Final Report: Risk Analysis Using Econometric Models

Asmin Acharya

12-04-2024

Description

This report provides an analysis of two stocks (SPXL, MSFT) . Various types of statistical methods and models are used in order to understand the individual behavior of returns and understand the relationship between both stocks. The results of the analysis will provide sufficient evidence to make cogent claims about each stock, their returns, and risk profile.

Introduction

The aim of this project is to assess and manage financial risk of two different stocks. We will use the quantmod package in R and apply certain econometric techniques and models to the daily log returns of SPXL and Microsoft (MSFT) over a recent ten year period. Before starting on the main analysis, we will first explore the datasets and gain critical insights into serial correlation, volatility, and cross-sectional dependencies of the two stocks. This is a very important step as it gives us clear evidence about certain statistical characteristics that our data exhibits. Instead of making assumptions, we can first explore the data in order to understand the statistical properties of the stock returns. This would include information about the data such as stationarity, identifying serial correlation, and examining how the returns of the two stocks are correlated to each other. Exploring these essential features will allow us to accurately capture the inherent risk dynamics of these stocks.

Once we are done with exploring the data, we will move onto creating and fitting an AR(1)-GARCH(1,1) model accounting for time-series properties such as autocorrelation and conditional heteroscedasticity of each stock. Next, we will fit standardized t-distributions to the residuals of these models and explore several parametric copula families like the t-copula, Gaussian copula, Gumbel copula, and Clayton copula. The optimal copula will be selected based on the Akaike Information Criterion (AIC). Finding the best fit copula will give us vital information about the dependency structure between the daily log returns of SPXL and MSFT stocks. Afterwards, we will conduct residual analysis in order to evaluate whether the AR(1)-GARCH(1,1) model is a good fit for our data . If the residuals do not exhibit a distinct pattern, are uncorrelated, and are stable, then our analysis will be sufficient in confirming the model's suitability.

On the condition that our model is suitable, we move onto simulating risk measures like VaR and ES to capture the full range of possible losses, accurately model dependencies, and evaluate risks under different allocation scenarios. We will simulate VaR at the 99% confidence level and Expected Shortfall (ES) at the 97.5% level for various portfolio allocations. These risk estimates are calculated across portfolio weights which reflect a range of risk tolerance levels. This will provide us with insights as to how varying allocations impact portfolio risk.

By carefully assessing model adequacy through residual analysis and exploring risk metrics under diverse portfolio scenarios, this project aims to deliver an extensive assessment of potential financial risks associated with SPXL and MSFT stocks

Gathering Data

The first thing we have to do is gather our financial data using the quantmod package in R. We get daily stock price data for SPXL and MSFT from Yahoo Finance for our specified date range, which will be from 10 years ago up to present day. For each stock, we will make sure to extract the adjusted closing prices from our data. In general, it is best to use adjusted prices as it provides a more accurate reflection of price changes due to certain market movements. We will then calculate the daily log returns of SPXL and MSFT and remove the first missing value, which is caused from differencing. Using log returns is the best in this situation as it makes it easier to work with. In addition, it normalizes the changes in the stock prices which is much for suitable when using financial returns for statistical analysis. Now that we have gathered our data, we can go forward to the next step which is to explore our dataset.

Exploring the Data

As we explore our dataset, we will be able to understand its basic characteristics (mean, standard deviation, skewness, and kurtosis) , identify key patterns and relationships, and recognize dependencies in the SPXL and MSFT returns. Looking at the summary statistics we find that the mean log returns of SPXL (0.000886) and MSFT (0.000915) are both positive. These values represent the average daily return, which indicates that there is a slight upward trend in prices over time. SPXL (0.0338) has a higher standard deviation compared to MSFT (0.0171), suggesting that SPXL's returns in general are more volatile. Both SPXL (-1.39) and MSFT (-0.17) have negative skewness, but SPXL has a skewness that is significantly more negative.

The negative skewness indicates that SPXL has as strong left tail which translates to higher risk of extreme negative returns. MSFT has a slight negative skewness, so it has a more mild left tail compared to SPXL. MSFT is relatively less prone to extreme negative movements. The high kurtosis value of SPXL (18.81) is evidence that it has extreme tails. We can conclude that with this high kurtosis value, SPXL's returns experience more frequent and severe spikes than would be expected under a normal distribution. MSFT also has a high kurtosis value (8.05), but is less compared to SPXL. This implies that MSFT returns also have more extreme movements than normal. These movements, however, are less severe compared to SPXL which is more volatile and more prone to extremes.

We have gathered some basic statistical information on SPXL and MSFT. Next, using statistical tests and visualizations we will argue and see if it is necessary to model serial correlation, volatility, and cross-sectional dependence of log returns. The Ljung-Box test examines if there is significant autocorrelation in our series. As long as our p-value is less than 0.05, it suggests that the returns are autocorrelated. Based on our output, we can see that in the analysis both SPXL and MSFT have p-values < 0.05 which is strong evidence that serial correlation is present. Additionally, if we look at the ACF plots of both the SPXL and MSFT returns we can see that there are several spikes that exceed the blue dotted line, which represent the significance bounds. This is also evidence of serial correlation. The top left plot represents the time series of our daily log returns for SPXL and MSFT. The overall magnitude of returns is smaller for MSFT compared to SPXL where the y-axis scale for MSFT is approximately -0.15 to 0.05 vs SPXL's -0.4 to 0.2. The returns plot shows significant volatility clustering, where there are periods of high volatility followed by periods of low volatility. A little bit left of the time axis or around observation 1500 we can see a notable spike. SPXL and MSFT returns seem to have a consistent pattern of fluctuations around 0, but with varying intensity over time. This time series plot in itself is a visual evidence of heteroskedasticity or changing variance over time. The ACF plot of the squared returns on the bottom right shows strong and persistent autocorrelation in squared returns. We can see that many of the lags are significant and have a slow decaying pattern. This is also clear evidence of volatility clustering. Lastly, if we look at the Q-Q plot on the top right we can check how the distribution of our returns compare to a normal distribution. Even though we have already computed the skewness and kurtosis values, it is important that we also use Q-Q plots. Using the statistical measures and the plots together gives us a complete picture of non-normality. The Q-Q plot of SPXL shows that the tails of the distribution deviate significantly from the normal distribution, especially in the negative tail. This indicates heavier tails than a normal distribution would predict. The presence of more pronounced extreme values provides further evidence that the distribution of SPXL returns is not normal. Similarly, the Q-Q plot for MSFT shows that both tails deviate from the normal distribution, but less severely than SPXL. We see a more symmetric deviation pattern for MSFT compared to SPXL. The Q-Q plot for MSFT still suggests heavier tails than normal distribution.

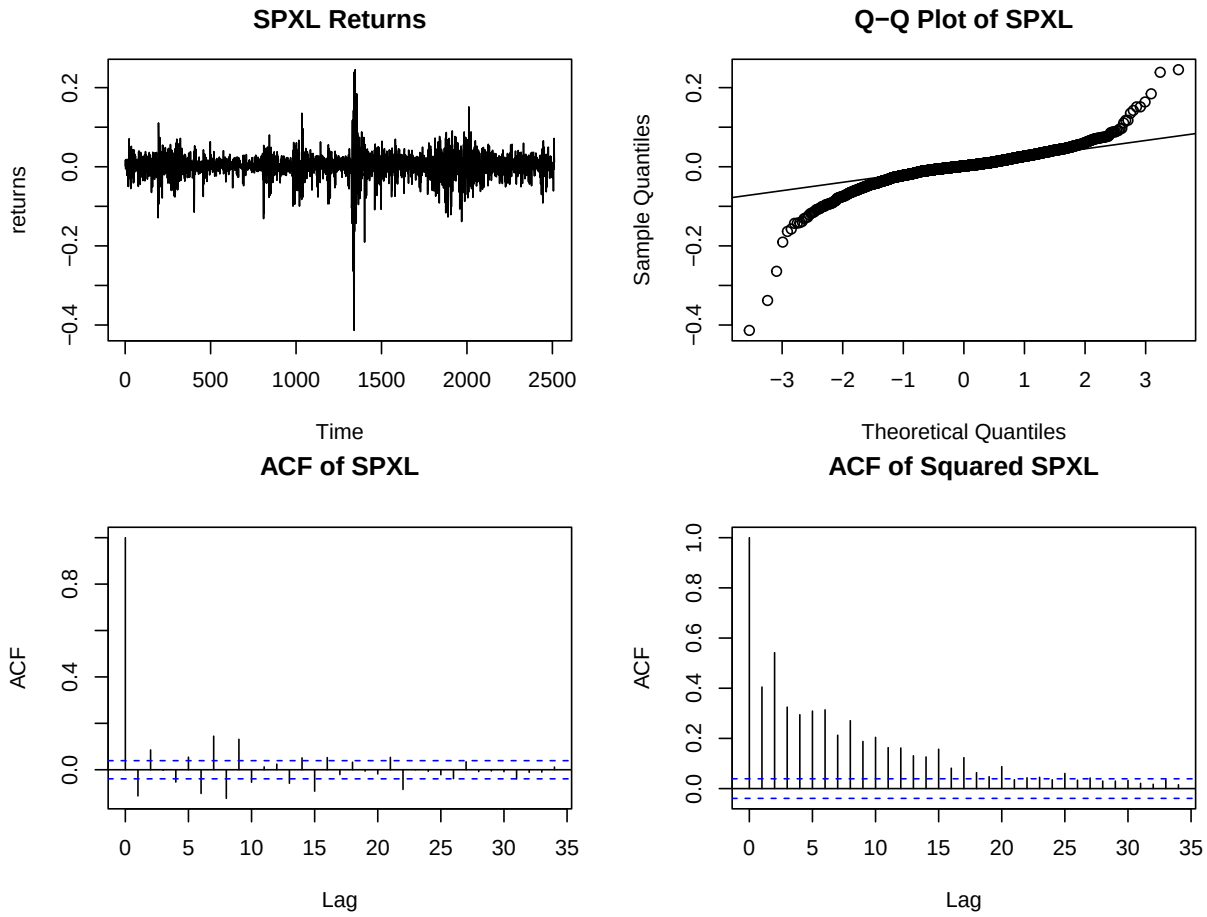
The scatterplot analysis and the correlation measures will help us interpret the cross-sectional dependence between SPXL and MSFT. The scatterplot shows that there is a clear positive relationship between the SPXL and MSFT returns where the points cluster along a diagonal pattern. This indicates that both stocks tend to move in the same direction. We can see that there is a dense concentration around (0,0) and the tails are more scattered. There are some extreme outliers in the plot, especially near the bottom left negative region. Both of the correlation measures indicate that a positive association holds between the two stocks. There are, however, some key differences between the Pearson and Spearman correlations. The Pearson correlation (0.7946) measures the linear relationship between the actual returns, is sensitive to extreme values, and assumes normal distribution. The Spearman correlation (0.7502) measures monotonic relationship using ranks, which is less sensitive to outliers, and is a distribution free measure. It is clear to see that the Spearman correlation measure is slightly lower than the Pearson measure. The positive difference between these values suggests that some non-linearity exists in the relationship. As a result, we cannot assume that the relationship between SPXL and MSFT is purely linear which means we must consider non-linear dependencies. In addition to using the correlation measures, we also need to account for tail dependence which will provide us crucial information about extreme co-movements which the correlation measures alone cannot capture. The empirical lower tail dependence coefficient represents the likelihood of observing simultaneous extreme losses in both assets. The lower tail coefficient (0.6131) suggests that there is a 61.31% chance of joint extreme losses. The empirical upper tail dependence represents the likelihood of seeing large, simultaneous gains in both assets. The upper tail coefficient (0.4936) suggests that there is a

49.36% chance of joint extreme gains. The lower tail dependence is higher than the upper tail dependence, emphasizing that in times of extreme market crashes, SPXL and MSFT are likely to experience large losses simultaneously.

