CAPM Practical Assignment

FIN30200

Rory Byrne

21373851

Contents

Introduction

A fundamental idea of modern finance is that an investor needs a financial incentive to take a risk. In other words, the expected return on a risky investment, R, must exceed the return on a safe, or risk-free, investment, Rf. Thus, the expected excess return, $R - Rf$, on a risky investment, like owning stock in a company, should be positive. At first, it might seem like the risk of an asset should be measured by its variance. Much of that risk, however, can be reduced by holding other assets in a "portfolio" (i.e. by diversifying your financial holdings). This means that the right way to measure the risk of an asset is not by its variance but rather by its covariance with the market. The Capital Asset Pricing Model (CAPM) formalises this idea. According to this model, the expected excess return on an asset is proportional to the expected excess return on a portfolio of all available assets ("the market portfolio"). The CAPM says:

$$(R_{it} - R_{ft}) = \alpha j + \beta j(R_{mt} - R_{ft})$$

where R_{it} is the expected return of the asset j, R_{ft} measures the expected of return of a risk-free asset during period t (the Federal Funds Rate, FFR), and R_{mt} is the expected return on the market portfolio during period t (SP500).

According to the CAPM, an asset with a $\beta < 1$ has less risk than the market portfolio and therefore has a lower expected excess return than the market portfolio. Meanwhile, an asset with $\beta > 1$ is riskier than the market portfolio and thus commands a higher expected excess return.

The purpose of this assignment is to use the CAPM to model two cryptocurrencies, Bitcoin (BTC) and Ethereum (ETH). The risk-free rate, S&P500 and cryptocurrency data was sourced from the Federal Reserve of St. Louis with the weekly (ending Sunday) frequency used. Data was sourced for period starting the 01st September 2015 and ending on the 31st August 2024 for all subjects with the exception of Ethereum who's data starts on the 29th May 2016.

A.i Data Gathering and Transformation

After gathering the data for our model we must first complete some transformations on our data before running our regression. I used the below formula to convert weekly returns into weekly log returns for both cryptocurrencies and the S&P 500.

$$R_t = \ln\left(\frac{P_t}{P_{t-1}}\right) * 100$$

There are multiple reasons why we do this with some of them outlined below (Brooks,2014):

Normal Distribution

Logarithmic transformations help to bring a skewed distribution closer to a normal distribution. As CAPM models usually assume returns are normally distributed this is a useful property of logarithms for our analysis.

Homoskedasticity

Homoskedasticity implies that the variance of returns is constant over time. By taking the log of data it helps to rescale the data so that its variance is more constant which is one of the assumptions of classical linear regression we will discuss later. Log returns are more likely to have a constant variance which will be important in our analysis of CAPM.

Additive Property

Logarithms can be used as a method to turn a non-linear multiplicative relationship between variables into a linear additive one.

A.i.i Variable Calculation

Next, we calculated both the independent and dependent variables for our model. Our dependent variables were derived by calculating the excess return of both BTC and ETH by subtracting the risk-free return from their log return for each observation, $(R_{it} - R_{ft})$. We then calculate the independent variable by subtracting the risk-free rate from the log return of the S&P500 for each observation, $(R_{m_t} - R_{ft})$.

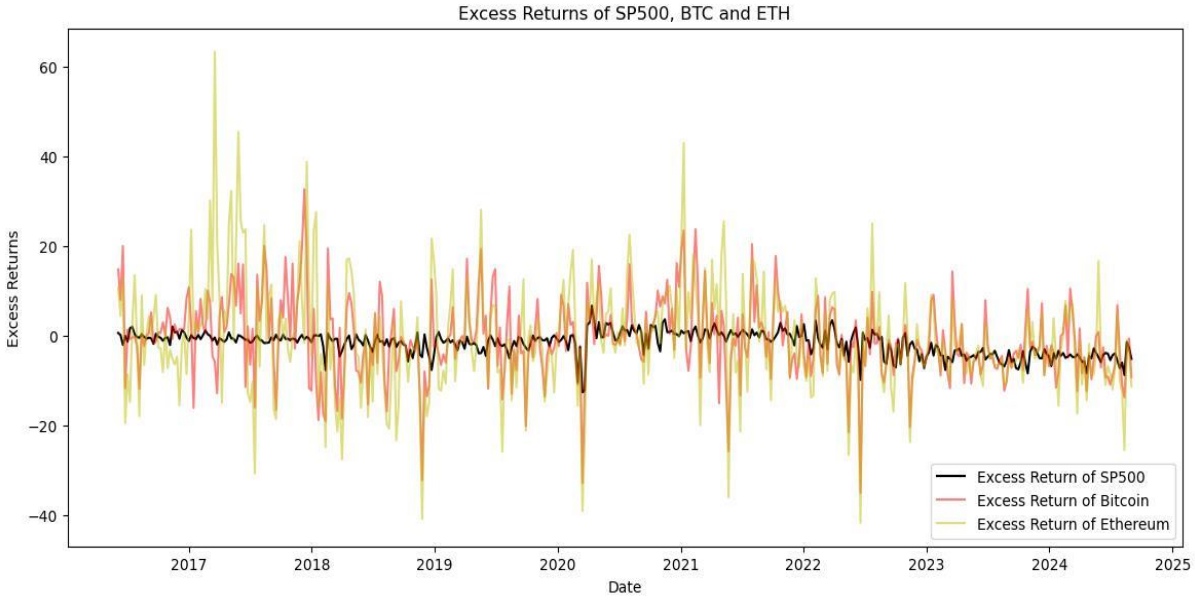


Figure 1: Weekly Excess Returns (S&P500, BTC and ETH)

B. Model CAPM using Linear Regression

Regression analysis is one of the most important tools in econometrics. It is concerned with the relationship between variables and attempts to explain movements in a variable by reference to movements in one or more other variables. Movements in the dependent variable (Y) is what seeks explanation and our independent variables (x1,x2 …, Bn) are used to explain this. More specifically, regression is used to establish if there is a statistically significant relationship between the variables. It assumes a linear relationship between the independent variables and the dependent variables.

The general model is expressed below (Brooks,2014):

$$y_t = \beta_0 + \beta_1 x_{1t} + \beta_2 x_{2t} + \cdots + \beta_i x_{it} + u_t, \; t = 1,2,\ldots.,T$$

Where:

y_t : Dependent Variable

x_{it} : Independent Variables

β_0 : Intercept (Constant)

β_i: Coefficients of Independent Variables

u_t : Error (Residuals)

In the case of our analysis, we will focus on the Classical Linear Regression Model expressed as follows:

$$y_t = \beta_0 + \beta_1 x_t + u_t$$

And in the case of our analysis:

$$(R_{it} - R_{ft}) = \alpha j + \beta j(R_{mt} - R_{ft}) + u_t$$

Where:

y_t : Dependent Variable = $(R_{it} - R_{ft})$

x_t : Independent Variable = $(R_{m_t} - R_{ft})$.

β_0: Intercept (Constant) = αj

β_1: Coefficient of Independent Variable = βj

u_t : Error (Residuals)

In addition to this, there also a number of assumptions underlying classical linear regression (Brooks (2014)):

1. $E(u_t) = 0$: Errors have a mean of 0
2. $Var(u_t) = \sigma^2$: Homoskedasticity - Variance of errors is constant over time
3. $Cov(u_i, u_j) = 0 \; for \; i \neq j$: Errors are linearly independent of each other
4. $Cov(u_t, x_t) = 0$: There is no relationship between errors and the corresponding independent variable
5. $u_t \sim N(0, \sigma^2)$: Errors are normally distributed

B.i Regression Statistics

F-Statistic:

The F-statistic is a key metric in linear regression analysis that evaluates the overall effectiveness of a regression model in explaining variations in the dependent variable. It tests whether the model is statistically significant by comparing the variation explained by the regression model (the "explained variance") to the variation that remains unexplained (the "residual variance"). The F-statistic is calculated as the ratio of these two variances and helps determine if the model fits the data better than a model without predictors.

