Financial Econometrics

Forecasting and Risk Analytics

1. Introduction

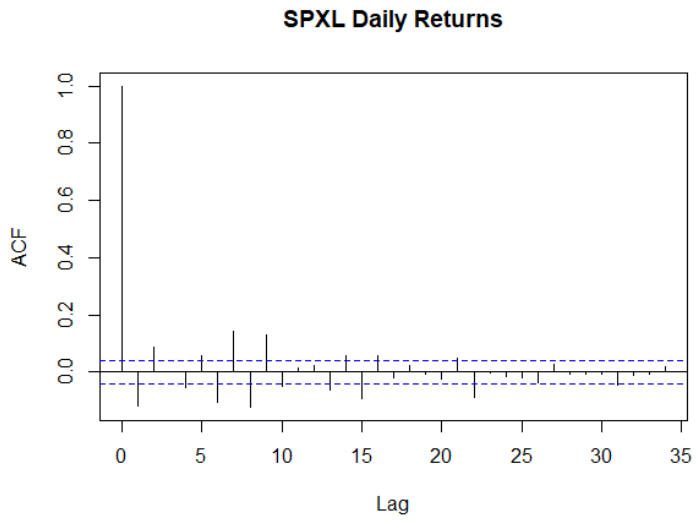
In this small case study, two stocks are evaluated them based on certain volatility models, their summary statistics and their forecasted loss in terms of value-at-risk (VaR) and expected shortfall (ES). These are namely AMD and SPXL, which tracks 3 times the daily performance of the SPX index fund. We use 10 years of daily data for both stocks and calculate their log returns from adjusted closing prices. As a result, we have 1000 entries for each of them.

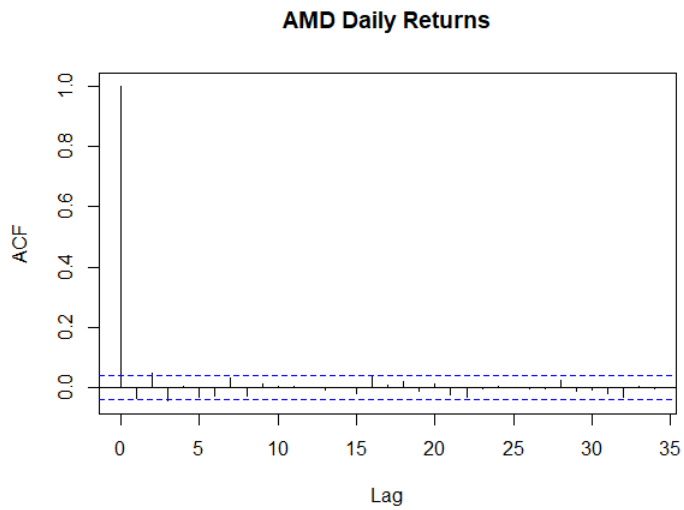
The primary goal of this assignment is to evaluate how risk analysis and forecasting can help individual investors and asset managers protect their portfolios, in according with different risk tolerance. Obviously, returns are primary goals for almost any appetite of risk tolerance but to properly account for risk analysis, requires a tedious approach and often involves large amount of computational time. To account for uncertainty, many analysts implement Monte Carlo or bootstrapping to adjust for every possible scenario so that the average values of risk metrics are reported. However, in this report I only show few scenarios regarding how some risk metrics may vary due to different weights of these two stocks. Before doing anything, we first take a look at our raw data

2. Modeling

Firstly, daily data is far from normal and sometimes be skewed and thus normality assumptions can easily be refuted. Before fitting any model, we would like to assess the stationary of the time series, indicated by their respective autocorrelation functions