Based on the result of our statistical tests and the ACF plots, there is strong evidence to support the use of an AR component in our model due to the presence of serial correlation in both stocks. Furthermore, the clear evidence of volatility clustering strongly supports the need for a GARCH model. These findings strongly support our proposed AR(1)-GARCH(1,1) model with t-distributed errors. Additionally, based on our cross sectional dependence analysis we have seen that a linear correlation alone is not sufficient to understand the relationship between SPXL and MSFT. Tail dependence needs to be taken into account in order to gain additional information about extreme co-movements that the correlation measures alone cannot capture. This is where we have to consider using more sophisticated models like copulas which can better capture non-linear relationships and tail dependence. We will make sure to fit our model to each series of log returns and find the best fit copula based on the AIC value.

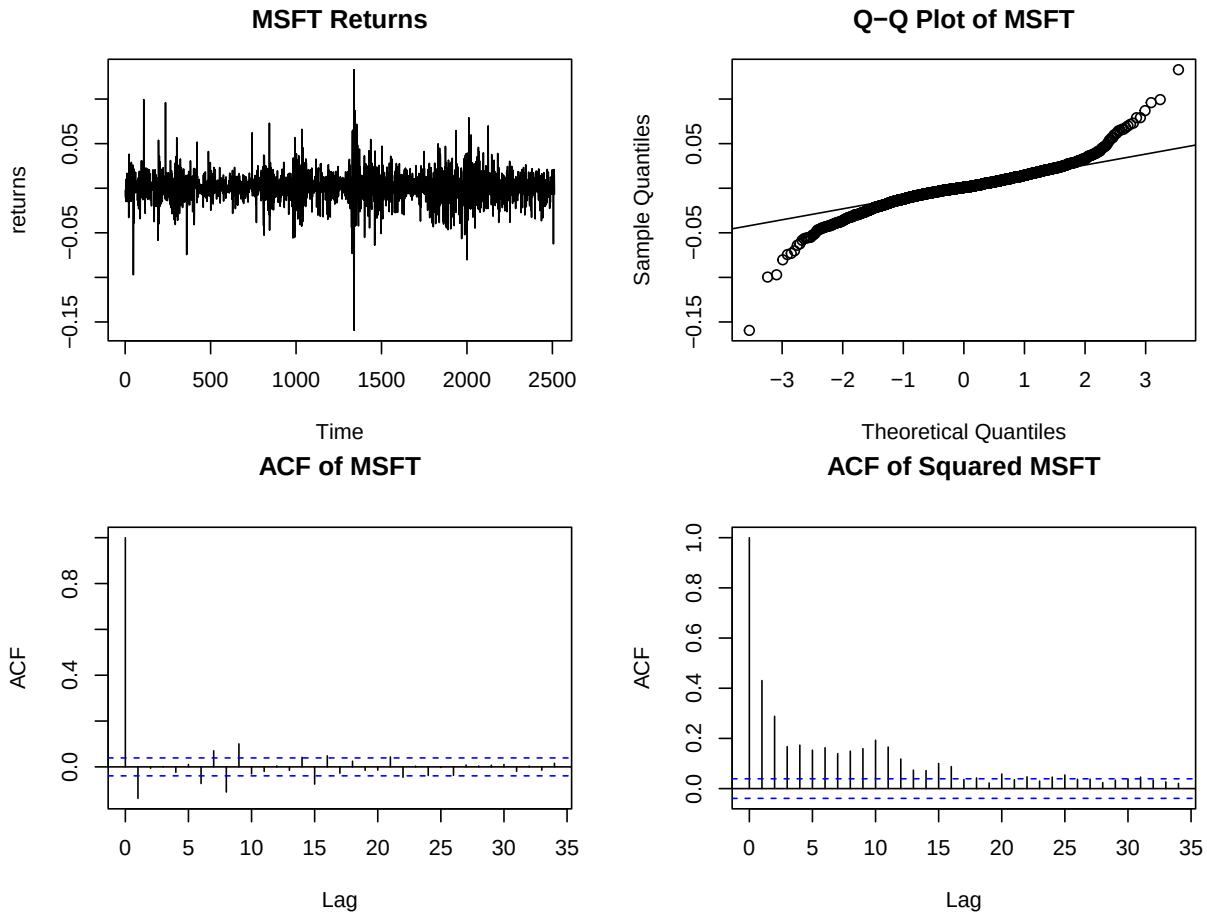
```r
# Adjust margins to reduce white space
par(mfrow = c(2, 2), mar = c(4, 4, 3, 2) + 0.1)

explore_data(all_returns$SPXL, "SPXL")
```

```
##
##  Analysis for SPXL
## Mean:  0.0008864595
## Std Dev:  0.03381209
## Skewness:  -1.39306
## Kurtosis:  18.81363
##
## Ljung-Box Test for Serial Correlation:
##
##  Box-Ljung test
##
## data:  returns
## X-squared = 31.917, df = 1, p-value = 1.609e-08
##
##
## Ljung-Box Test for ARCH effects (squared returns):
##
##  Box-Ljung test
##
## data:  returns^2
## X-squared = 411.96, df = 1, p-value < 2.2e-16
```

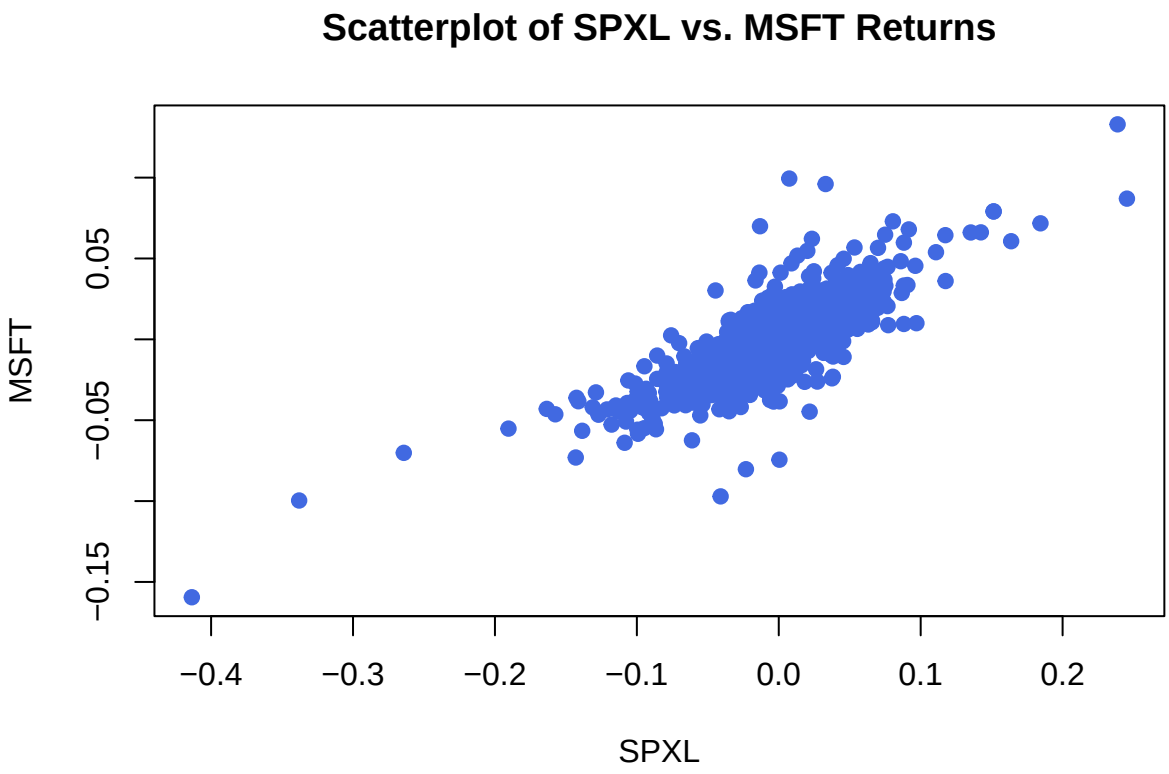



```
explore_data(all_returns$MSFT, "MSFT")
```

```
##
##  Analysis for MSFT
## Mean:  0.0009153377
## Std Dev:  0.0171025
## Skewness:  -0.1686723
## Kurtosis:  8.05074
##
## Ljung-Box Test for Serial Correlation:
##
##  Box-Ljung test
##
## data:  returns
## X-squared = 47.528, df = 1, p-value = 5.422e-12
##
##
## Ljung-Box Test for ARCH effects (squared returns):
##
##  Box-Ljung test
##
## data:  returns^2
## X-squared = 466.3, df = 1, p-value < 2.2e-16
```




```
explore_dependence(all_returns$SPXL, all_returns$MSFT, "SPXL", "MSFT")
```




```
##
## Pearson Correlation: 0.7946179
## Spearman Correlation: 0.7501614
##
## Empirical lower tail dependence coefficient: 0.6130573
## Empirical upper tail dependence coefficient: 0.4936306
```

Best Fit Distribution

Now that we have successfully explored our data, we can move onto fitting certain distributions to our SPXL and MSFT returns. The four distributions we will look at include: normal distribution, t-distribution, skew normal distribution, and skew t-distribution. The following distributions will be fitted to our daily log returns for each stock and the best fitting distribution will be obtained based on the lowest AIC value. In our code, we define a function for each distribution which will fit the distribution to our returns, calculate the log-likelihood of the fitted distribution, and then compute the AIC. Another function is defined, which will fit each of the four distributions and store our computed AIC values in a data frame. The variables spxl_analysis and msft_analysis call the function where we extract the daily log returns of SPXL and MSFT from our dataset. The variables return the AIC values in order of each fitted distribution, which will allow us to assess the best fitting distribution. Based on our output, the best fitting distribution for both returns was the skew t-distribution. The skew t-distribution had the lowest AIC (-10748.80) for SPXL and the skew t-distribution had the lowest AIC for MSFT as well (-13834.00).

In addition to evaluating the AIC values, we can also look at the observed density plot and the Q-Q plot of each return. The Q-Q plot of SPXL shows that there is noticeable deviation from the red line at both tails. This deviation is particularly more pronounced in the lower tail, suggesting a heavier left tail than predicted. The observed density plot of SPXL shows that the density is highly concentrated around zero. There is

also asymmetry, where the left tail is slightly longer. The observed density plot (black line) shows higher peaks compared to what we would find with a normal distribution. The Q-Q plot of MSFT shows that it has better adherence to the theoretical line and has smaller deviations at the tails compared to SPXL. The observed density plot of MSFT shows that it is less peaked, is more symmetric, and has a narrower range of returns compared to SPXL. From evaluating these density plots, we can see that the distributions for SPXL and MSFT are leptokurtic in nature. The distribution has heavier tails and a more peaked center compared to a normal distribution. Furthermore, when we were exploring our data, we found that the kurtosis value of these returns was far greater than three. The leptokurtic nature and the skewness of these distributions supports our earlier claim that the skew-t distribution is the best fit for both returns.

The fact that the skew t-distribution is the best fit for both SPXL and MSFT daily log returns provides us with valuable insights about the behavior of these stocks. The negative skewness implies that the distribution has a longer tail on the left which indicates that there is potential for larger than expected negative returns. Investors should be aware of the fact that these stocks experience more sharp declines than sharp increases. The high kurtosis values of the distribution of returns indicates heavy tails. This means that there are more extreme values in the SPXL and MSFT distribution compared to a normal distribution. Both stocks are prone to occasional large fluctuations due to certain economic or market related factors. These large fluctuations can result in sudden gains or losses, but due to the negative skewness, there is a greater concern for potential losses. It is safe to conclude that based on our best fit distribution (skew t) SPXL and MSFT demonstrate characteristics of increased risk and unpredictability.