The null hypothesis (H_0) and alternative hypothesis (H_A) associated with the F – statistic are as follows:

$$H_0: \beta_i = 0$$

Under the null hypothesis all regression coefficients are equal to 0, which implies that a regression model with independent variables is no better at predicting the dependent variable than a model that does not have any predictors

$$H_A: B_i \neq 0$$

Under the alternative hypothesis at least one of the independent variable coefficients does not equal zero, indicating at least one of them significantly predicts the dependent variable.

The F-Statistic is used to compare a restricted model (model with only an intercept and no independent variables as predictors) to an unrestricted model (full regression model with relevant predictors). If the F-statistic is larger than the F-critical value, H_0 is rejected. This implies that the unrestricted model is a better fit than the restricted model. An alternative way of testing this is to find the p-value associated with the F-statistic with a small p-value indicating H_0should be rejected.

The formula to calculate to F-statistic can be seen below:

$$F = \frac{(TSS-RSS)/p}{RSS/(n-p-1)}$$

Where:

- TSS : Total sum of squares: Sum of the squared differences between each observed dependent variable value (y_i) and its mean ($\bar{y}$).
- RSS: Residual sum of squares: Sum of the squared differences between the observed dependent variable values (y_i) and the predicted values from the regression model ($\hat{y}$).
- p : Number of predictors
- n: Number of observations

R-Squared:

R^2 is the most common goodness of fit statistic used in linear regression. It is used to measure how well a model fits the data, i.e. how much of the variability of the dependent variable is explained by the model.

R^2 is obtained by dividing the explained sum of squares, ESS, by the total sum of squares, TSS. TSS captures the total variation across all observations of the dependent variable. This can then be split into two parts, the variance the model can explain (ESS) and the variance not explained by the model (RSS).

To calculate R^2 we use the below formula:

$$R^2 = \frac{ESS}{TSS}$$

which is expressed as:

$$R^2 = \frac{\sum_{i=1}^{n}(y_i-\hat{y})^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2}$$

Where:

- y_i : The observed value of the dependent variable
- $\bar{y}$: The mean of the observed values
- $\hat{y}$: Values of the dependent variable predicted by the model
- n: Number of observations

In addition, given $TSS = ESS + RSS$ we can also obtain the R^2 value as below:

$$R^2 = 1 - \frac{RSS}{TSS}$$

The R^2 value must lie between 0 and 1. A model with an R^2 value close to 1 suggests that the model is a good fit for the data and explains a large amount of the variance in the dependent variable. A value close to 0 indicates a poorly fitting model that does little to explain the variance of the dependent variable

Skewness:

Skewness defines the shape of the distribution and measures the extent to which it is not symmetric about its mean value (Brooks, 2014).

Skewness can be calculated using the below formula:

$$Skewness = \frac{\frac{1}{n-1}\sum(y_i - \bar{y})^3}{\sigma^3}$$

If data is normally distributed we would expect it to have zero skewness.

Kurtosis:

Kurtosis measure how fat the tails of a distribution are and can be defined as:

$$Kurtosis = \frac{\frac{1}{n-1}\sum(y_i - \bar{y})^4}{\sigma^4}$$

A normal distribution should have a kurtosis of 3. When data is distributed with a kurtosis greater than 3 it is said to be leptokurtic. A leptokurtic distribution is characterised by fatter tails and is more peaked at the mean. A platykurtic distribution is less peaked at the mean with thinner tails.

Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC):

AIC and BIC are two commonly used information criteria used in regression analysis. Both AIC and BIC measure the goodness-of-fit of a model and aid with model selection. They are particularly useful when it comes to the choice of how many parameters to include in a model as both these criteria penalise the addition of extra parameters.

AIC penalizes extra parameters more softly, leading to models with more parameters, while the BIC introduces a stricter penalty term, typically leading to more parsimonious models.

B.ii Model Results and Interpretation

As we have already calculated our dependent and independent variables, we now move to run our regression in Python.

In addition, we will check the regression coefficients for statistical significance by the testing the below hypotheses:

$$H_0: \beta_i = 0$$

$$H_A: B_i \neq 0$$

See summary of regression results below (For the full regression output see the appendix section of this paper):

Model	BTC	ETH
Intercept	1.3006	2.3827
SE (intercept)	0.465	0.636
SP500 Coefficient	1.2409	1.8107
SE (SP500)	0.149	0.204
p-value (SP500)	0	0
R-squared	0.139	0.155
Adjusted R-squared	0.137	0.153
F-statistic	69.24	78.95
Prob (F-statistic)	1.17e-15	1.77e-17
AIC	3027	3297
BIC	3035	3305
Jarque-Bera	30.75	248.185
Prob (JB)	2.1e-07	1.28e-54
Durbin-Watson	1.465	1.389
Skewness	0.219	0.674
Kurtosis	4.233	6.464

Table 1: Results Summary of Non-Robust Linear Regression Model of BTC and ETH

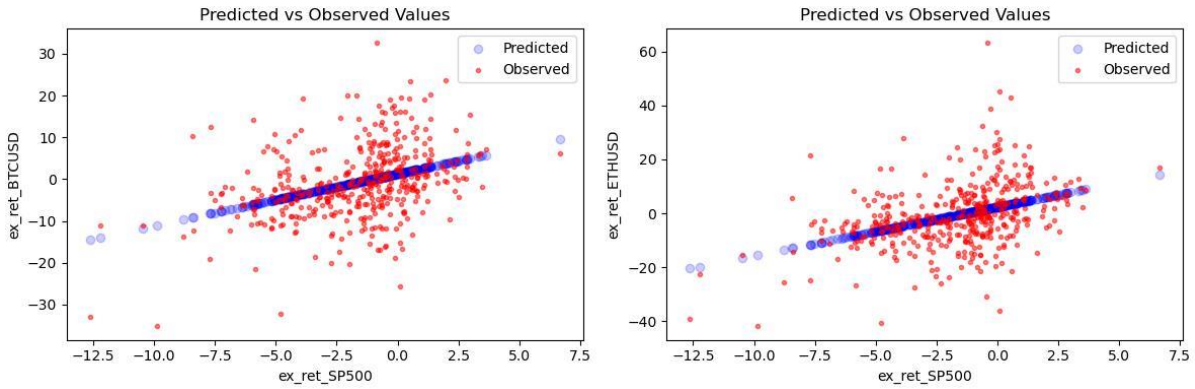


Figure 2: Simple Linear Regression

BTC Model

Using the results from our regression we can express the model as follows:

$$(R_{it} - R_{ft}) = 1.3006 + 1.2409(R_{mt} - R_{ft}) + u_t$$

Regression Coefficients

As seen above, we obtain an intercept of 1.3006. We can interpret this result as the excess return of Bitcoin when the excess return of the S&P 500 (market) is zero. Given the intercept is positive we expect the asset to outperform the market when the return from the market is equal to zero.

The model has also produced a $\hat{\beta}$ coefficient of 1.2409. This coefficient captures the relation between the price of Bitcoin and movements in the wider market. We can interpret this result as for every percentage change in the excess return of the market, the excess return of Bitcoin would be expected to experience a corresponding change in the same direction of 1.2409 percentage points. With the $\hat{\beta}$ of our model being above 1 we can say that the asset experiences slightly more volatile swings in prices compared to a percentage point move in the market.

F-Statistic

The model has an F-statistic of 69.24. In the case of this analysis since we only have one independent variable, we are testing the null hypothesis that our slope coefficient is equal to zero. Our F- statistic corresponds to a p-value of 1.17e-15 which is very small and thus we reject the null hypothesis. This indicates that our coefficient does hold significance in explaining the variability of the dependent variable

R^2

Our BTC model produced an R^2 value of 0.139. This implies that our model is not a good fit and does not explain the variation in the excess return of Bitcoin particularly well with our independent variable only explain 13.9% of this variation.