Figure 1

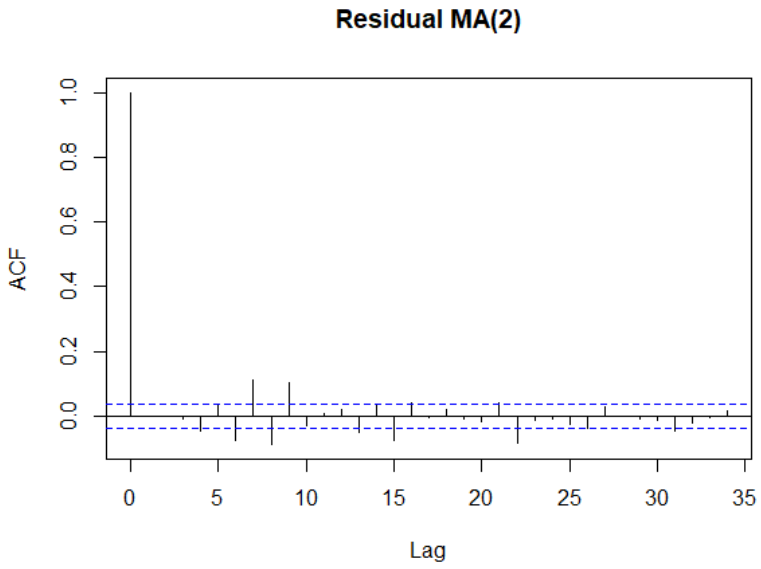


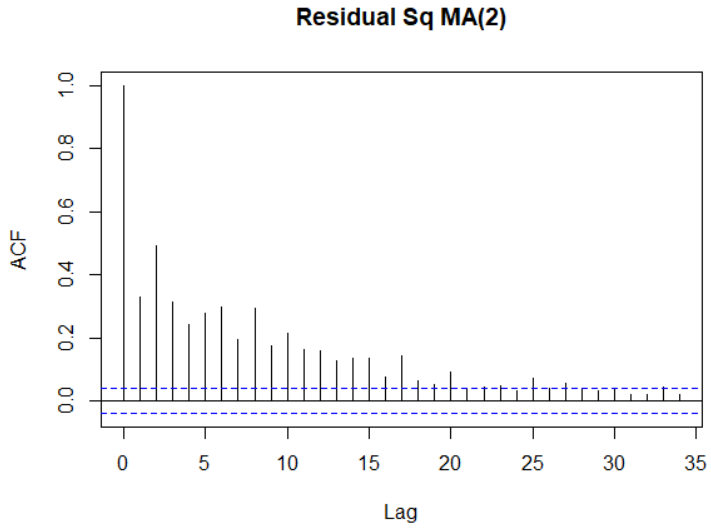


Although the AMD daily returns might seem stationary, SPXL return does not. To forecast or do any modeling with SPXL, we need to first take in account of stationarity by incorporating either first differences, AR or MA terms. If we just fit a MA (2) process to SPXL return data, we still see that the series does not exhibit white noise, as there are still spikes beyond the blue line. Similarly, volatility analysis can be shown through residual square. From the second plot of Figure 2, we see that the spikes decay gradually, at much later lags. Other combinations of AR & Consideration are also tried with Auto-ARIMA computation but does not yield homoskedasticity. It is essential to consider time dependencies and volatility and that is why AR-GARCH models are important at instance.

The AR-GARCH (Autoregressive Generalized Autoregressive Conditional Heteroskedasticity) are preferred because they are able to capture volatility clustering. The AR model is good enough to capture serial correlation but they cannot take account for changes in the variance over time. The AR model assume constant variance in the error terms but this is not logical for time series as they vary with time. It is often seen that period of high volatility tend to follow with high volatility and vice versa.

Figure 2





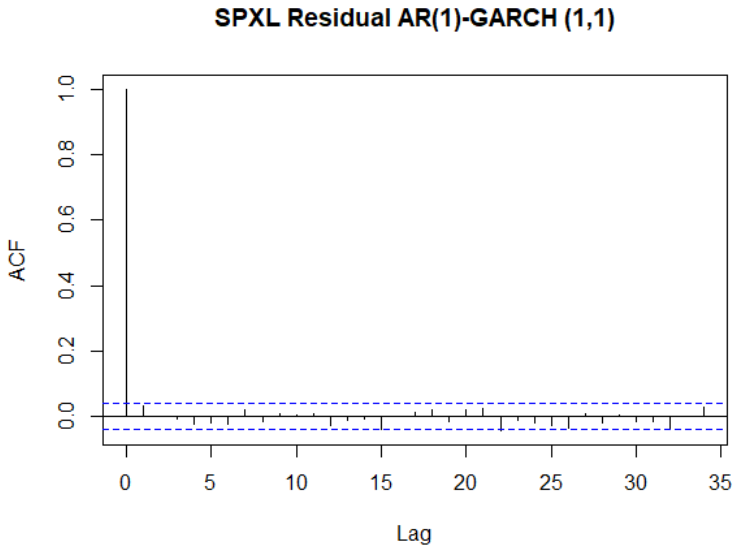
We start by fitting AR(1)-GARCH (1,1) model on AMD and SPXL. After fitting the model, we extract the residuals and fit a t-distribution. The following table shows the estimates for AMD & SPXL residuals

Table 1

	Mean	Standard deviation	Degrees of Freedom (df)
AMD	-0.01716731	0.69049286	3.81114031
SPXL	-0.008514616	0.803515091	5.613651485

We also now test the residuals of the fitted models above to see if they exhibit white noise or not enough to catch serial correlation.