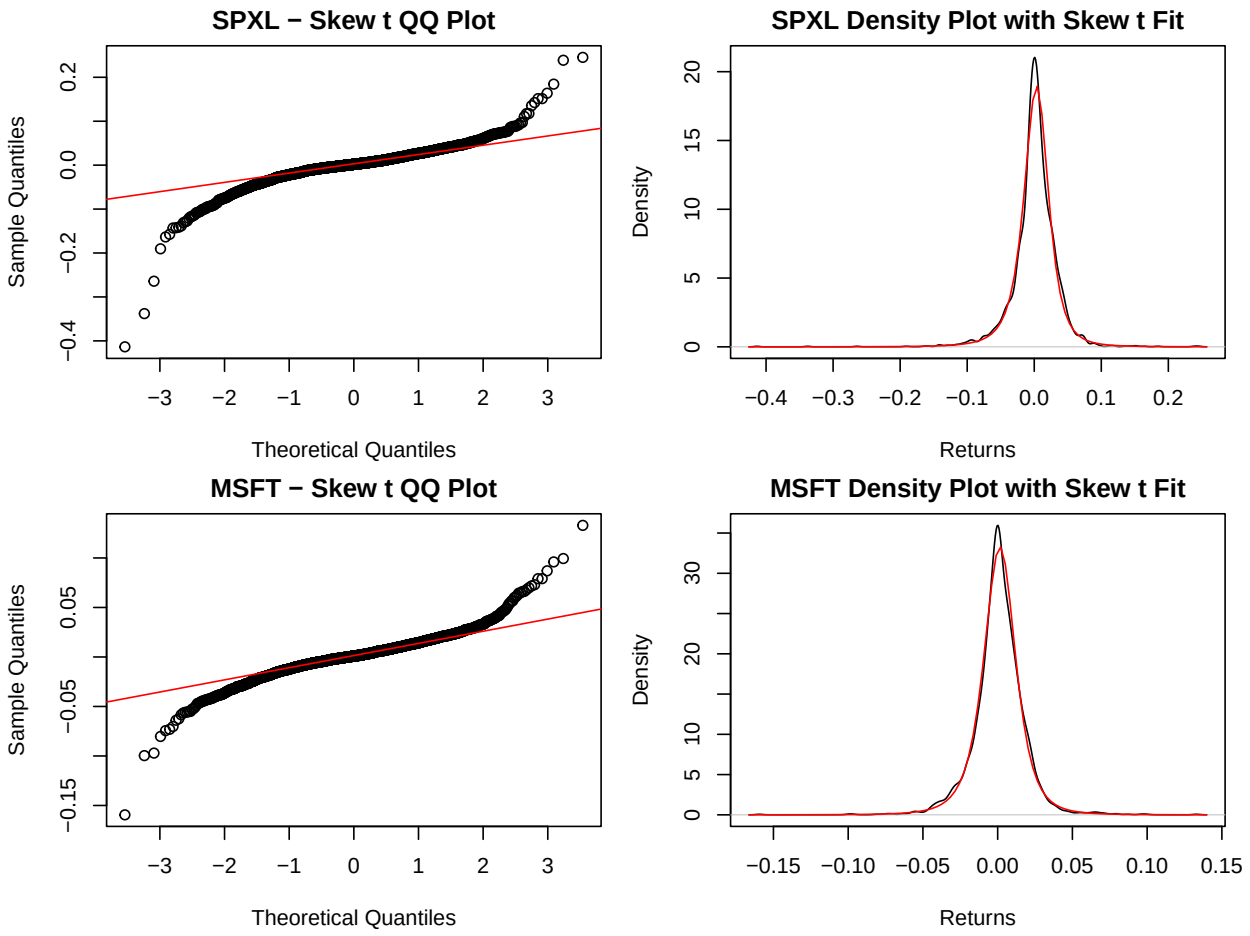
```
# Analyze best fit distribution for both stocks
spxl_analysis <- compare_distributions(all_returns$SPXL, "SPXL")
```

```
##
## Results for SPXL :
##   Distribution        AIC
## 4       Skew t -10748.803
## 2  Student's t -10743.494
## 3  Skew Normal -10000.402
## 1       Normal  -9884.224
```

```
msft_analysis <- compare_distributions(all_returns$MSFT, "MSFT")
```

```
##
## Results for MSFT :
##   Distribution       AIC
## 4       Skew t -13833.99
## 2  Student's t -13728.65
## 3  Skew Normal -13322.19
## 1       Normal -13308.55
```

```r
# Set up the layout
layout(matrix(c(1,2,3,4), nrow = 2, ncol = 2, byrow = TRUE))

# Adjust margins to reduce white space
par(mar = c(4, 4, 2, 1))  # Bottom, left, top, right

# SPXL QQ plot
best_dist_spxl <- spxl_analysis$results$Distribution[1]
qqnorm(all_returns$SPXL, main = paste("SPXL -", best_dist_spxl, "QQ Plot"))
qqline(all_returns$SPXL, col = "red")

# SPXL Density plot with theoretical fit
plot(density(all_returns$SPXL),
     main = paste("SPXL Density Plot with", best_dist_spxl, "Fit"),
     xlab = "Returns")
curve(dst(x,
          xi = spxl_analysis$skew_t$fit@param$dp["xi"],
          omega = spxl_analysis$skew_t$fit@param$dp["omega"],
          alpha = spxl_analysis$skew_t$fit@param$dp["alpha"],
          nu = spxl_analysis$skew_t$fit@param$dp["nu"]),
      add = TRUE, col = "red", lwd = 1)

# MSFT QQ plot
best_dist_msft <- msft_analysis$results$Distribution[1]
qqnorm(all_returns$MSFT, main = paste("MSFT -", best_dist_msft, "QQ Plot"))
qqline(all_returns$MSFT, col = "red")

# MSFT Density plot with theoretical fit
plot(density(all_returns$MSFT),
     main = paste("MSFT Density Plot with", best_dist_msft, "Fit"),
     xlab = "Returns")
curve(dst(x,
          xi = msft_analysis$skew_t$fit@param$dp["xi"],
          omega = msft_analysis$skew_t$fit@param$dp["omega"],
          alpha = msft_analysis$skew_t$fit@param$dp["alpha"],
          nu = msft_analysis$skew_t$fit@param$dp["nu"]),
      add = TRUE, col = "red", lwd = 1)
```



Build Time Series Models AR(1)-GARCH(1,1)

We can now fit an AR(1)-GARCH(1,1) model to each series of log returns. In R we specify a standard GARCH model and set the GARCH order to (1,1), meaning the variance equation will have one lag of the past squared return (ARCH term) and one lag of past variance (GARCH term). Next, we specify an AR(1) model in the mean equation, which uses one lag or returns. For the distribution model, we will use t-distribution for the errors allowing for fat tails in the return distribution. The GARCH model specification is now complete and we can now fit this model to both SPXL and MSFT returns. These models in our code are referred to as spxl_garch and msft_garch and we can call these variables in order to get a summary of our models. The summary for each of our models is very long, so we will not cover everything that is in the output.

In this section, we will just go over the model parameters for SPXL and MSFT. The mu parameter represents the average daily returns for both stocks and here we can see that these values are positive, with SPXL having a higher value. The ar1 parameter measures the relationship between today's return and yesterday's return. If this value is negative ,as is the case here for both SPXL and MSFT, it means that a positive return yesterday would result in a higher likelihood of having a negative return today and vice versa. Based on the ar1 values, there seems to be a greater tendency for MSFT to reverse the previous day's movement compared to SPXL. The omega parameter represents the base level of variance or volatility. SPXL and MSFT have very small omega values, but MSFT does have a lower baseline volatility than SPXL. The alpha1 parameter measures how much yesterday's return shock affects today's volatility. The higher alpha1 value in SPXL suggests that SPXL's returns are more sensitive to daily shocks than MSFT's returns. Finally, the beta1 parameter measures how much of today's volatility depends on yesterday's volatility. A high beta1 value

close to 1 suggests that volatility is persistent. MSFT has a higher beta1 value which indicates that MSFT has more persistent volatility than SPXL.

```
# Define AR(1)-GARCH(1,1) specification
garch_spec <- ugarchspec(
  variance.model = list(model = "sGARCH", garchOrder = c(1,1)),
  mean.model = list(armaOrder = c(1,0)),
  distribution.model = "std"
)

# Fit the model to the returns
spxl_garch <- ugarchfit(spec = garch_spec, data = spxl_returns)
msft_garch <- ugarchfit(spec = garch_spec, data = msft_returns)

spxl_garch
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,0)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.002681    0.000371   7.2202 0.000000
## ar1    -0.035755    0.020678  -1.7291 0.083784
## omega   0.000018    0.000005   3.7489 0.000178
## alpha1  0.178400    0.023035   7.7446 0.000000
## beta1   0.820599    0.020501  40.0268 0.000000
## shape   5.477947    0.586727   9.3364 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.002681    0.000326   8.2373 0.000000
## ar1    -0.035755    0.017711  -2.0188 0.043506
## omega   0.000018    0.000006   3.2651 0.001094
## alpha1  0.178400    0.026704   6.6807 0.000000
## beta1   0.820599    0.023607  34.7604 0.000000
## shape   5.477947    0.601520   9.1068 0.000000
##
## LogLikelihood : 5689.484
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.5251
## Bayes        -4.5111
## Shibata      -4.5251
```

```
## Hannan-Quinn -4.5200
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      1.567  0.2106
## Lag[2*(p+q)+(p+q)-1][2]     1.572  0.3981
## Lag[4*(p+q)+(p+q)-1][5]     2.191  0.6560
## d.o.f=1
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.8287  0.3626
## Lag[2*(p+q)+(p+q)-1][5]    1.5888  0.7181
## Lag[4*(p+q)+(p+q)-1][9]    3.8027  0.6224
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]  0.006482 0.500 2.000  0.9358
## ARCH Lag[5]  1.574545 1.440 1.667  0.5729
## ARCH Lag[7]  3.258643 2.315 1.543  0.4661
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  2.9498
## Individual Statistics:
## mu     0.3975
## ar1    0.3504
## omega  0.8495
## alpha1 0.5119
## beta1  0.9639
## shape  1.1483
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.49 1.68 2.12
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value      prob sig
## Sign Bias           3.3917 0.0007053 ***
## Negative Sign Bias  0.6960 0.4865026
## Positive Sign Bias  0.1831 0.8547046
## Joint Effect       18.7504 0.0003079 ***
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     88.65    5.742e-11
```

```
## 2    30    117.75    1.179e-12
## 3    40    123.57    1.013e-10
## 4    50    136.25    3.760e-10
##
##
## Elapsed time : 0.2942429
```

```
msft_garch
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,0)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.001351    0.000223   6.0560 0.000000
## ar1    -0.071523    0.020024  -3.5719 0.000354
## omega   0.000008    0.000007   1.0849 0.277961
## alpha1  0.124083    0.027184   4.5645 0.000005
## beta1   0.860854    0.027935  30.8163 0.000000
## shape   4.464570    0.688657   6.4830 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.001351    0.000250  5.41570 0.000000
## ar1    -0.071523    0.019580 -3.65288 0.000259
## omega   0.000008    0.000037  0.21029 0.833444
## alpha1  0.124083    0.116760  1.06272 0.287910
## beta1   0.860854    0.116875  7.36559 0.000000
## shape   4.464570    3.039389  1.46890 0.141859
##
## LogLikelihood : 7076.104
##
## Information Criteria
## ------------------------------------
##
## Akaike       -5.6291
## Bayes        -5.6151
## Shibata      -5.6291
## Hannan-Quinn -5.6240
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.1894  0.6634
## Lag[2*(p+q)+(p+q)-1][2]    1.9879  0.2174
```

```
## Lag[4*(p+q)+(p+q)-1][5]    3.4023  0.3372
## d.o.f=1
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.02859  0.8657
## Lag[2*(p+q)+(p+q)-1][5]   0.63756  0.9345
## Lag[4*(p+q)+(p+q)-1][9]   2.46400  0.8430
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.3992 0.500 2.000  0.5275
## ARCH Lag[5]    1.0449 1.440 1.667  0.7196
## ARCH Lag[7]    2.2351 2.315 1.543  0.6676
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  10.0673
## Individual Statistics:
## mu     0.2811
## ar1    0.2840
## omega  0.4419
## alpha1 1.1176
## beta1  1.3666
## shape  2.2469
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.49 1.68 2.12
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.6236 0.5330
## Negative Sign Bias  0.9430 0.3458
## Positive Sign Bias  1.0495 0.2941
## Joint Effect        2.2266 0.5267
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     42.70     0.001426
## 2    30     51.11     0.006834
## 3    40     61.63     0.011918
## 4    50     82.39     0.001991
##
##
## Elapsed time : 0.365202
```