Statistical Significance

Coefficient	t-value	Crit Val	P>|t|	[0.025	0.975]
$\hat{\alpha}$	2.796	1.96	0.005	0.386	2.215
$\hat{\beta}$	8.321	1.96	0.000	0.948	1.534

Table 2: Summary of results of BTC Model needed for hypothesis testing

$\hat{\alpha}$:

We will first test the intercept coefficient of the model. From our regression we obtain a t-statistic of 2.796. We firstly compare this against the critical value of 1.96 and see that our test statistic is larger than the critical value. Our $\hat{\alpha}$ coefficient also has a p-value close to zero (0.005). Additionally we observe that our 95% confidence interval, [0.386,2.215], does not contain the value zero. Therefore, we reject the null hypothesis and conclude that our intercept coefficient is statistically significant at a 5% significance level.

$\hat{\beta}$:

Moving on, we see that our slope coefficient has a t-value of 8.321, which is larger than 1.96, with a corresponding p-value of 0. We also observe that 0 is not contained in the 95% confidence interval. As a result, we reject the null hypothesis and conclude that our slope coefficient is significantly different than zero at the 5% level of significance.

ETH Model

Using the results from our regression we can express the model as follows:

$$(R_{it} - R_{ft}) = 2.3827 + 1.8107(R_{mt} - R_{ft}) + u_t$$

Regression Coefficients

For the ETH model, we obtain an intercept of 2.3827. We can interpret this result as the log excess return of Ethereum when the log excess return of the S&P 500 (market) is zero. Given the intercept is positive we expect the asset to outperform the market when the return from the market is equal to zero.

The model has also produced a $\hat{\beta}$ coefficient of 1. 8107. This coefficient captures the relation between the price of Ethereum and movements in the wider market. We can interpret this result as for every percentage change in the log excess return of the market, the price of Ethereum would be expected to experience a corresponding change in the same direction of 1.8107 percentage points. With the $\hat{\beta}$ of our model being above 1 we can say that the asset experiences slightly more volatile swings in prices compared to a percentage point move in the market.

F-Statistic

The model has an F-statistic of 78.95. In the case of this analysis since we only have one independent variable, we are testing the null hypothesis that our slope coefficient is equal to zero. Our F- statistic corresponds to a p-value of 1.77e-17 which is very small and thus we reject the null hypothesis. This indicates that our coefficient does hold significance in explaining the variability of the dependent variable

R^2

Our ETH model produced an R^2 value of 0.155. Similar to our Bitcoin model this model is not a good fit and does little to explain the variation in the excess return of Ethereum. While this model is a marginally better fit than the Bitcoin model, the excess return of the market only explains 15.5% of this variation.

Statistical Significance:

Coefficient	t-value	Crit Val	P>|t|	[0.025	0.975]
$\hat{\alpha}$	3.749	1.96	0.000	1.134	3.632
$\hat{\beta}$	8.886	1.96	0.000	1.410	2.211

Table 3: Summary of results of ETH Model needed for hypothesis testing

$\hat{\alpha}$:

We first test the intercept coefficient of the model. From our regression we obtain a t-statistic of 3.749 When compared to the critical value of 1.96 we see that our test statistic is larger

than the critical value. Our $\hat{\alpha}$ coefficient also has a p-value of zero. Additionally, we observe that our 95% confidence interval, [1.134, 3.632], does not contain zero. Therefore, we reject the null hypothesis and conclude that our intercept coefficient for our Ethereum model is statistically significant at a 5% significance level.

$\widehat{\boldsymbol{\beta}}$:

We see that our slope coefficient has a t-value of 8.886, which is larger than 1.96, with a corresponding p-value of 0. We also observe that 0 is not contained in the 95% confidence interval. As a result, we reject the null hypothesis and conclude that our slope coefficient is significantly different than zero at the 5% level of significance.

Figure 2: Simple Linear Regression

B.iii Analysis of Residuals

To test the validity of our model we will test the residuals for the following:

- Normality of distribution
- Heteroskedasticity
- Autocorrelation

B.iii.i Normality of Distribution

To test whether the residuals of our model are normally distributed we will use the Jarque-Bera test for normality.

The test examines whether the coefficients of skewness and excess kurtosis are jointly zero which would be the case if the distribution is normal

The hypotheses associated with this test are outlined below:

$$H_0: The\ residuals\ are\ normally\ distributed$$

$$H_A: The\ residuals\ are\ not\ normally\ distributed$$

The Jarque-Bera statistic follows a Chi-squared distribution and is calculated as follows:

$$JB = \frac{n}{6}\left(s^2 + \frac{1}{4}(k-3)^2\right)$$

Where:

- JB: Jarque-Bera test statistic
- s: Skewness
- k: Kurtosis
- n: Sample Size

The BTC model produces a JB statistic of 30.75 with an associated p-value of 2.10e-07. As the p-value is very close to 0 we can reject the null hypothesis. The test produces a skewness of 0.219, differing from zero, and a kurtosis of 4.233 which is greater than 3 that is expected if residuals are normally distributed.

Moving on to our ETH model, we obtain a JB statistic of 248.185 with an associated p-value of 1.28e-54. As the p-value is very close to 0 we can reject the null hypothesis. The test produces a skewness of 0.674, differing from zero, and a kurtosis of 6.464 which is greater than 3 that is expected if residuals are normally distributed.

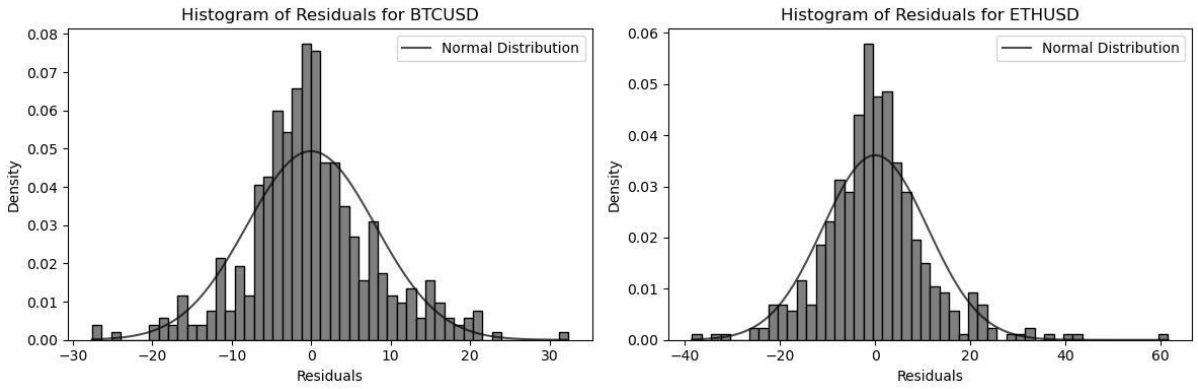


Figure 3: Distribution of Linear Regression Residuals

In addition, from analysing the above figure it is apparent that our residuals are not normally distributed.

Omnibus Test:

The Omnibus Test tests the joint hypothesis that both the skewness and the kurtosis of residuals is zero which is consistent with the assumption of normality. It simultaneously checks for both skewness and kurtosis. The tests statistic is calculates using the below formula:

$$Omibus = n \times \left(\frac{S^2}{6} + \frac{(k-3)^2}{24}\right)$$

The test statistic follows a Chi-squared distribution with 2 degrees of freedom and tests the same hypotheses as the Jarque- Bera test.

Our BTC model reports an Omnibus statistic of 16.66 with an associated p-value of zero while the ETH model has an Omnibus statistic of 68.837 and a corresponding p-value of zero. Thus for both models we can reject H_0 that the residuals of each model are normally distributed.

B.iii.ii Heteroskedasticity

We must also test the homoskedasticity assumption for our regression models. Heteroskedasticity refers to the situation where the variance of residuals is not constant over time.

A potential method of detecting heteroskedasticity is by visually inspecting plots such as Figure 4 below. However, we rarely know the cause or form of heteroskedasticity, and so plots are not a particularly effective method of detection and so we will carry out some further tests.