Figure 3



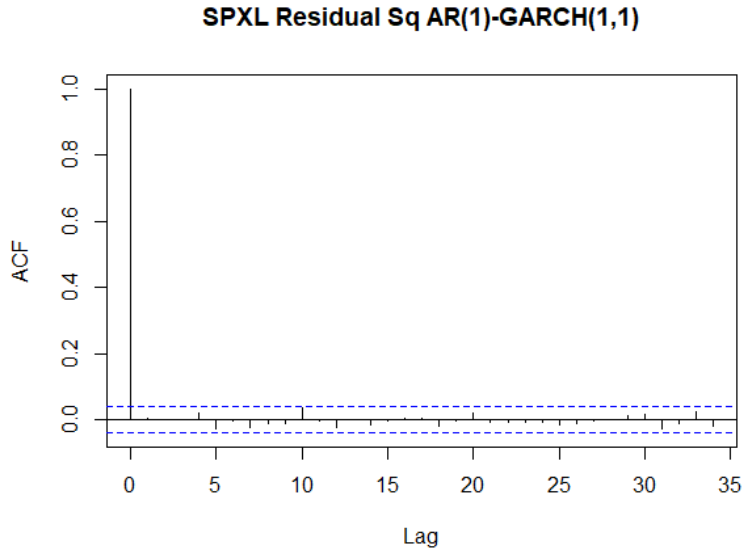
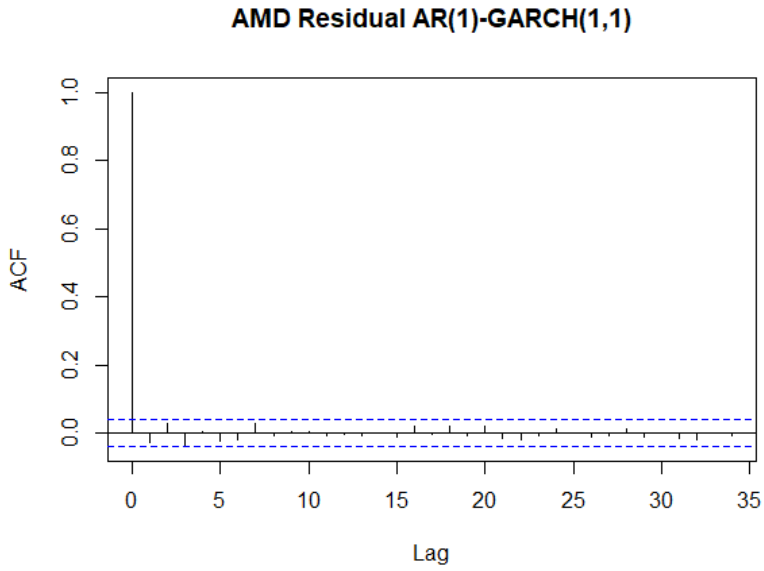
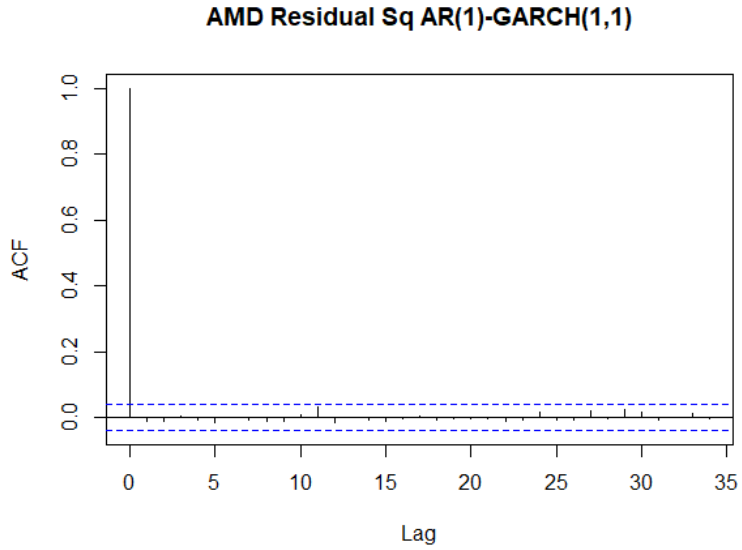