Finding the Best Copula

The AR(1)-GARCH(1,1) model has been fit to each of the log returns for SPXL and MSFT. We have also compared the model parameters for each stock which has given us important insight into average daily return, volatility, and the overall behavior of the returns. In the following section, we will extract the standardized residuals from both of our time series models and find the best fitting copula in order to understand the true relationship between the two assets. This is a very important step as using standardized residuals allow the copula model to focus on the pure interdependence between two assets without the noise of individual trends and volatility patterns. For financial time series data, returns often show time-varying volatility or heteroskedasticity and autocorrelation. Here our AR-GARCH models address these issues by capturing these trends, serial correlations, and volatility clustering. These standardized residuals are essentially what remains after the model has accounted for these patterns. This will give us a more accurate and meaningful representation of the dependency structure of the two assets.

In our process, the first thing we do is extract the standardized residuals from the AR-GARCH model for SPXL and MSFT. We set the standardize argument in our code to true which will scale the residuals by the model's conditional standard deviation. This will normalize these values to a mean of 0 and variance of 1. This process will allow us to evaluate the dependency between SPXL and MSFT returns directly. In the following steps, we define a function called fit_copulas which will transform our data, fit various copulas, and give us the best copula based on the AIC results. Inside the fit_copulas function we make sure that the residuals are in numeric format which is needed for the subsequent computations. These residuals are then transformed into uniform margins (values between 0 and 1) which is necessary in order to fit the various copulas. Once the transformation is complete, the function will fit various copulas using the maximum likelihood estimation (MLE) method. In our report, we fit four different copulas which includes the t-copula, Gaussian copula, Gumbel copula, and Clayton copula. The AIC for each fitted copula model is then calculated which we will use in order to compare and find the best model. Running our function, we are provided with two different types of output. The first output plainly gives us the best copula and its parameters while the second output gives us the AIC values of all the copula models. From our results, we find that the t-copula was selected as the best copula based on the AIC (-2119.21). Our best copula includes the parameters rho and degrees of freedom (df). The rho parameter measures the correlation between the two assets in the copula model. Here rho = 0.7553, which indicates a strong positive linear association between the residuals of SPXL and MSFT. This reflects a strong positive co-movement of the SPXL and MSFT residuals. The df parameter looks at the heaviness of the tails in t-copula where lower df values lead to heavier tails, which suggests stronger tail dependence. Here df = 5.42, which is relatively low, suggesting a moderate tail dependence. Interpreting the df parameter we can see that the t-copula is able to capture the probability of both stocks experiencing extreme, simultaneous returns.

These results prove that the t-copula is the best fit for modeling the dependency between SPXL and MSFT returns. The positive rho value indicates that SPXL and MSFT tend to move together, and the relatively low degrees of freedom suggest that the stocks are somewhat prone to joint extreme returns. From these results, it is safe to assume that during times of high market volatility SPXL and MSFT may experience large, similar movements.

```
fit_copulas(spxl_resid, msft_resid)
```

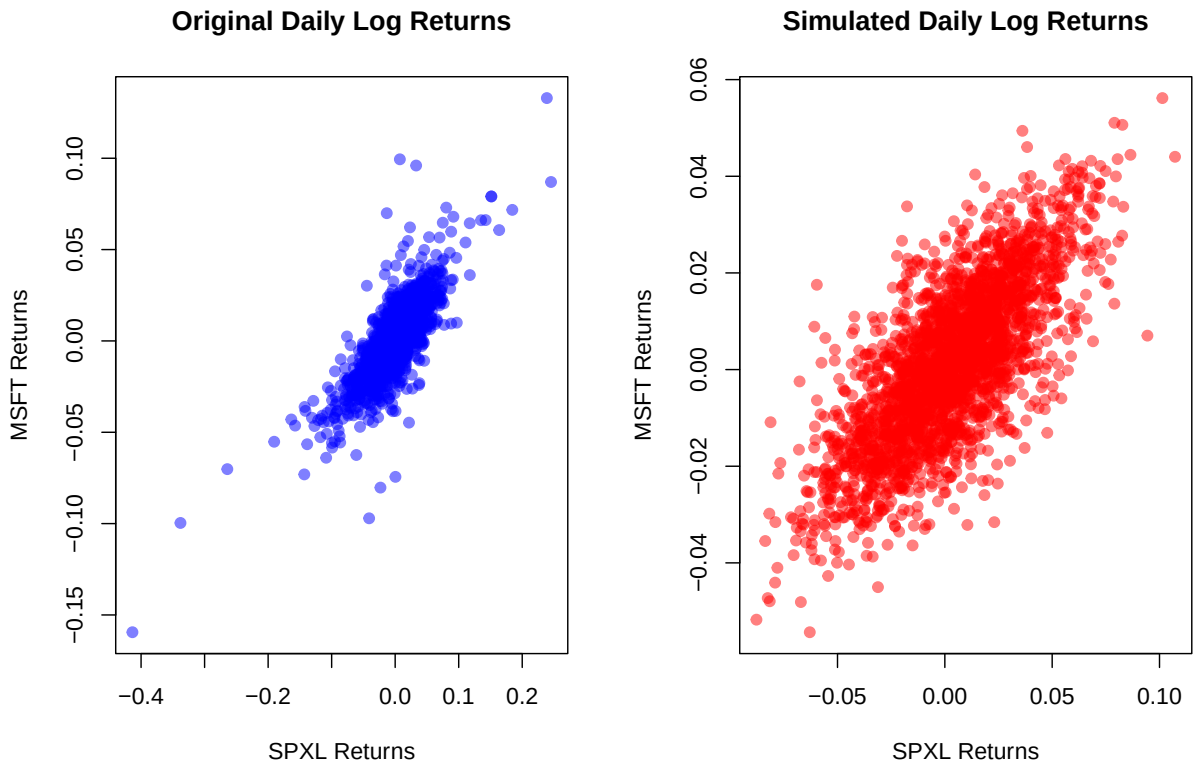
```
## $best_copula
## Call: fitCopula(tCopula(), data = cbind(u1, u2), ... = pairlist(method = "ml"))
## Fit based on "maximum likelihood" and 2512 2-dimensional observations.
## Copula: tCopula
##  rho.1     df
## 0.7553 5.4157
## The maximized loglikelihood is 1062
## Optimization converged
##
## $aic_values
##           t  gaussian    gumbel   clayton
## 1 -2119.289 -1983.927 -1842.464 -1594.397
```

Simulated vs. Original Daily Log Returns

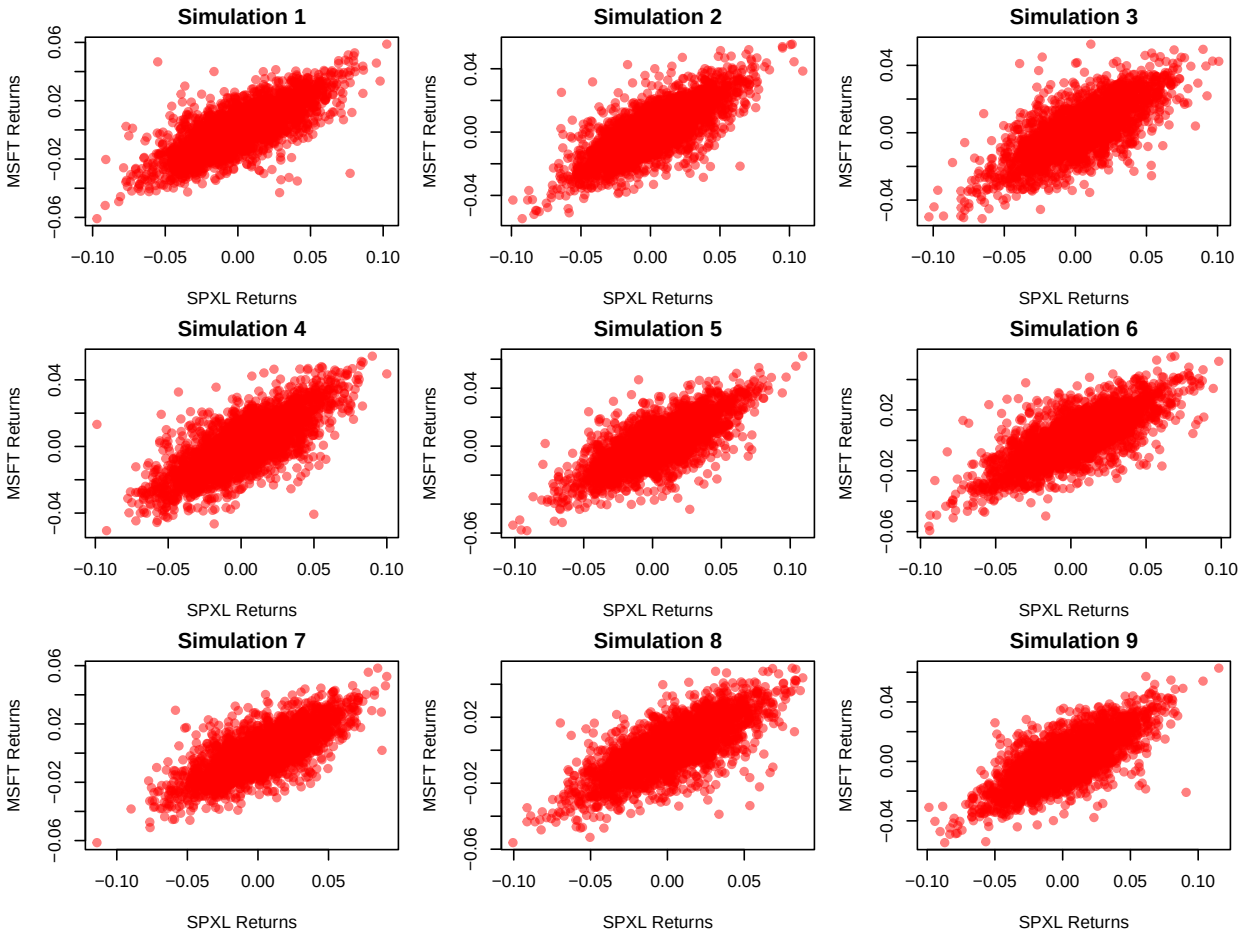
Before we move onto our residual analysis, it is crucial that we assess and validate the accuracy of our fitted models. In order to do this, we are going to define a function in our code called simulate_returns. This will generate a simulated random sample of our daily log returns based on the fitted AR(1)-GARCH(1,1) models and the best-fit copula. The simulation process begins by extracting dependency structures using our best-fit copula (t-copula), transforming uniform random variables into standardized residuals, and reconstructing our daily log returns using our AR(1)-GARCH(1,1) model parameters. By including the autoregressive terms, mean returns, and conditional volatility, the function creates a random sample of simulated returns that exhibit similar characteristics of our original time series data. This is an important method to use as it allows us to generate an artificial dataset which maintains the complex, non-linear relationship we observed in the original daily log returns. In the next steps, we will first set a random seed so that our results are reproducible and generate the simulated data. We fit our best copula model to the standardized residuals of SPXL and MSFT returns, then use the simulate_returns() function to simulate our returns. Additionally, the plot_comparison() function will be used to visually compare the original and simulated daily log returns. This visual comparison allows us to capture the dependency structure of the two stocks and assess joint extreme movements or tail dependence.