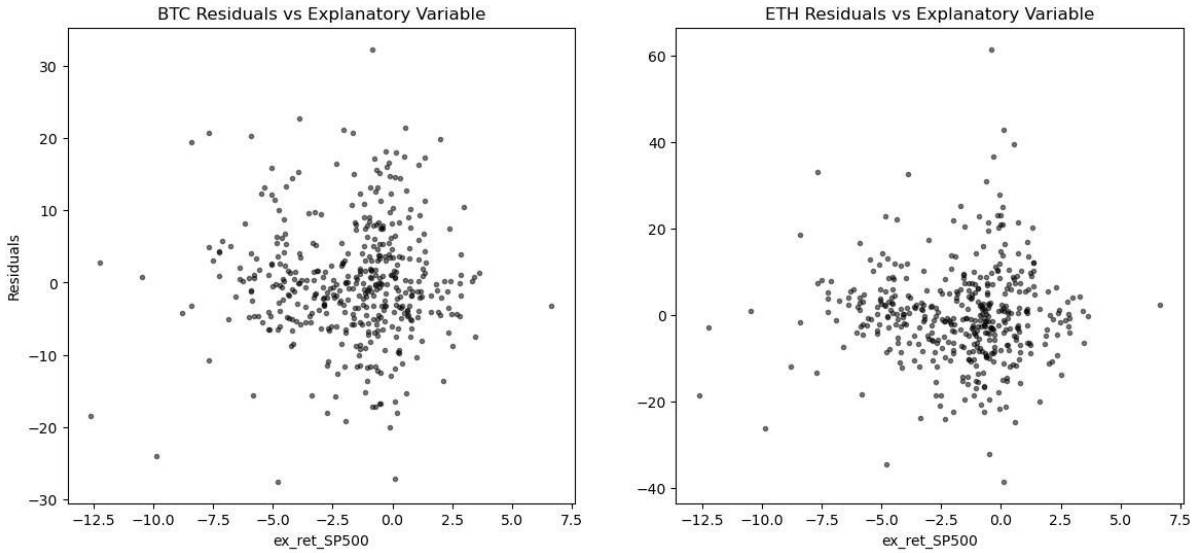


Figure 4: Residuals and Explanatory Variables

White Test:

The White Test examines if there is a significant relationship between the squared residuals of the model and its independent variables and their higher order terms (Wooldridge, J. M. (2016)).

Unlike the Breusch-Pagan test that assumes heteroskedasticity is a linear function of the independent variables, the White test can test for a wider range of heteroskedasticity.

The steps for performing the test are as follows:

1. **Auxiliary Regression:** Once the residuals of the original regression model have been obtained, an auxiliary regression is run. The squared residuals of the original model are regressed onto a set of regressors made up of the original explanatory variables, the squares of the explanatory variables and their cross-products. This auxiliary regression aims to investigate if there is a systematic relationship between the squared residuals and other variables relevant to the model (Brooks,2014)

   The auxiliary regression is seen below:

   $$\hat{u}_t^2 = \gamma_0 + \gamma_1 x_{1t} + \gamma_2 x_{2t} + \gamma_3 x_{3t}^2 + \gamma_4 x_{4t}^2 + \gamma_5 x_{2t} x_{3t} + v_t$$

   Where:

   - $\hat{u}_t^2$: Squared residual at observation t
   - $\gamma_0, \gamma_1, \gamma_2, \dots \gamma_k$: Regression coefficients
   - $x_{1t}, x_{2t}, \dots x_{kt}$: Independent variables, also taking the form of cross-products and squares
   - v_t: Normally distributed error term independent of u_t

2. **Calculate the Lagrange Multiplier Statistic:** The test statistic of the White Test the form of a Lagrange Multiplier (LM) test statistic.

$$LM = TR^2$$

Where:

- T: Number of observations
- R^2: The R^2 value for the auxiliary regression. This represents the amount of variation in the squared residuals explained by the independent variables

The White Test LM statistic follows a Chi – Squared distribution with m degrees of freedom. The degrees of freedom are the amount of regressors in the auxiliary regression with the exception of the constant term.

The hypotheses of the test are as follows:

$$H_0: The\ residuals\ are\ homoskedastic$$

$$H_A: The\ residuals\ are\ heteroskedastic$$

Upon running the test in Python we obtained the following results:

Results of the White Heteroskedasticty test for BTC below:

LM Statistic	2.11392
LM P-Value	0.34751
F Statistic	1.05478
F Statistic P-Value	0.349174

Our BTC model produced an LM statistic of 2.11392 with a corresponding p-value of 0.34751. Therefore, we fail to reject H_0at a 5% level of significance. There is not enough evidence to suggest heteroskedasticity exists in the residuals of the model.

Results of the White Heteroskedasticty test for ETH below:

LM Statistic	0.351172
LM P-Value	0.838965
F Statistic	0.174506
F Statistic P-Value	0.839931

Our ETH model produced an LM statistic of 0.351172 with a corresponding p-value of 0.838965. As a result, we fail to reject H_0at a 5% level of significance. There is not enough evidence to suggest heteroskedasticity exists in the residuals of the model.

Breusch – Pagan Test:

This test examines if the variance in the residuals of the model is related to the independent variables.

The test is carried out in a similar fashion to the White Test however the auxiliary regression for Breusch- Pagan does not use squares or cross products like the White Test does. It only captures linear forms of heteroskedasticity as you can see in the regression function below:

$$\hat{u}_t^2 = \gamma_0 + \gamma_1 x_{1t} + \gamma_2 x_{2t} + \cdots + \gamma_k x_{kt} + v_t$$

Where:

- $\hat{u}_t^2$: Squared residual at observation t
- $\gamma_0, \gamma_1, \gamma_2, \ldots \gamma_k$: Regression coefficients
- $x_{1t}, x_{2t}, \ldots x_{kt}$: Linear independent variables
- v_t: Normally distributed error term independent of u_t

The remainder of the test follows the same steps as the White Test with the same test statistic and hypotheses used.

See below the results of the Breusch-Pagan Test ran on our models:

Results of the Breusch-Pagan Heteroskedasticty test for BTC below:

LM Statistic	0.734752
LM P-Value	0.391347
F Statistic	0.732591
F Statistic P-Value	0.392522

Our BTC model returned an LM Statistic of 0.734752 with a corresponding p-value of 0.391347. Thus, we fail to reject H_0 suggesting there is not enough evidence to suggest heteroskedasticity exists in the residuals of the model.

Results of the Breusch-Pagan Heteroskedasticty test for ETH below:

LM Statistic	0.351153
LM P-Value	0.553461
F Statistic	0.349809
F Statistic P-Value	0.554533

Similarly, for our ETH model, with an LM Statistic of 0.351153 and a corresponding p-value of 0.553461 ,we fail to reject H_0 as there is insufficient evidence to suggest heteroskedasticity exists in the residuals of the model.

Goldfeld-Quandt:

The Goldfeld-Quandt Test is a statistical tool used in regression to detect heteroskedasticity(Goldfeld, S. M., & Quandt, R. E. (1965)). The test splits the data into two subgroups and examines whether or not the variances of the errors in each subset are equal.

The hypotheses for this test are:

$$H_0: The\ variances\ of\ the\ errors\ in\ the\ two\ subsamples\ are\ equal\ (\sigma_1^2 = \sigma_2^2)$$

$$H_A: The\ variances\ of\ the\ errors\ in\ the\ two\ subsamples\ are\ not\ equal\ (\sigma_1^2 \neq \sigma_2^2)$$

The steps for carrying out the test are as follows:

1. Divide n observations into h groups of sizes $n_1 \dots n_h$
2. Choose two groups to test the above hypotheses
3. Run two separate regressions on each of the subgroups
4. We use the below formula to calculate the test statistic:

$$G = \frac{\frac{RSS_L}{n_L - k}}{\frac{RSS_S}{n_S - k}}$$

Where:

- RSS_L: The residual sum of squares for the subgroup with the larger variance
- RSS_S: The residual sum of squares for the subgroup with the smaller variance
- n_L/n_S: The number of observations in each sample
- k: Number of regressors in the regression including the constant

The G test statistic is an F statistic with $(n_L - k)$ and $(n_S - k)$ degrees of freedom. In addition, this test is the only test for heteroskedasticity in this paper that works when we regress without a constant.