SPXL Residual Sq AR(1)-GARCH(1,1)
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
35
Lag

Figure 4


AMD Residual AR(1)-GARCH(1,1)
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
35
Lag



Both Figures 3 and 4 indicate that we observe white noise on both of these models as well have as homoskedastic residuals instead of heteroskedastic. Hence, we can use this model to forecast risk and expected losses from the returns data.

Our next step is to use copula model to estimate their dependence structure from their marginals. We can't directly use AR-GARCH estimates because they are conditional and we could not have used the returns directly due to time dependency and volatility clustering. The reason behind using these residuals in copula is because we are interested in capturing the unpredicted portion, i.e. the portion that is not captured by the AR(1)-GARCH (1,1) model after accounting time dependency and heteroskedasticity.

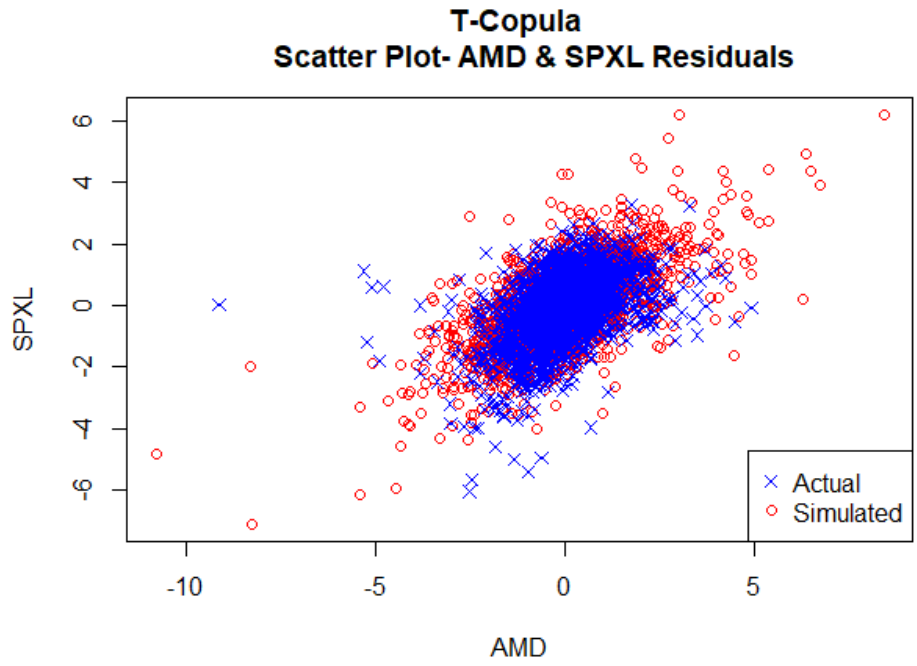
As for the next step, we use copula models namely: T, Clayton, Gaussian & Gumbel. The following table reports the transformed uniformly distributed residuals' estimates:

Table 2

Model	Estimates	AIC
T-Copula	ρ=0.6418775, df=7.1109967	-841.8355
Clayton Copula	α=1.056993	-661.855
Gaussian Copula	ρ=0.6410975	-802.677
Gumbel Copula	α=1.753815	-767.7808

From the above table, we know that T-Copula represents the best way to simulated the random component of the returns.

Figure 5



The second and the crucial step that all these results will come in handy for is evaluating and forecasting risk analysis. That is, we need forecasted volatility one day ahead, i.e. σ_{n+1}. The exact way to obtain this has been shown in the appendix, which is from AR(1)-GARCH(1,1) model. The forecasted standard deviation for AMD is 0.0348 and 0.0273 for SPXL.

Before calculating the portfolio VaR or ES, we need the residuals or the shocks to be scaled properly that is multiplying them with GARCH component of the volatility. By doing this, we are essentially scaling them to the level, the GARCH model expects them to be.