Looking at the visual comparison of the original and simulated returns, we can see that the t-copula was effectively able to capture the dependency structure observed in the original data. The original scatterplot shows a positive correlation between SPXL and MSFT returns. Our simulated data on the right using t-copula, shows the same relationship. One of the main advantages of the t-copula is that it is able to capture tail dependence. In the original data, we observe some extreme outliers most noticeably on the bottom left corner. In our simulated returns, there is a presence of some spread and clustering towards the tails. Although this is not as pronounced, the t-copula has managed to reflect this tail dependence. Lastly, both plots show a very dense region in the center, suggesting that the simulation retains the general volatility clustering we found in our original returns. The AR(1)-GARCH(1,1) model is likely capturing the volatility changes over time effectively. If we run the simulated returns multiple different times, we can see that the simulated scatterplots are successfully capturing the dependency structure between SPXL and MSFT. The various simulations also seem to replicate the original distribution's tail dependence and volatility clustering well. In general, the simulated scatterplots capture the key statistical properties of the original SPXL and MSFT daily log returns well. The visual similarity indicates that the t-copula model provides a good fit for modeling the dependence structure between the two financial time series.

```r
# Generate simulations
set.seed(123) # for reproducibility
n_sim <- nrow(all_returns) # same number of points as original data
copula_fit <- fit_copulas(spxl_resid, msft_resid)
simulated_returns <- simulate_returns(n_sim, spxl_garch, msft_garch,
                                      copula_fit, all_returns)


# Create comparison plots
plot_comparison(all_returns, simulated_returns)
```




```r
# Call the function
create_simulation_scatterplots(
  n_plots = 9,
  original_returns = all_returns,
  spxl_garch = spxl_garch,
  msft_garch = msft_garch,
  copula_fit = copula_fit
)
```



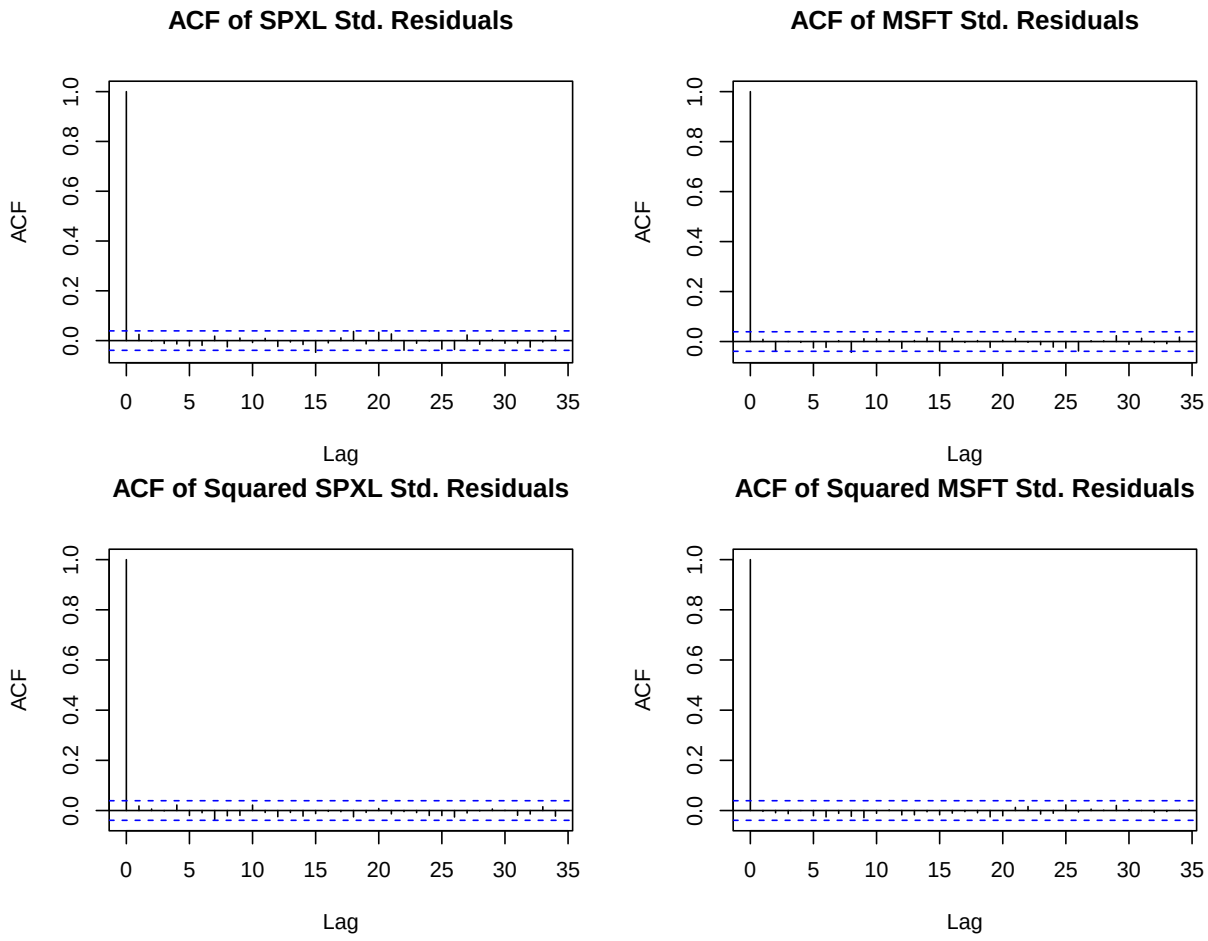
Residual Analysis

In this section, we are going to perform residual analysis in order to see if the AR(1)-GARCH(1,1) model was good enough to catch serial correlation and heteroscedasticity. Based on the results of our analysis, we will either keep the model that we have or specify a new model that better captures these effects. Here, we will use some ACF plots and run statistical tests in order to examine autocorrelation and heteroscedasticity (volatility) in our residuals and squared residuals. The top left and top right plots look at the ACF of the standardized residuals. For both the SPXL and MSFT standardized residuals, we can see that the residuals at various lags are close to zero. The values fall within the confidence bounds or inside the blue dotted lines. This suggests that the residuals have little to no remaining serial correlation. Additionally, the Ljung-Box Test results show that there is no significant serial correlation for the SPXL and MSFT due to the fact that the p-value in each test was significantly greater than 0.05. These high p-values mean that we fail to reject our null hypothesis that there is no autocorrelation in the SPXL and MSFT standardized residuals.

The next thing we need to examine is heteroscedasticity which requires that we first create squared residuals. These squared residuals represent volatility and if there is significant autocorrelation in squared residuals, it means that volatility clustering still persists. If that is the case, then heteroscedasticity still remains. Looking at the ACF plots of the squared residuals, we can see that the ACF is close to zero with no significant spikes. As we did before for the standardized residuals, we can also use the Ljung-Box Test for our squared residuals in order to see if heteroscedasticity still persists. Since the Ljung-Box Test gives us p-values that are much greater than 0.05 for the SPXL and MSFT squared residuals, we fail to reject the null that there is no serial correlation in the squared residuals. Looking at both the ACF plots of the squared residuals and the results from the Ljung-Box Test, it is clear to see that heteroscedasticity no longer remains. In summary, the AR(1)-GARCH(1,1) model was successful in catching serial correlation and heteroscedasticity. The residual

analysis has provided us proof that our model specifications were appropriate, so there is no need to try different models.

```
# Residual Analysis
spxl_resid_squared <- spxl_resid^2
msft_resid_squared <- msft_resid^2

par(mfrow = c(2, 2), mar = c(4, 4, 3, 2) + 0.1)

# ACF of standardized residuals
acf(spxl_resid, main = "ACF of SPXL Std. Residuals")
acf(msft_resid, main = "ACF of MSFT Std. Residuals")
## ACF of squared standardized residuals
acf(spxl_resid_squared, main = "ACF of Squared SPXL Std. Residuals")
acf(msft_resid_squared, main = "ACF of Squared MSFT Std. Residuals")
```




```
# Ljung Box test for serial correlation
cat("\n=== Tests for SPXL ===\n")
```

```
##
## === Tests for SPXL ===
```

```r
cat("\nLjung-Box Test for Serial Correlation in SPXL Std. Residuals:\n")
```

```
##
## Ljung-Box Test for Serial Correlation in SPXL Std. Residuals:
```

```r
print(Box.test(spxl_resid, type = "Ljung-Box", lag = 10))
```

```
##
##  Box-Ljung test
##
## data:  spxl_resid
## X-squared = 7.2873, df = 10, p-value = 0.6981
```

```r
cat("\nLjung-Box Test for Serial Correlation in SPXL Squared Std. Residuals:\n")
```

```
##
## Ljung-Box Test for Serial Correlation in SPXL Squared Std. Residuals:
```

```r
print(Box.test(spxl_resid_squared, type = "Ljung-Box", lag = 10))
```

```
##
##  Box-Ljung test
##
## data:  spxl_resid_squared
## X-squared = 9.8482, df = 10, p-value = 0.4539
```

```r
cat("\n== Tests for MSFT ===\n")
```

```
##
## == Tests for MSFT ===
```

```r
cat("\nLjung Box Test for Serial Correlation in MSFT Std. Residuals:\n")
```

```
##
## Ljung Box Test for Serial Correlation in MSFT Std. Residuals:
```

```r
print(Box.test(msft_resid, type = "Ljung-Box"), lag = 10)
```

```
##
##  Box-Ljung test
##
## data:  msft_resid
## X-squared = 0.18945, df = 1, p-value = 0.6634
```

```r
cat("\nLjung-Box Test for Serial Correlation in MSFT Squared Std. Residuals:\n")
```

```
##
## Ljung-Box Test for Serial Correlation in MSFT Squared Std. Residuals:
```

```r
print(Box.test(msft_resid_squared, type = "Ljung-Box", lag = 10))
```

```
##
##  Box-Ljung test
##
## data:  msft_resid_squared
## X-squared = 7.326, df = 10, p-value = 0.6943
```

Risk Calculation For Conditional Value-at-Risk

The last portion of this report is dedicated to measuring portfolio risk in extreme situations. In the following sections, we will calculate the conditional Value-at-Risk (VaR) and conditional Expected Shortfall (ES) for a portfolio composed of SPXL and MSFT. Calculating the VaR and ES is very useful in portfolios with leveraged stocks like SPXL and in volatile market scenarios. These measures allow us to assess the potential for extreme losses and understand the risk that is associated with holding these assets in a portfolio, especially under situations where we have high volatility. Since we are calculating "conditional" values, these values are calculated based on current information and estimated parameters from our models and copulas.