See below the results of the Goldfeld-Quandt Test ran on our models:

Results of the Goldfeld-Quandt test for BTC below:

F-statistic	0.633351
p-value	0.999547

Our BTC model returned an F Statistic of 0.633351 with a corresponding p-value of 0.999547. Thus, we fail to reject H_0 suggesting there is not enough evidence to suggest heteroskedasticity exists in the residuals of the model.

Results of the Goldfeld-Quandt test for ETH below:

F-statistic	0.458033
p-value	1

Similarly, for our ETH model, with an F Statistic of 0.458033 and a corresponding p-value of 1 ,we fail to reject H_0 as there is insufficient evidence to suggest heteroskedasticity exists in the residuals of the model.

B.iii.iii Autocorrelation

Autocorrelation, also referred to as serial correlation, occurs when the residuals are not uncorrelated with one another. The autocorrelation function (ACF) can be calculated using the below expression:

$$p_k = \frac{Cov(\hat{u}_t, \hat{u}_{t-k})}{Var(\hat{u}_t)}$$

Where:

- $\hat{u}_t$: Residual at time t
- k: Lag that the ACF is calculated at

It refers to the correlation of a time series to its past values. In the context of regression models, autocorrelation occurs when the residuals from the model are correlated across observations, violating assumption 3 outlined above, that errors are independently distributed. (Wooldridge, J. M. (2010)) This can lead to inefficient estimates and incorrect inferences.

As a result, we must test our models for serial correlation using the below two tests:

- The Durbin-Watson Test
- The Breusch-Godfrey Test

Durbin Watson:

Durbin-Watson is a statistical test for first order autocorrelation, meaning that it tests for a relationship between an error term and its immediate past value(Durbin, J., & Watson, G. S. (1950)).

The test statistic can be obtained using the below formula:

$$DW = \frac{\sum_{i=2}^{n}(\hat{u}_i - \hat{u}_{i-1})^2}{\sum_{i=1}^{n}\hat{u}_i^2}$$

Where:

- $\hat{u}_i$: Residuals at time i
- n: Total observations

The hypotheses for this test are as follows:

$$H_0: p = 0 : Residuals\ are\ not\ autocorrelated$$

$$H_A: p \neq 0 : Residulas\ are\ autocorrelated$$

The DW statistic takes values from 0 to 4 with values closer to 0 indicating positive serial correlation and values closer to 4 indicating negative serial correlation. Values close to 0 indicate that there is not enough evidence to suggest autocorrelation exists between residuals.

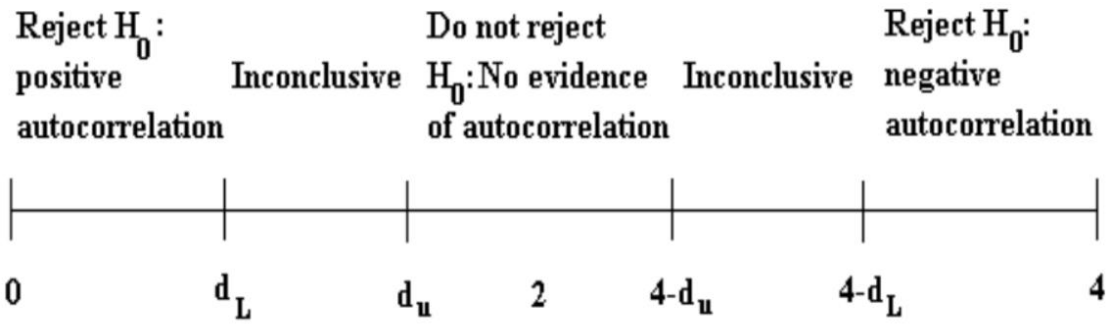


Figure 5: Rejection regions for Durbin – Watson Test

In addition, we can also obtain critical values, d_l and d_u ,for the test using a Durbin Watson table. Given we have 431 observations and 1 independent variable we have the following critical values:

d_l = 1.841 which can be interpreted as the smallest DW statistic that would be expected if no autocorrelation was present

d_u = 1.85 which can be interpreted as the largest DW statistic that would be expected if no autocorrelation was present

There are also a number of conditions that must be satisfied for the DW test to be valid (Brook, 2014):

- The regression must have a constant term
- Regressors must be non-stochastic
- There must be no lags of the dependent variable

If any of these conditions are not fulfilled the test statistic would be biased towards 2 and so H_0 could not be rejected when it should be.

Our BTC model produced a DW statistic of 1.465. Since our test statistic falls below the lower bound of our critical values, we can reject H_0. The test suggests that there is evidence of positive autocorrelation between the residuals in our model.

Our ETC model produced a DW statistic of 1.389 and thus we can draw the same conclusion from the test as with our BTC model.

As the Durbin- Watson test only detects first-order autocorrelation it may fail to identify higher -order autocorrelation effectively. Thus, we will conduct another more general test for autocorrelation, the Breusch – Godfrey Test.

Breusch-Godfrey:

This is a more general test for autocorrelation among residuals as it tests for higher order autocorrelation beyond the first lag up to the r^{th} order (Breusch, T. S. (1978). The test is conducted using the below steps:

1. Fit an auxiliary regression model to regress the residuals ($\hat{u}_t$) on lagged values of the original independent variables and lagged residuals. The auxiliary regression for this given by:

$$\hat{u}_t = \gamma_1 + \gamma_2 x_{2t} + \gamma_3 x_{3t} + \gamma_4 x_{4t} + \rho_1 \hat{u}_{t-1} + \rho_2 \hat{u}_{t-2} + \cdots + \rho_t \hat{u}_{t-r} + v_t$$

Where:

- $\hat{u}_t$: Estimated residual at time t
- $\gamma_1, \gamma_2, \ldots \gamma_k$: Regression coefficients
- $p_1, p_2, \ldots p_t$: Coefficients of lagged independent variables
- v_t : Error term of the auxiliary regression

2. Compute the test statistic. The Lagrange Multiplier (LM) statistic tests whether the lagged residuals and the independent variables are jointly significant in explaining the residuals from our original regression,$\hat{u}_t$. The LM statistic follows a chi-squared distribution with degrees of freedom equal to the number of lagged residuals and independent variables in regression.

The hypotheses for the test are outlined below:

$$H_0: Residuals\ are\ not\ autocorrelated$$

$$H_A: Residulas\ are\ autocorrelated$$

See below the results of the test ran in Python for both models:

Results of the Breusch-Godfrey auto-correlation test for BTC below:

LM Statistic	39.0014
LM P-Value	0.000105176

As we can see above, our BTC model produced an LM statistic of 39.0014 with a corresponding p-value of 0.0001. As a result, we reject H_0 at a level of 5% significance, suggesting that we have evidence of autocorrelation between residuals.

Results of the Breusch-Godfrey auto-correlation test for ETH below:

LM Statistic	51.7608
LM P-Value	6.83491e-07

Our ETH model produced an LM statistic of 51.76 with a corresponding p-value of very close to 0. As a result, we reject H_0 at a level of 5% significance, suggesting that we have evidence of autocorrelation between residuals.

