVM,1)
plot(RVM,2)
```

PENGUKURAN EXPECTED SHORTFALL

1. Membangkitkan Data Sesuai Parameter yang Ada

```
cond = unifAset[,2]
simulation = CDVineCondSim(RVM, cond)
• Mengukur VaR dan ES Dengan Simulasi 1000 Pengulangan
```

2. Mengukur VaR dan ES DVine dengan Simulasi 1000 Iterasi

```
x1 = c()
x2 = c()
x3 = c()
VaRDvine = c()
ESDvine = c()
returnPorto =c()

set.seed(1)

for (i in 1:1000){
  # Memperoleh return iterasi ke-4
  u_cop = CDVineCondSim(RVM, cond)
  thetaLaplacePANR2 = median(u_cop[,3])
  thetaLaplaceAALI2 = median(u_cop[,2])
  thetaLaplaceICBP2 = median(u_cop[,1])
  bPANR2 = mean(abs(u_cop[,3]-thetaLaplacePANR2))
  bAALI2 = mean(abs(u_cop[,2]-thetaLaplaceAALI2))
  bICBP2 = mean(abs(u_cop[,1]-thetaLaplaceICBP2))
  z_cop = qlaplace(u_cop,
  m=c(thetaLaplaceICBP,thetaLaplaceAALI,thetaLaplacePANR),s=c(bICBP,bAALI,bPANR))
  z_PANR = z_cop[,3]
  z_AALI = z_cop[,2]
  z_ICBP = z_cop[,1]

  x1 = SeriesFcPANR + sigmaFcPANR*z_PANR
  x2 = SeriesFcAALI  + sigmaFcAALI*z_AALI
  x3 = SeriesFcICBP + sigmaFcICBP*z_ICBP
  return = cbind(x1,x2,x3);
  # untuk semua weight
  weightPANR =  0.25;
  weightAALI = 0.43;
  weightICBP = 0.32;
  weights = rbind(weightPANR, weightAALI, weightICBP);
  #pitfall
  returnPorto = return%*%weights;
  returnPorto_sort = sort(returnPorto)
  VaRDvine[i] = returnPorto_sort[51];
  ESDvine[i] = mean(returnPorto_sort[1:51]);
}

VaR = mean(VaRDvine)
ES = mean(ESDvine)
```