In our code, we create a function called calculate_risk_measures which will help us calculate the VaR and ES values. The arguments inside our function include our AR-GARCH models for SPXL and MSFT, our best fit copula (t-copula), our various portfolio weights (0.1 to 0.9), and the number of simulations we want to generate from our fitted copula which in this case will be 10000. Inside our function, we extract and retrieve the most recent return values for SPXL and MSFT making sure that they are in numeric format. Next, we extract the AR-GARCH model parameters for SPXL and MSFT and extract the most recent conditional volatility estimates from the models for both stocks. A data frame will be created which will store our results for VaR at the 99% confidence level and ES at the 97.5% confidence level for each portfolio weight. Additionally, we generate random samples (B = 10000) from the best fitted copula model, which will capture the joint dependence between the two stocks. The samples are then transformed to follow a t-distribution which ensures that the simulations match the return distributions observed in SPXL and MSFT. Using the equation form the project outline, we calculate the simulated portfolio returns at each portfolio weight. Finally, we store our VaR and ES values in our designated data frame. The 99% VaR represents the return threshold below which we expect the portfolio returns to fall only 1% of the time. The 97.5% ES represents the average return of the worst 2.5% of portfolio returns.

We will first go over our VaR values at different portfolio weights and then interpret the results. VaR tells us the maximum expected loss at a given confidence level which in this case is 99%. This is essentially a threshold where we expect losses in our portfolio to exceed this level only a small percentage of the time (1%). The 99% confidence level for VaR means that, given current market conditions, there is a 99% probability that the portfolio's loss will not exceed the VaR amount on the next day. We would expect that losses greater than the VaR amount will only occur 1% of the time. We will go through some of these VaR values in our table and interpret these results. When rho = 0.1, the VaR = 0.0725. This means that with 10% of the portfolio in SPXL and 90% in MSFT, the 99% VaR is approximately equal to 7.25%, so there is a 99% probability that the portfolio will not lose more than 7.25% of its value in a single day. If our portfolio is balanced (rho = 0.5) with 50% SPXL and 50% MSFT, there is a 99% probability that the portfolio loss will not exceed 8.90% on the next day. With 90% of the portfolio in SPXL, there is a 99% probability that losses will not exceed 11.40%. As we can see the increase in the VaR values reflects higher potential losses. This pattern shows that increasing our exposure to SPXL leads to an increase in the portfolio's risk profile and potential for extreme losses.

Risk Calculation For Conditional Expected Shortfall

The Expected Shortfall values provide us insight into the severity of extreme losses beyond the VaR threshold. The ES values in our table represent the average of the portfolio returns in the most extreme 2.5% of cases. For a portfolio weighted 10% in SPXL and 90% in MSFT (rho = 0.1), the ES at the 97.5% confidence level is approximately equal to 0.0764 This indicates that, in the worst 2.5% of return outcomes, the average return (or loss) is expected to be around 7.64%. If the portfolio is equally weighted between SPXL and MSFT (rho = 0.5), the ES value suggests that in the worst 2.5% of outcomes, the average return (or loss) is around 9.30%. Lastly, if SPXL makes up 90% of the portfolio weight (rho = 0.9) then in the worst 2.5% of outcomes the average return (or loss) is around 11.69%. For both calculations, we can see that as rho increases both VaR and ES tend to increase as well. Again, a higher weight in SPXL increases portfolio risk exposure. This suggests that a higher concentration in SPXL results in greater potential for extreme downside risks, as our ES values reflect these intensified losses in the tail.

```
# Calculate risk measures
risk_results <- calculate_risk_measures(spxl_garch, msft_garch,
                                        fit_copulas(spxl_resid,
                                                    msft_resid)$best_copula)
risk_results
```

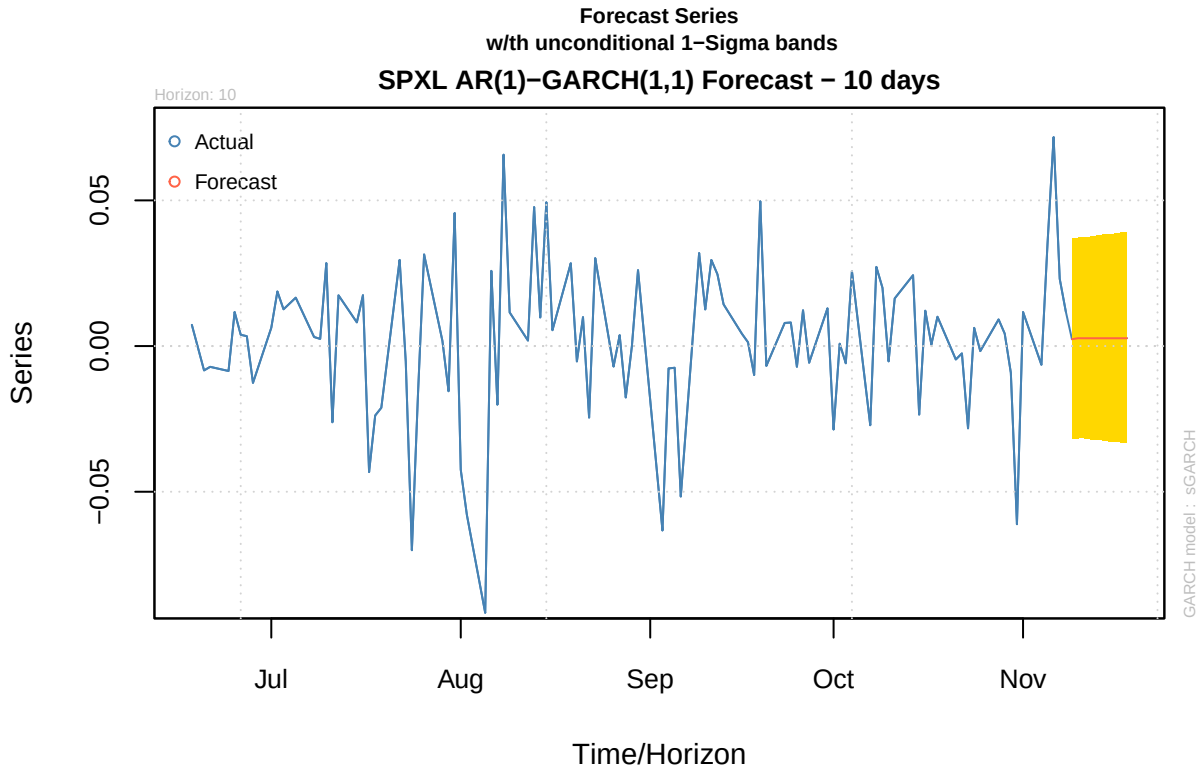
```
##   rho     VaR_99     ES_975
## 1 0.1 0.07246025 0.07640172
## 2 0.2 0.07618902 0.07963989
## 3 0.3 0.07849513 0.08358765
## 4 0.4 0.08338405 0.08806549
## 5 0.5 0.08905239 0.09304744
## 6 0.6 0.09585823 0.09852614
## 7 0.7 0.10208518 0.10436593
## 8 0.8 0.10817986 0.11044756
## 9 0.9 0.11396446 0.11686286
```

Forecasting Returns

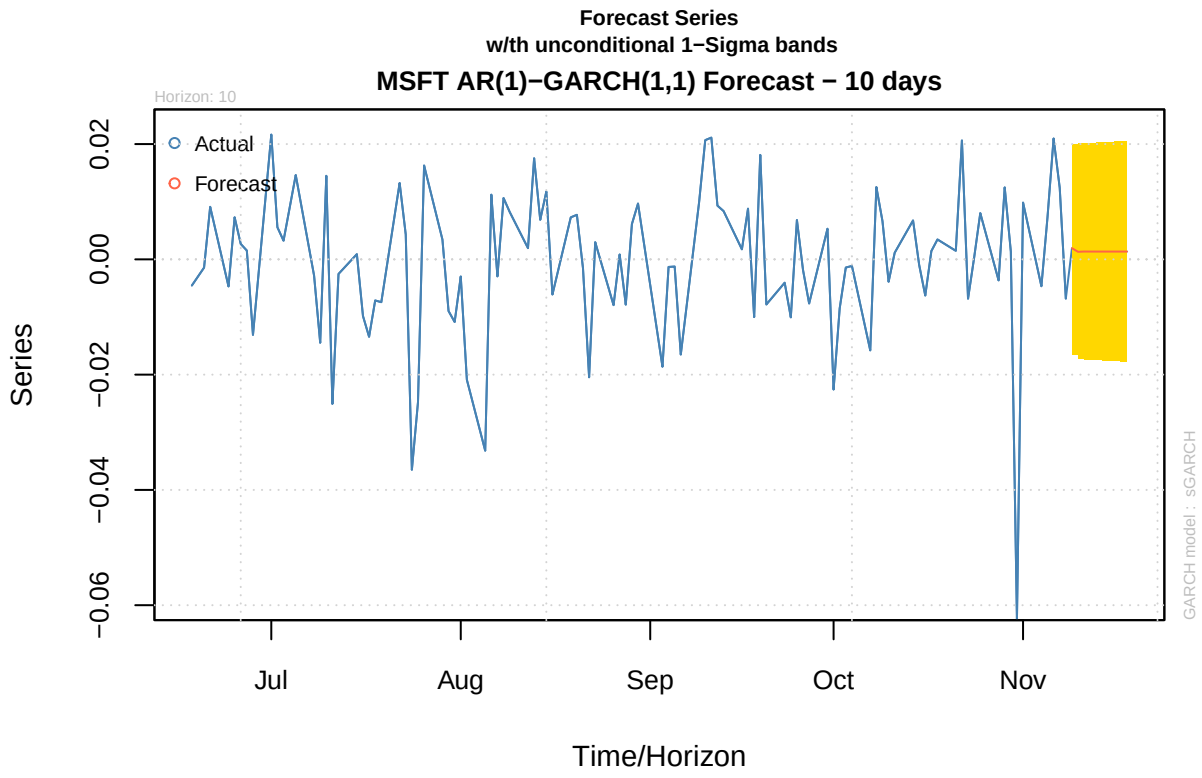
With additional analysis, we can forecast SPXL and MSFT daily log returns using our AR(1)-GARCH(1,1) model. For our model we will set our forecast horizon to 10, which means that we want to create a 10-day forecast for the SPXL and MSFT log returns. Using the ugrachforecast() function we are able to create forecasts for SPXL and MSFT based on the AR(1)-GARCH(1,1) model. Once we have created our forecast, we are going to plot and print these forecast results.