Just like the forecasted standard deviation, we can also get the forecasted expected return in the same manner. Consequently, $\widehat{X}_{n+1}$ is obtained for SPXL and AMD. Their expected returns for one day ahead are 0.0014 and 0.0027.

Adding those two components each for SPXL and AMD, we can then make a for loop based on the weights ranging from 0.1 to 0.9 and then calculate VaR and ES. The Value-at-Risk quantifies the potential loss that portfolio might incur over a period of time, given confidence level. But since our data is dealt in returns not losses, reporting will be slightly different but the primary essence of the concept remains the same.

Table 3

Weights (rho)	99% VaR
0.1	-0.0842
0.2	-0.0863
0.3	-0.0916
0.4	-0.0966
0.5	-0.0992
0.6	-0.1037
0.7	-0.1104
0.8	-0.1165
0.9	-0.1269

N.B: Negative values are shown due to returns (gain distribution) as opposed to losses (loss distribution)

So we see that as we increase the weights towards AMD, the portfolio VaR increases non-linearly. Clearly, this means that having more of AMD increases our tail risk significantly.

As for ES, the interpretation is different. It defines the average loss greater than VaR threshold. The VaR does not say anything about size of the losses in the tail risk while ES does. The results follow a similar trend at 97.5% confidence interval. As weights towards AMD increases, portfolio ES increases

Table 4

Weights (rho)	97.5% ES
0.1	-0.0922
0.2	-0.0945
0.3	-0.0982
0.4	-0.1032
0.5	-0.1088
0.6	-0.1150
0.7	-0.1222
0.8	-0.1300
0.9	-0.1381

N.B: Negative values are shown due to returns (gain distribution) as opposed to losses (loss distribution)

3. Conclusion

From the results of risk analysis and its prediction, we can state if someone is risk averse, they ought to put more investment allocation towards SPXL and AMD because the ETF is more diversified and stable than AMD, a single stock. But remember, SPXL is a leveraged ETF. If the index of SPX goes by 1%, SPXL goes up by 3%. Same for losses. There is a multiplicative effect. Just because it is well diversified than AMD, does not mean its volatility is miniscule. Perhaps, it is advisable to stay within the range of 0.1-0.4 for a risk averse investor.