Graphical Autocorrelation Analysis:

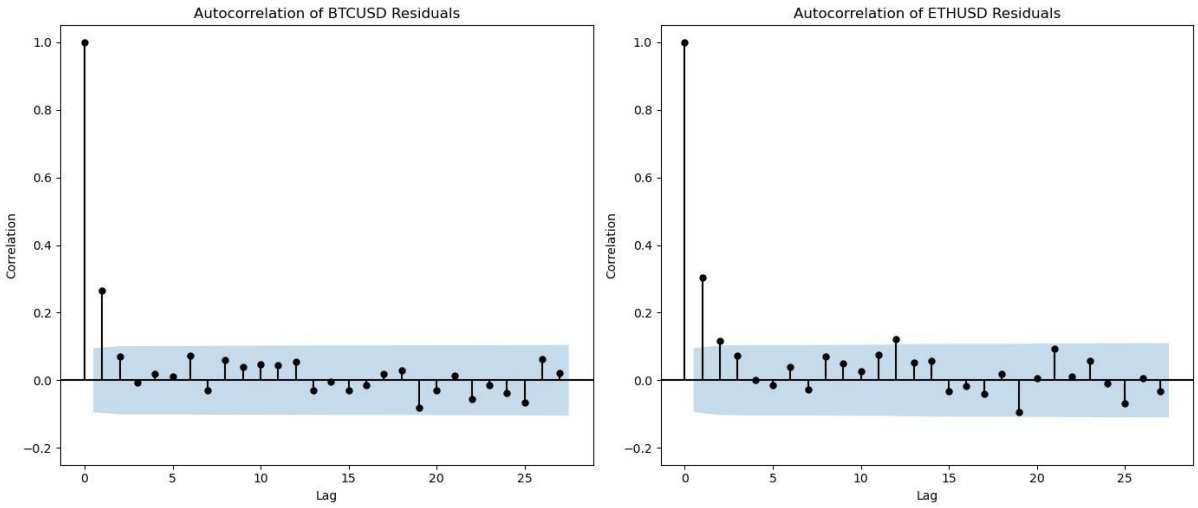


Figure 6: Autocorrelation of Residuals of BTC and ETH models

Additionally, we can visualise the autocorrelation of the residuals using ACF values. The shaded area in Figure 5 illustrates where autocorrelation may potentially be present. Similarly to Durbin-Watson, this only takes first-order autocorrelation into account .

C. Linear Regression using Robust Standard Errors

Upon testing our non-robust regression model in the previous section, we discovered that the residuals of our model showed evidence of autocorrelation. This violates the third assumption of classical linear regression as outlined in section B. We also found that residuals for both models were not normally distributed. This can lead to inefficient estimators, biased standard errors and incorrect inferences.

We will now use robust standard errors in our regression to address these issues. While there was no evidence of heteroskedasticity in the residuals of our models, we will take a conservative approach and use the Newey- West estimator which is commonly used to overcome the challenges of autocorrelation and heteroskedasticity in regression analysis (Newey & West, 1987).

The Newey-West estimator can be expressed as:

$$Var(\hat{\beta}) = (X'X)^{-1}X'\Omega(X'X)^{-1}$$

Where:

- $\hat{\beta}$: Vector of estimated coefficients
- Ω: Newey-West estimator matrix
- X : Regressor matrix

The Newey – West estimator is an effective method to correct for the problems of autocorrelation and heteroskedasticity. Despite the fact that we know autocorrelation exists in the residuals of our model, this method allows us to confidently take inference from our tests.

C.i Robust Regression Results

Robust Model	BTC	ETH
Intercept	1.3006	2.3827
SE (intercept)	0.621	0.939
SP500 Coefficient	1.2409	1.8107
SE (SP500)	0.18	0.246
p-value (SP500)	0	0
R-squared	0.139	0.155
Adjusted R-squared	0.137	0.153
F-statistic	47.48	54.01
Prob (F-statistic)	1.99e-11	1.02e-12
AIC	3027	3297
BIC	3035	3305
Jarque-Bera	30.75	248.185
Prob (JB)	2.1e-07	1.28e-54
Durbin-Watson	1.465	1.389
Skewness	0.219	0.674
Kurtosis	4.233	6.464

Table 4: Regression Results Summary for BTC and ETH using Robust Standard Errors

C.ii. Model Interpretation:

We observe that the intercept and slope coefficients as well as the R^2 value of our models has remained unchanged which is evident from Figure 7 below.

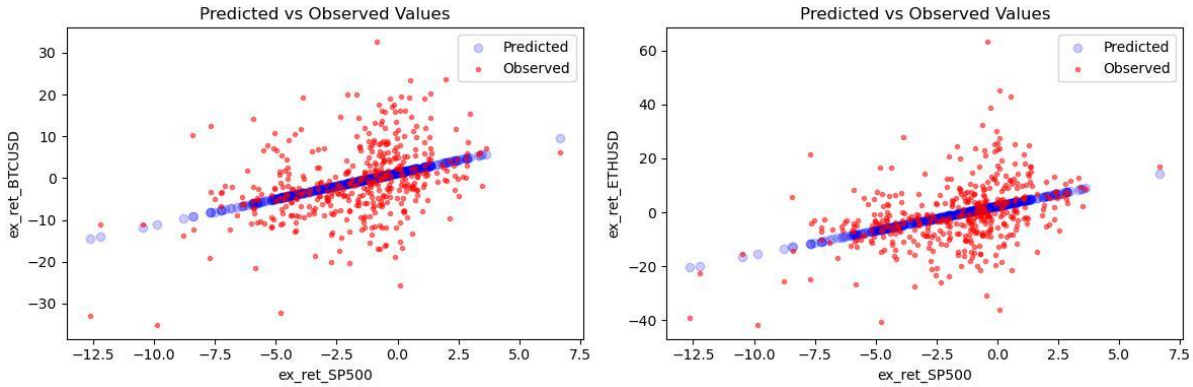


Figure 7: Regression using Robust Standard Errors

However, we did observe a change in the standard errors of the models. Both models experienced an increase in standard errors. The BTC model saw the standard error for the intercept coefficient increase from 0.465 to 0.621 and the standard error for the slope coefficient increase from 0.149 to 0.180.

Similarly, our ETH model saw the standard error of the intercept coefficient increase from 0.636 to 0.939 and the standard error for the slope coefficient increase from 0.204 to 0.246.

The increase in the standard errors of our model coefficients would be expected as our robust model is making the model more conservative to account for violations of the regression assumptions.

A lower F-statistic was recorded for both models however we still reject the null hypothesis that our slope coefficient is equal to zero.

In addition, I also re-ran the tests for normality, serial correlation and heteroskedasticity on the residuals of this model with no changes recorded in the results of those tests.

No improvement has resulted from our use of standard errors in this regression. We saw no improvement in the autocorrelation of our residuals.

Statistical Significance

We will also examine if there has been any change in the significance level of our regression coefficients as a result of using robust standard errors in the regression.

BTC

Coefficient	Z-value	Crit Val	P>|z|	[0.025	0.975]
$\hat{\alpha}$	2.095	1.96	0.036	0.084	2.518
$\hat{\beta}$	6.891	1.96	0.000	0.888	1.594

Table 5: Summary of results of BTC Model using robust standard errors to test statistical significance

By carrying out the same testing methods as in section B.ii , it is evident both the $\hat{\alpha}$ and the $\hat{\beta}$ remain statistically significant at the 5% level. One thing we do notice from Table 5 is that the p-value of the intercept has increased indicating the intercept coefficient of our model is slightly less statistically significant using robust standard

errors. We can also observe from Table 5 than the 95% confidence interval for both coefficients has widened as a result of the increase in standard errors.

ETH

Coefficient	Z-value	Crit Val	P>|z|	[0.025	0.975]
$\hat{\alpha}$	2.537	1.96	0.011	0.542	4.224
$\hat{\beta}$	7.349	1.96	0.000	1.328	2.294

Table 6: Summary of results of ETH Model using robust standard errors to test statistical significance

As is the case with our Bitcoin model, our regression coefficients for our Ethereum model remain statistically significant at the 5% level. We again notice a slightly larger p-value for our intercept coefficient and wider 95% confidence intervals for both coefficients.

Explanation

The increase in the p-value of $\hat{\alpha}$ in both of our models can be explained by the increase in the standard error for the intercept coefficient experienced by both models. The increase in the standard error of our $\hat{\beta}$ coefficients was much more modest compared to the increase observed in the standard errors of $\hat{\alpha}$, which may explain why we did not observe much change in the p-value associated with our slope coefficient.

D. Confidence Interval

In order to test the null hypothesis that the β (risk) of each cryptocurrency is the same as the average risk over the entire market we construct a 95% confidence interval.