In our SPXL and MSFT forecast plots, the actual data (historical log returns) represents the blue line whereas our forecasted mean returns is represented by the red line. The yellow shaded portion of the plots represents the 1-sigma (standard deviation) prediction interval where a wider interval suggests higher uncertainty in the forecast. First, looking at the SPXL plot and the mean forecasted values we can see that there is a gradual increase in returns. These returns stabilize around 0.00268 after the second forecast period which suggests that SPXL's expected log returns are somewhat positive in the near future.The forecasted mean values for MSFT are lower than SPXL's but they also show a slight positive trend. The MSFT values seem to stabilize around 0.00135 after the second forecast period. Additionally, our computed sigma values confirm that SPXL's expected volatility is consistently higher than MSFT's over the same 10-day forecast period. For SPXL, the sigma values start at 0.034 and increase to 0.036 by the 10th day whereas for MST the sigma values start at 0.0186 and increase slightly to 0.0191 by the 10th day. This larger conditional standard deviation for SPXL shows that it has higher volatility and uncertainty compared to MSFT.

```r
forecast_periods <- 10

# SPXL Forecasts
# GARCH
spxl_garch_pred <- ugarchforecast(spxl_garch, n.head = forecast_periods)
title_spxl <- sprintf("SPXL AR(1)-GARCH(1,1) Forecast - %d days",
                      forecast_periods)
plot(spxl_garch_pred, which = 1)
title(main = title_spxl, line = 0.5, cex.main = 0.8)
```




```r
# MSFT Forecasts
# GARCH
msft_garch_pred <- ugarchforecast(msft_garch, n.head = forecast_periods)
title_msft <- sprintf("MSFT AR(1)-GARCH(1,1) Forecast - %d days",
                      forecast_periods)
plot(msft_garch_pred, which = 1)
title(main = title_msft, line = 0.5, cex.main = 0.8)
```




```
cat("\nSPXL Forecasts:\n")
```

```
##
## SPXL Forecasts:
```

```
print(data.frame(
  Mean = as.numeric(spxl_garch_pred@forecast$seriesFor),
  Sigma = as.numeric(spxl_garch_pred@forecast$sigmaFor)
))
```

```
##           Mean      Sigma
## 1  0.002354782 0.03404622
## 2  0.002692954 0.03429284
## 3  0.002680863 0.03453746
## 4  0.002681295 0.03478011
## 5  0.002681280 0.03502085
## 6  0.002681280 0.03525970
## 7  0.002681280 0.03549670
## 8  0.002681280 0.03573190
## 9  0.002681280 0.03596533
## 10 0.002681280 0.03619702
```

```
cat("\nMSFT Forecasts:\n")
```

```
##
## MSFT Forecasts:
```

```
print(data.frame(
  Mean = as.numeric(msft_garch_pred@forecast$seriesFor),
  Sigma = as.numeric(msft_garch_pred@forecast$sigmaFor)
))
```

```
##           Mean      Sigma
## 1  0.001935529 0.01857935
## 2  0.001309574 0.01864631
## 3  0.001354344 0.01871203
## 4  0.001351142 0.01877654
## 5  0.001351371 0.01883985
## 6  0.001351354 0.01890201
## 7  0.001351356 0.01896303
## 8  0.001351356 0.01902294
## 9  0.001351356 0.01908177
## 10 0.001351356 0.01913953
```

Comparing Models

In addition to creating forecasts, we have also created two new models which we will use to compare with our original AR(1)-GARCH(1,1) model. Below we have specified and fit our new AR(1)-GARCH(1,2) and AR(2)-GARCH(1,1) models to the returns. Here, garchOrder() specifies the GARCH model we should have, which includes 1 for the lagged conditional variance term (ARCH term) and 2 for the lagged squared residuals term (GARCH term). The armaOrder() keeps the AR(1) component in the mean equation without an MA term. For our AR(2)-GARCH(1,1) model, we use AR(2) with two autoregressive lags and GARCH model with one ARCH term and one GARCH term. Once we have created the models, we can now compare our new models to our original AR(1)-GARCH(1,1) model using AIC. Based on our results, we find that the AR(1)-GARCH(1,1) model has the lowest AIC for SPXL, which suggests that it provides the best fit among the three models for SPXL returns. The AR(2)-GARCH(1,1) model has the lowest AIC for MSFT returns, but the difference is very small compared to our original AR(1)-GARCH(1,1) model. Adding a second AR term seems to marginally improve the model for MSFT returns, but this improvement is not substantial.

```
# Define AR(1)-GARCH(1,2) specification
garch_spec_1_2 <- ugarchspec(
  # Set garchOrder to (1,2)
  variance.model = list(model = "sGARCH", garchOrder = c(1, 2)),
  mean.model = list(armaOrder = c(1, 0)), # AR(1) with no MA component
  distribution.model = "std" # Use the standard Student-t distribution
)

garch_spec_2_1 <- ugarchspec(
  # Set garchOrder to (1,1)
  variance.model = list(model = "sGARCH", garchOrder = c(1,1)),
  mean.model = list(armaOrder = c(2,0)), # AR(2) with no MA component
  distribution.model = "std"
)


# Fit the AR(1)-GARCH(1,2) model to the returns
spxl_garch_1_2 <- ugarchfit(spec = garch_spec_1_2, data = spxl_returns)
```

```r
msft_garch_1_2 <- ugarchfit(spec = garch_spec_1_2, data = msft_returns)

# Fit the AR(2)-GARCH(1,1) model to the returns
spxl_garch_2_1 <- ugarchfit(spec = garch_spec_2_1, data = spxl_returns)
msft_garch_2_1 <- ugarchfit(spec = garch_spec_2_1, data = msft_returns)

# Extract AIC from models
spxl_garch_aic <- infocriteria(spxl_garch)["Akaike", 1]
msft_garch_aic <- infocriteria(msft_garch)["Akaike", 1]
spxl_garch12_aic <- infocriteria(spxl_garch_1_2)["Akaike", 1]
msft_garch12_aic <- infocriteria(msft_garch_1_2)["Akaike", 1]
spxl_garch21_aic <- infocriteria(spxl_garch_2_1)["Akaike", 1]
msft_garch21_aic <- infocriteria(msft_garch_2_1)["Akaike", 1]

# Print AIC values
paste("SPXL AR(1)-GARCH(1,1) AIC:" , spxl_garch_aic)
```

```
## [1] "SPXL AR(1)-GARCH(1,1) AIC: -4.52506672511978"
```

```r
paste("MSFT AR(1)-GARCH(1,1) AIC: ", msft_garch_aic)
```

```
## [1] "MSFT AR(1)-GARCH(1,1) AIC:  -5.62906359221581"
```

```r
paste("SPXL AR(1)-GARCH(1,2) AIC: ", spxl_garch12_aic)
```

```
## [1] "SPXL AR(1)-GARCH(1,2) AIC:  -4.52395628409501"
```

```r
paste("MSFT AR(1)-GARCH(1,2) AIC: ", msft_garch12_aic)
```

```
## [1] "MSFT AR(1)-GARCH(1,2) AIC:  -5.62835498767831"
```

```r
paste("SPXL AR(2)-GARCH(1,1) AIC:", spxl_garch21_aic)
```

```
## [1] "SPXL AR(2)-GARCH(1,1) AIC: -4.52427600118173"
```

```r
paste("MSFT AR(2)-GARCH(1,1) AIC:", msft_garch21_aic)
```

```
## [1] "MSFT AR(2)-GARCH(1,1) AIC: -5.62941676168054"
```

Conclusion

We have successfully analyzed the daily log returns of SPXL and MSFT. Throughout our analysis, we were able to find some important and meaningful insights. Exploring our data we found that both assets have positive average daily returns, with SPXL exhibiting higher volatility. SPXL's returns were more negatively skewed (-1.39) and have significantly heavier tails (kurtosis: 18.81), which suggests greater risk of extreme negative events compared to MSFT. Using our Ljung-Box test and ACF plots we were able to confirm the presence of serial correlation for both stocks. The persistent autocorrelation in the squared returns' ACF was evidence that volatility clustering was present, which indicates heteroskedasticity and supports our case of using a GARCH model. Our cross-sectional dependence analysis showed a positive relationship between SPXL and MSFT, but with certain indications of non-linearity. The Spearman and Pearson correlation measures indicated that we have to account for non-linear dependence. This was further justified when performing tail dependence analysis where our lower tail dependence coefficient was 61.31%. Just from exploring our data alone, we were able to validate the use of an AR(1)-GARCH(1,1) model in order to capture serial correlation and volatility clustering. Furthermore, our cross-sectional analysis provided us with evidence that we need to use copulas in order to model non-linear relationships and account for tail dependence to make a thorough analysis of our stock returns.

The statistical modeling process revealed other important findings as well. First of all, the skew t-distribution ended up being the best fit for both SPXL and MSFT returns. The distribution was able to capture their leptokurtic nature and asymmetric risk profiles. The choice of our optimal distribution indicates that the stocks have a tendency toward extreme movements and negative skewness. The AR(1)-GARCH(1,1) model parameters showed that MSFT has a stronger tendency to reverse previous day's movements, SPXL shows higher sensitivity to daily shocks, and MSFT demonstrates more persistent volatility patterns. The t-copula provided the best fit for modeling the dependency structure between SPXL and MSFT returns.The t-copula's rho parameter (0.7533) indicated a strong positive correlation between SPXL and MSFT, while the degrees of freedom parameter (5.42) suggested moderate tail dependence. By analyzing our best fit copula, we can conclude that during periods of market stress, SPXL and MSFT seem to experience simultaneous, significant movements. Additionally, visual comparisons between the original and simulated data showed that the t-copula was effectively capturing the dependency between SPXL and MSFT. While the simulated data's extreme tail behavior was less pronounced, it still indicated the tendency for joint extreme movements, especially in the negative tail. Both the original and simulated returns show dense central regions, which means that we were successfully able to capture the volatility clustering that was present in our original data. Repeated simulations were performed, which reinforced the reliability of our AR(1)-GARCH(1,1) model combined with the t-copula in modeling the relationship between SPXL and MSFT.

The residual analysis was able to confirm that the AR(1)-GARCH(1,1) model effectively captured serial correlation and volatility (heteroscedasticity). The ACF plots showed that residuals at various lags were close to zero, with values being within the confidence bounds, suggesting little to no serial correlation. Furthermore, the Ljung-Box Test results showed p-values that were significantly greater than 0.05 for both SPXL and MSFT, confirming that any serial correlation in our residuals was insignificant. The ACF plots of the squared residuals including the Ljung-Box Test also confirmed that heteroscedasticity or volatility no longer persists. Overall, the residual analysis demonstrated that the AR(1)-GARCH(1,1) model was sufficient in capturing these key features.

Finally, the risk analysis that we performed through VaR and ES illustrated that increasing portfolio allocation to SPXL ended up increasing our risk exposure. At the 99% confidence level, 10% of the portfolio in SPXL and 90% of the portfolio in MSFT showed a VaR value of 7.25%. A balanced portfolio where SPXL and MSFT have equal weights showed a VaR value of 8.90%. If 90% of the portfolio was in SPXL and 10% of the portfolio was in MSFT, it ended up showing a VaR value of 11.39%.

Once we were done with our main analysis, we moved onto creating a 10 day forecast for SPXL and MSFT returns using our AR(1)-GARCH(1,1) model. SPXL's forecasted mean returns had stabilized around 0.00268, while MSFT's forecasted mean returns stabilized around 0.00135. This indicates that both SPXL and MSFT have a slight positive trend. SPXL showed higher volatility, with sigma values rising from 0.034 to 0.036, compared to MSFT's lower volatility, with sigma values increasing from 0.0186 to 0.0191. In our

additional analysis, we also ended up comparing our original AR(1)-GARCH(1,1) model against our new AR(1)-GARCH(1,2) and AR(2)-GARCH(1,1) models. Comparing the models, we found that the AR(1)-GARCH(1,1) model outperformed our new models for SPXL returns. This suggests that adding a second GARCH lag or adding a second AR term did not significantly improve the fit for SPXL returns. However, the AR(2)-GARCH(1,1) model did have the lowest AIC for MSFT returns compared to our original AR(1)-GARCH(1,1) model. Since this improvement was extremely small for MSFT, it would be best to choose the AR(1)-GARCH(1,1) as it is less complex and performs nearly as well for MSFT returns. The forecast can provide us useful insights for the short-term in order to make certain investment decisions. Expanding our analysis even further, we can create additional GARCH models or use advanced techniques to make better predictions.

An important thing to keep in mind is that SPXL is a leveraged ETF which aims to provided three times more daily returns compared to a traditional index fund like the S&P 500. This essentially means that if the S&P 500 goes up by 1% in one day then SPXL aims to go up 3% and vice versa. This characteristic of the stock is the reason why we see higher volatility and greater tail risk compared to MSFT. This can be detrimental during market downturns as the leveraged nature of SPXL dramatically amplifies our losses. The increasing VaR values provide further proof that adding more of the leveraged ETF (SPXL) into our portfolio increases potential losses.

These findings emphasize that our portfolio needs to be carefully allocated when we are combining SPXL with traditional stocks like MSFT. While both assets seem to show positive returns over time, their risk profiles are quite different. SPXL presents higher potential for extreme losses. Investors should be very wary when determining the portfolio weight of these stocks. This is especially true given the strong positive dependency between the assets during periods of market stress.