Appendix-Code

```
library(copula)
library("fGarch")
library("forecast")
library(quantmod)
library(MASS)
library(fitdistrplus)
getSymbols("SPXL",src='yahoo',auto.assign = TRUE,from='2013-09-29',to='2023-09-29')
getSymbols("AMD",src='yahoo',auto.assign = TRUE,from='2013-09-29',to='2023-09-29')
spxl_r=dailyReturn(SPXL$SPXL.Adjusted,type='log')
amd_r=dailyReturn(AMD$AMD.Adjusted,type='log')
spxl_r=coredata(spxl_r)
amd_r=coredata(amd_r)
plot(amd_r)
points(spxl_r)
## Preliminary Stats
plot(spxl_r)
plot(amd_r)
par(mfrow=c(1,1))
acf(spxl_r,main='SPXL Daily Returns')
acf(amd_r, main='AMD Daily Returns')
summary(amd_r)
summary(spxl_r)
##Let's Try seeing the residuals with AR(1) and validate the justification of time dependence & volatility clustering
fitAR1_s=arima(spxl_r,order=c(0,0,2),include.mean=TRUE)#fit ARIMA model
fitAR1_s
res_s=residuals(fitAR1_s)
acf(res_s,main='Residual MA(2)') ## Not white noise. Therefore, it is more important to capture the time dependencies
acf(res_s^2,main='Residual Sq MA(2)') ## Numerous spikes at variety of lags
##AR(1)+GARCH(1,1)
fit_s=garchFit(formula=~arma(1,0)+garch(1,1),data=spxl_r,cond.dist="norm",trace=FALSE)
fit_a=garchFit(formula=~arma(1,0)+garch(1,1),data=amd_r,cond.dist="norm",trace=FALSE)
fit_a
res_a=residuals(fit_a,standardize=TRUE) ##White noise
res_s=residuals(fit_s,standardize=TRUE)
fit_ta=fitdistr(res_a,'t',upper=10)
fit_ts=fitdistr(res_s,'t',upper=10)
fit_ts
par(mfrow=c(1,1))
acf(res_s,main='SPXL Residual AR(1)-GARCH (1,1)')
```

```
acf(res_s^2, main='SPXL Residual Sq AR(1)-GARCH(1,1)')
acf(res_a,main='AMD Residual AR(1)-GARCH(1,1)')
acf(res_a^2,main='AMD Residual Sq AR(1)-GARCH(1,1)')
## Copula modeling

est1=fit_ta$estimate
est2=fit_ts$estimate

u1=pt(res_a,df=est1['df'])
u2=pt(res_s,df=est2['df'])

##Combining uniform data
uhat=cbind(u1,u2)

##Copula
Ct=fitCopula(copula=tCopula(dim=2),data=uhat,method="ml")## T Copula
Ct@estimate
Ct_logL=loglikCopula(param=Ct@estimate,u=uhat,copula=tCopula(dim=2));#compute loglikelihood function
Ct_logL
-2*Ct_logL+2*length(Ct@estimate)#AIC
Cg=fitCopula(copula=normalCopula(dim=2),data=uhat,method="ml")#fit Gaussian copula
Cg@estimate;
Cg_logL=loglikCopula(param=Cg@estimate,u=uhat,copula=normalCopula(dim=2));
Cg_logL
-2*Cg_logL+2*length(Cg@estimate);
Ccl=fitCopula(copula=claytonCopula(dim=2),data=uhat,method="ml");#fit clayton copula
Ccl@estimate
Ccl_logL=loglikCopula(param=Ccl@estimate,u=uhat,copula=claytonCopula(dim=2));
Ccl_logL
-2*Ccl_logL+2*length(Ccl@estimate);#compute AIC

Cgmbl=fitCopula(copula=gumbelCopula(dim=2),data=uhat,method="ml");#fit gumbel copula
Cgmbl@estimate
Cgmbl
Cgmbl_logL=loglikCopula(param=Cgmbl@estimate,u=uhat,copula=gumbelCopula(dim=2))
Cgmbl_logL
-2*Cgmbl_logL+2*length(Cgmbl@estimate);
```

```r
##So T Copula is the best
Ct@estimate
rho=Ct@estimate[1]
correlation_matrix=matrix(c(1,rho,rho,1),nrow=2)
correlation_matrix
confidence_level <- 0.99
n=length(amd_r)
Simu_U=rCopula(n,tCopula(dim=2,Ct@estimate[1],df=Ct@estimate[2]))
Simu_X1=qt(Simu_U[,1],df=est1['df'])
Simu_X2=qt(Simu_U[,2],df=est2['df'])
plot(Simu_X1,Simu_X2,xlab='AMD',ylab='SPXL',main='T-Copula
     Scatter Plot-Simulated AMD & SPXL Residuals ',col='red')
points(res_a,res_s,col='blue',pch=4)
summary(Simu_U)

forecasted_s=predict(fit_s,n.ahead=1)
forecasted_a=predict(fit_a,n.ahead=1)

forecasted_volatility_a=forecasted_a$standardDeviation
forecasted_volatility_s=forecasted_s$standardDeviation

forecasted_s$meanForecast
forecasted_a$meanForecast## Scaling
scaled_residuals_x1=Simu_X1*forecasted_volatility_a
scaled_residuals_x2=Simu_X2*forecasted_volatility_s


weight_values=seq(0.1,0.9,by=0.1)
n_simulations=10000
VaR_estimates=numeric(length(weight_values))
VaR_estimates

for (i in seq_along(weight_values)){
  weight=weight_values[i]
  portfolio_returns=weight*(forecasted_a$meanForecast+scaled_residuals_x1)+(1-weight)*(forecasted_s$meanForecast+scaled_residuals_x2)
  VaR_estimates[i]=quantile(portfolio_returns,1-confidence_level)
}
VaR_estimates
##Expected Shortfall
```

```r
##Expected Shortfall
confidence_level=0.975
ES_estimates=numeric(length(weight_values))
VaR_estimates=numeric(length(weight_values))
for (i in seq_along(weight_values)){
  weight=weight_values[i]
  portfolio_returns=weight*(forecasted_a$meanForecast+scaled_residuals_x1)+(1-weight)*(forecasted_s$meanForecast+scaled_residuals_x2)
  VaR_estimates[i]=quantile(portfolio_returns,1-confidence_level)
  ES_estimates[i]=mean(portfolio_returns[portfolio_returns<VaR_estimates[i]])}

ES_estimates
```