$H_0: \beta = 1$: The crypto currency is as risky as the market

$H_A: \beta \neq 1$: The crypto currency has a different risk than the market

Model	Coefficient	Std Err	T value	P>|t|	[0.025	0.975]
BTC	1.2409	0.149	1.616	0.107	0.948	1.534
ETH	1.8107	0.204	3.978	0.000	1.410	2.211

Table 7: Confidence Intervals for Beta of BTC and ETH Models

BTC

We can observe from Table 6 that our 95% confidence interval ([0.948, 1.534]) contains the value of 1. We also have a large p-value. Thus, we fail to reject H_0 . As a result, we cannot conclude with a level of 95% confidence that Bitcoin exhibits any more or less risk than the market

ETH

The 95% confidence interval for Ethereum of [1.410, 2.211] does not contain the value of 1. In addition, we observe a p-value of 0. Therefore, we reject H_0 and can conclude with 95 % confidence that Ethereum exhibits more risk than the market.

D.i Interpretation of Results

The results, at first, did not align with my expectations regarding the risk profiles of the cryptocurrencies. I anticipated that both cryptocurrencies would display riskier returns than the broader market. The fact that I couldn't demonstrate that Bitcoin's β was not significantly different from 1, indicating no difference in risk compared to the market, surprised me. This conflicted with my prior understanding that Bitcoin prices were more volatile than the market. In addition to this prior knowledge we also saw earlier in our analysis that for every 1% change in market returns Bitcoin would experience a corresponding change of 1.2409.

It was only when I remembered the difference between how β measures risk and how volatility measures risk that the reason for Bitcoin's β not differing from the market started to make sense.

Volatility, measured by standard deviation, captures the total risk of an asset. This includes systematic and unsystematic risk. β on the other hand only measures systematic risk i.e. the risk inherent to the market. Thus it is possible for an asset such as Bitcoin to exhibit higher volatility/ total risk while at the same time exhibiting a $\beta = 1$ since the measure of volatility encompasses more than just risk relative to the market.

This presents a potential limitation of the Capital Asset Pricing Model as it only focuses on systematic risk and ignores the effects of unsystematic risk.

E. Linear Regression on Two Subsamples

We will now split our samples into two subsets in order to assess whether there is a structural change or a significant shift in the relationship between the dependent and independent variables at a specific point in the data

E.i Data Splitting

I split my data into two subsets by choosing the midpoint as the date to split the data on. Each subset contains 215 observations before and 215 observations after the split date.

E.ii Subsample Regression

We are now ready to run a regression on each subset.

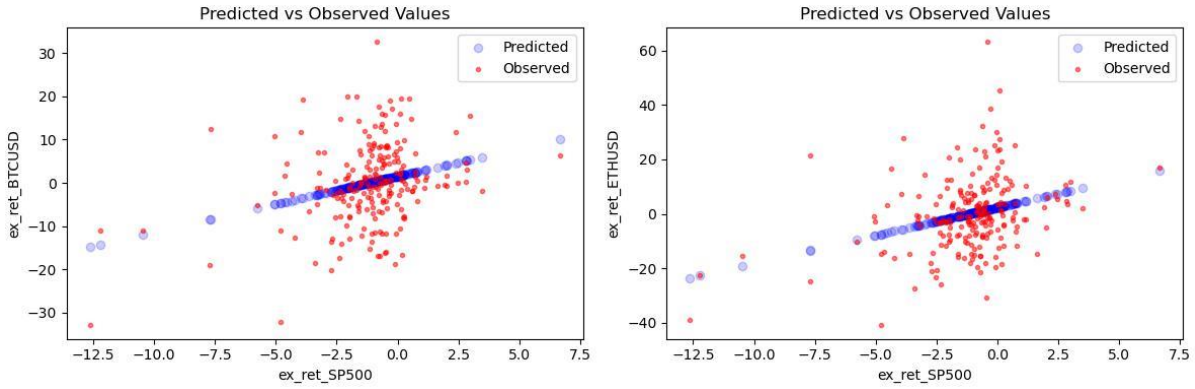


Figure 8: Regression of BTC and ETH using Subset 1

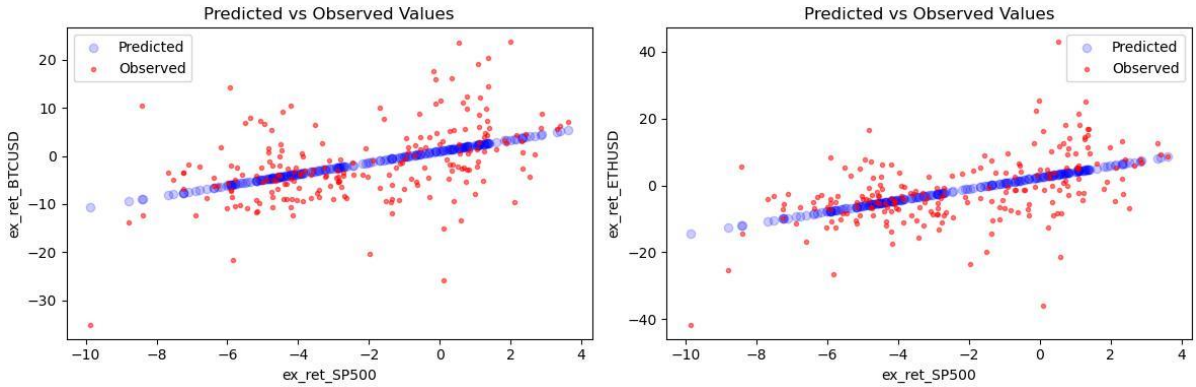


Figure 9: Regression of BTC and ETH using Subset 2

See summary of results below:

BTC:

BTC	Subset 1	Subset 2
Intercept	1.5032	1.0197
SE (intercept)	0.693	0.613
SP500 Coefficient	1.2884	1.1898
SE (SP500)	0.287	0.166
p-value (SP500)	0	0
R-squared	0.086	0.193
Adjusted R-squared	0.082	0.19
F-statistic	20.14	51.28
Prob (F-statistic)	1.17e-05	1.27e-11
AIC	1562	1464
BIC	1569	1471
Jarque-Bera	6.077	33.935
Prob (JB)	0.0479	4.28e-08
Durbin-Watson	1.431	1.522
Skewness	0.23	0.178
Kurtosis	3.682	4.909

Table 8: Regression Results Summary Subsets of BTC Model

Model	$\hat{\alpha}$ t-value	$\hat{\beta}$ t-value	Crit Val	$\hat{\alpha}$ P >|t|	$\hat{\beta}$ P >|t|	$\hat{\alpha}$ CI	$\hat{\beta}$ CI
Subset 1	2.171	4.488	1.96	0.031	0	[0.138, 2.868]	[0.723, 1.854]
Subset 2	1.664	7.161	1.96	0.097	0	[-0.188,2.227]	[0.862,1.517]

Table 9: Summary of results of BTC model subsets needed for hypothesis testing

Sample 1:

$$(R_{it} - R_{ft}) = 1.5032 + 1.2884(R_{mt} - R_{ft}) + u_t$$

From our sample 1 model we obtained an intercept coefficient of 1. 5032.We can interpret this result as the log excess return of Bitcoin when the log excess return of the market is zero. Given the intercept is positive we expect the asset to outperform the market when the return from the market is equal to zero over this period.

The model has also produced a $\hat{\beta}$ coefficient of 1.2884. This coefficient captures the relation between the price of Bitcoin and movements in the wider market over the sample period. We can interpret this result as for every percentage change in the log excess return of the market, the price of Bitcoin would be expected to experience a corresponding change in the same direction of 1.2884 percentage points

Statistical Significance

From Table 9 we can conclude that both our $\hat{\alpha}$ and $\hat{\beta}$ coefficients are statistically significant at the 5% level.

Sample 2:

$$(R_{it} - R_{ft}) = 1.0197 + 1.1898(R_{mt} - R_{ft}) + u_t$$

From our sample 2 model we obtained an intercept coefficient of 1.0197. Given the intercept is positive we expect the asset to outperform the market when the return from the market is equal to zero over this period.