Appendix

```r
library(quantmod)
library(rugarch)
library(copula)
library(PerformanceAnalytics)
library(xts)
library(fGarch)
library(sn)
library(fitdistrplus)
library(stats)
library(forecast)


## Specify 10 year period between start and end date
end_date <- as.Date("2024-11-11")
start_date <- end_date - 3650

## Get stock prices
getSymbols(c("SPXL", "MSFT"), from = start_date, to = end_date)
spxl_prices <- Ad(SPXL)
msft_prices <- Ad(MSFT)

# Get log returns
spxl_returns <- diff(log(spxl_prices))[-1]
msft_returns <- diff(log(msft_prices))[-1]

# Align dates and create data frame for both returns
all_returns <- merge(spxl_returns, msft_returns, join = "inner")
colnames(all_returns) <- c("SPXL", "MSFT")

# Define function to explore dataset
explore_data <- function(returns, name) {

  # Basic Statistic Summary
  cat("\n Analysis for", name, "\n")
  cat("Mean: ", mean(returns), "\n")
  cat("Std Dev: ", sd(returns), "\n")
  cat("Skewness: ", skewness(returns), "\n")
  cat("Kurtosis: ", kurtosis(returns), "\n")

  # Ljung-Box Test
  cat("\nLjung-Box Test for Serial Correlation:\n")
  print(Box.test(returns, type = "Ljung-Box"))

  cat("\nLjung-Box Test for ARCH effects (squared returns):\n")
  print(Box.test(returns^2, type = "Ljung-Box"))

  # Plots
  par(mfrow = c(2,2))
  plot.ts(returns, main = paste(name, "Returns"))
  qqnorm(returns, main = paste("Q-Q Plot of", name))
  qqline(returns)
  acf(returns, main = paste("ACF of", name))
```

```r
  acf(returns^2, main = paste("ACF of Squared", name))
  par(mfrow = c(1,1))
}

explore_dependence <- function(returns1, returns2, name1, name2) {

  returns1 <- as.numeric(returns1)
  returns2 <- as.numeric(returns2)

  plot(returns1, returns2,
       main = paste("Scatter Plot of", name1, "vs.", name2, "Returns"),
       xlab = name1, ylab = name2, pch = 19, col = "royalblue")

  cat("\nPearson Correlation:", cor(returns1, returns2), "\n")
  cat("Spearman Correlation:", cor(returns1, returns2,
                                   method = "spearman"), "\n")

  lower_tail <- sum(returns1 < quantile(returns1, 0.05) &
                      returns2 < quantile(returns2, 0.05)) / length(returns1)
  upper_tail <- sum(returns1 > quantile(returns1, 0.95) &
                      returns2 > quantile(returns2, 0.95)) / length(returns1)

  cat("\nEmpirical lower tail dependence coefficient:", lower_tail * 20, "\n")
  cat("Empirical upper tail dependence coefficient:", upper_tail * 20, "\n")
}

# Function to calculate AIC for normal distribution
fit_normal <- function(data) {
  fit <- fitdistr(data, "normal")
  loglik <- sum(dnorm(data, fit$estimate[1], fit$estimate[2], log = TRUE))
  aic <- -2 * loglik + 2 * 2  # 2 parameters: mean and sd
  return(list(aic = aic, fit = fit))
}

# Function to calculate AIC for Student's t distribution
fit_t <- function(data) {
  fit <- fitdistr(data, "t")
  loglik <- sum(dt((data - fit$estimate[1])/fit$estimate[2],
                   df = fit$estimate[3], log = TRUE) - log(fit$estimate[2]))
  aic <- -2 * loglik + 2 * 3  # 3 parameters: location, scale, df
  return(list(aic = aic, fit = fit))
}

# Function to calculate AIC for skew normal distribution
fit_skew_normal <- function(data) {
  fit <- selm(data ~ 1)
  aic <- AIC(fit)
  return(list(aic = aic, fit = fit))
}

# Function to calculate AIC for skew t distribution
fit_skew_t <- function(data) {
  fit <- selm(data ~ 1, family = "ST")
```

```
  aic <- AIC(fit)
  return(list(aic = aic, fit = fit))
}

# Function to fit all distributions and compare
compare_distributions <- function(data, stock_name) {
  # Fit all distributions
  normal_fit <- fit_normal(data)
  t_fit <- fit_t(data)
  skew_normal_fit <- fit_skew_normal(data)
  skew_t_fit <- fit_skew_t(data)

  # Create results data frame
  results <- data.frame(
    Distribution = c("Normal", "Student's t", "Skew Normal", "Skew t"),
    AIC = c(normal_fit$aic, t_fit$aic, skew_normal_fit$aic, skew_t_fit$aic)
  )

  # Sort by AIC
  results <- results[order(results$AIC), ]

  # Print results
  cat("\nResults for", stock_name, ":\n")
  print(results)

  return(list(
    normal = normal_fit,
    t = t_fit,
    skew_normal = skew_normal_fit,
    skew_t = skew_t_fit,
    results = results
  ))
}

# Fit Copula Models

# Extract standardized residuals
spxl_resid <- residuals(spxl_garch, standardize = TRUE)
msft_resid <- residuals(msft_garch, standardize = TRUE)

# Copula Fitting Function
fit_copulas <- function(eps1, eps2) {

  # Transform into numeric values
  eps1 <- as.numeric(eps1)
  eps2 <- as.numeric(eps2)

  # Transform to uniform margins
  u1 <- pobs(eps1)
  u2 <- pobs(eps2)

  # Fit different copulas
  cop_t <- fitCopula(tCopula(), cbind(u1, u2), method = "ml")
```

```
  cop_gaussian <- fitCopula(normalCopula(), cbind(u1, u2), method = "ml")
  cop_gumbel <- fitCopula(gumbelCopula(), cbind(u1, u2), method = "ml")
  cop_clayton <- fitCopula(claytonCopula(), cbind(u1, u2), method = "ml")

  # Compare AIC
  aic_t <- AIC(cop_t)
  aic_gaussian <- AIC(cop_gaussian)
  aic_gumbel <- AIC(cop_gumbel)
  aic_clayton <- AIC(cop_clayton)

  # Return best copula
  aic_values <- c(aic_t, aic_gaussian, aic_gumbel, aic_clayton)
  copulas <- list(cop_t, cop_gaussian, cop_gumbel, cop_clayton)
  best_copula <- copulas[[which.min(aic_values)]]

  return(list(
    best_copula = best_copula,
    aic_values = data.frame(
      t = aic_t,
      gaussian = aic_gaussian,
      gumbel = aic_gumbel,
      clayton = aic_clayton
    )
  ))
}

# Function to simulate returns using fitted models and copula
simulate_returns <- function(n_sim, spxl_garch, msft_garch, copula_fit,
                             original_returns) {
  # Extract the actual copula object from the fit
  fitted_copula <- copula_fit$best_copula@copula

  # Generate dependent uniform variates from the best copula
  sim_uniform <- rCopula(n_sim, fitted_copula)

  # Get the fitted parameters from GARCH models
  spxl_params <- coef(spxl_garch)
  msft_params <- coef(msft_garch)

  # Initialize matrices for simulated standardized residuals
  spxl_std_resid <- qnorm(sim_uniform[,1])
  msft_std_resid <- qnorm(sim_uniform[,2])

  # Get conditional volatility from GARCH models
  spxl_sigma <- sigma(spxl_garch)
  msft_sigma <- sigma(msft_garch)

  # Calculate mean volatility to use for simulation
  spxl_mean_sigma <- mean(spxl_sigma)
  msft_mean_sigma <- mean(msft_sigma)

  # Generate returns using AR(1)-GARCH(1,1) structure
  spxl_sim_returns <- spxl_params["mu"] + spxl_params["ar1"] *
```

```
    mean(original_returns$SPXL) +
    spxl_mean_sigma * spxl_std_resid
  msft_sim_returns <- msft_params["mu"] + msft_params["ar1"] *
    mean(original_returns$MSFT) +
    msft_mean_sigma * msft_std_resid

  # Combine simulated returns
  sim_returns <- data.frame(
    SPXL = spxl_sim_returns,
    MSFT = msft_sim_returns
  )

  return(sim_returns)
}

# Function to create comparison plots
plot_comparison <- function(original_returns, simulated_returns) {
  par(mfrow = c(1,2))

  # Original returns scatter plot
  plot(as.numeric(original_returns$SPXL),
       as.numeric(original_returns$MSFT),
       main = "Original Daily Log Returns",
       xlab = "SPXL Returns",
       ylab = "MSFT Returns",
       pch = 19,
       col = rgb(0,0,1,0.5))

  # Simulated returns scatter plot
  plot(simulated_returns$SPXL,
       simulated_returns$MSFT,
       main = "Simulated Daily Log Returns",
       xlab = "SPXL Returns",
       ylab = "MSFT Returns",
       pch = 19,
       col = rgb(1,0,0,0.5))
}

## Define function to create multiple random sample simulations
create_simulation_scatterplots <- function(n_plots = 9,
                                           original_returns,
                                           spxl_garch,
                                           msft_garch,
                                           copula_fit) {
  # Set up a grid of plots
  par(mfrow = c(3, 3), mar = c(4, 4, 2, 1))

  # Original returns plot first
  plot(original_returns$SPXL, original_returns$MSFT,
       main = "Original Returns",
       xlab = "SPXL Returns",
       ylab = "MSFT Returns",
       pch = 19,
```

```
       col = rgb(0,0,1,0.5))

  # Generate and plot multiple simulations
  for(i in 1:n_plots) {
    sim_returns <- simulate_returns(
      n_sim = nrow(original_returns),
      spxl_garch = spxl_garch,
      msft_garch = msft_garch,
      copula_fit = copula_fit,
      original_returns = original_returns
    )

    plot(sim_returns$SPXL, sim_returns$MSFT,
         main = paste("Simulation", i),
         xlab = "SPXL Returns",
         ylab = "MSFT Returns",
         pch = 19,
         col = rgb(1,0,0,0.5))
  }
}

# Function to calculate VaR and ES for different for different portfolio weights
calculate_risk_measures <- function(garch1, garch2, best_copula,
                                    rho_values = seq(0.1, 0.9, by = 0.1),
                                    B = 10000) {
  # Get last values and parameters
  spxl_last <- as.numeric(tail(spxl_returns, 1))
  msft_last <- as.numeric(tail(msft_returns, 1))

  # Extract parameters from GARCH models
  spxl_params <- coef(garch1)
  msft_params <- coef(garch2)

  # Get next period volatility forecasts
  spxl_sigma <- as.numeric(tail(sigma(garch1), 1))
  msft_sigma <- as.numeric(tail(sigma(garch2), 1))

  # Store results
  results <- data.frame(
    rho = rho_values,
    VaR_99 = NA,
    ES_975 = NA
  )

  # Generate copula samples
  cop_sample <- rCopula(B, best_copula@copula)

  # Transform to t-distribution margins
  spxl_sim <- qt(cop_sample[,1], df = coef(garch1)["shape"])
  msft_sim <- qt(cop_sample[,2], df = coef(garch2)["shape"])

  # Calculate for each portfolio weight
  for(i in seq_along(rho_values)) {
```

```
    rho <- rho_values[i]

    # Generate portfolio returns
    port_returns <- rho * (spxl_params["mu"] + spxl_params["ar1"] * spxl_last +
                             spxl_sigma * as.vector(spxl_sim)) +
      (1-rho) * (msft_params["mu"] + msft_params["ar1"] * msft_last +
                   msft_sigma * as.vector(msft_sim))

    # Calculate VaR and Expected Shortfall
    results$VaR_99[i] <- quantile(port_returns, 0.99)
    results$ES_975[i] <- mean(port_returns[port_returns > quantile(port_returns,
                                                                   0.975)])
  }
  return(results)
}
```