The model has also produced a $\hat{\beta}$ coefficient of 1.1898. We can interpret this result as for every percentage change in the log excess return of the market, the price of Bitcoin would be expected to experience a corresponding change in the same direction of 1.2884 percentage points.

Statistical Significance

From Table 9 we can conclude that our $\hat{\beta}$ coefficient is statistically significant at the 5% level. However, the $\hat{\alpha}$ coefficient for this sample with its t-statistic of 1.664, p-value of 0.097 and 95% confidence interval of [-0.188,2.227] is not statistically significant at this significance level. The intercept coefficient for this model is statistically significant at the 10% level.

Comparing Samples:

Differences in the intercept and slope coefficients are observed between the two subsets. We obtain lower values for both coefficients in the second sample compared to the first.

ETH:

ETH	Subset 1	Subset 2
Intercept	2.3746	2.4083
SE (intercept)	1.001	0.753
SP500 Coefficient	2.0547	1.7073
SE (SP500)	0.415	0.204
p-value (SP500)	0	0
R-squared	0.103	0.246
Adjusted R-squared	0.099	0.243
F-statistic	24.51	69.88
Prob (F-statistic)	1.5e-06	8e-15
AIC	1721	1553
BIC	1728	1560
Jarque-Bera	91.222	107.397
Prob (JB)	1.55e-20	4.78e-24
Durbin-Watson	1.284	1.62
Skewness	0.909	-0.025
Kurtosis	5.613	6.454

Table 10: Regression Results Summary Subsets of ETH Model

Model	$\hat{\alpha}$ **t-value**	$\hat{\beta}$ **t-value**	**Crit Val**	$\hat{\alpha}$ **P>|t|**	$\hat{\beta}$ **P>|t|**	$\hat{\alpha}$ CI	$\hat{\beta}$ CI
Subset 1	2.372	4.951	1.96	0.019	0	[0.401, 4.348]	[1.237, 2.873]
Subset 2	3.198	8.36	1.96	0.002	0	[0.924,3.893]	[1.305,2.11]

Table 11: Summary of results of ETH model subsets needed for hypothesis testing

Sample 1:

$$(R_{it} - R_{ft}) = 2.3746 + 2.0547(R_{mt} - R_{ft}) + u_t$$

From our sample 1 model we obtained an intercept coefficient of 2.3746. Given the intercept is positive we expect the asset to outperform the market when the return from the market is equal to zero over this period.

The model has also produced a $\hat{\beta}$ coefficient of 2.0547. We can interpret this result as for every percentage change in the log excess return of the market over the sample period, the excess return of Ethereum would be expected to experience a corresponding change in the same direction of 2.0547 percentage points

Statistical Significance

From Table 11 it is evident both our $\hat{\alpha}$ and $\hat{\beta}$ coefficients are statistically significant at the 5% level.

Sample 2:

$$(R_{it} - R_{ft}) = 2.4083 + 1.7073(R_{mt} - R_{ft}) + u_t$$

From our sample 2 model we obtained an intercept coefficient of 2.4083. As the intercept is positive we expect the asset to outperform the market when the return from the market is equal to zero over this period.

The model has also produced a $\hat{\beta}$ coefficient of 1.7073. We can interpret this result as for every percentage change in the log excess return of the market over the sample period, the price of Ethereum would be expected to experience a corresponding change in the same direction of 1.7073 percentage points.

Statistical Significance

From Table 11 it is evident both our $\hat{\alpha}$ and $\hat{\beta}$ coefficients are statistically significant at the 5% level.

Comparing Samples:

Differences in the intercept and slope coefficients are observed between the two subsets. Sample 2 reports a higher intercept than sample 1, however, sample 1 has a higher slope coefficient than sample 2. While both a $\hat{\beta}$ coefficients are positive there is the size of the relationship between the excess returns of Ethereum and that of that market in sample 1 is higher than that in sample 2.

E.iii Chow Test

The Chow Test is a parameter stability test used to determine whether the coefficients in the two different regression models (applied to the subgroups) are statistically different from each other i.e. if there is a structural break. (Chow, 1960)

The Chow Test statistic is calculated using the below formula (Brooks, 2014):

$$Chow = \frac{RSS - (RSS_1 + RSS_2)}{RSS_1 + RSS_2} \times \frac{T - 2k}{k}$$

Where:

- RSS : Residual sum of squares for whole sample
- RSS_1: Residual sum of squares for sub-sample 1
- RSS_2: Residual sum of squares for sub-sample 2
- T : Total number of observations

- k : Number of regressors in each sample

The hypotheses for the test are:

$$H_0: The\ coefficients\ of\ each\ subset\ are\ the\ same\ (no\ structural\ break)$$

$$H_A: The\ coefficients\ of\ each\ subset\ are\ not\ the\ same\ (\ structural\ break)$$

BTC

We obtained a chow test statistic of 0.193 with a corresponding p value of 0.96.

Chow Test Statistic	0.0192839
p-value:	0.980902

Fail to reject the null hypothesis: No evidence of a significant difference in coefficients between the two periods, indicating stability.

As we have a large p-value we fail to reject H_0 at a 5% level of significance. There is no evidence of a significant structural break in the data. As a result, we observe that there is no significant difference between the coefficients in each sample. This is somewhat surprising as our subsample 1 contains the data points associated with the extreme levels of volatility in the Bitcoin market at the onset of the Covid-19 pandemic. As such I would have expected some evidence of instability. A possible reason for this is the use of weekly data in our analysis which may have helped to smooth out the large daily price moves seen at the onset of the pandemic such as the 15th March 2020 when Bitcoin lost half of its value in one day.

ETH:

We obtained a chow test statistic of 0.276 with a corresponding p value of 0.759.

Chow Test Statistic	0.276131
p-value:	0.758848

Fail to reject the null hypothesis: No evidence of a significant difference in coefficients between the two periods, indicating stability.

As was the case with our Bitcoin samples, we fail to reject H_0 at a %5 significance level due to the large p-value reported in test . As a result, we observe that there is no significant difference between the coefficients in each sample for our Ethereum model.

F. Linear Regression Without a Constant

In the final section of our analysis, we will again run a regression but this time we will omit the constant from our models. Given that in section B we found that the $\hat{\alpha}$ our models was not statistically significant and that the CAPM model assumes α is zero, it will be intriguing to observe any potential changes to our regression results.

Regression model without a constant:

$$y_t = \beta_1 x_t + u_t$$

Where:

y_t : Dependent Variable = $\left(R_{it} - R_{ft}\right)$

x_t : Independent Variable = $\left(R_{m_t} - R_{ft}\right)$

β_1: Coefficient of Independent Variable

u_t : Error (Residuals)

In the case of running this regression model without the constant value, we assume that when our independent variable is zero, our dependent variable is also zero. It also means that the regression line does not a y-intercept and instead passes through the origin as demonstrated in Figure 10 below.

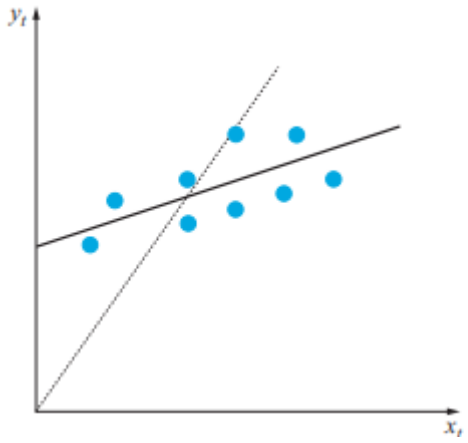


Figure 10: The effect of no intercept on the regression line (Brooks,2014)

For our CAPM analysis, regressing our models without the constant means we are assuming that when the excess returns of the market is zero, the excess returns of the cryptocurrency is also zero (as $\alpha = 0$).

Removing the constant term from our regression can result in some undesirable implications. It can result in potentially significant biases being introduced in the estimates of the slope coefficients of the model. In addition, it can result in the R^2 goodness-of-fit measure turning negative. This implies that the sample average ($\bar{y}$) explains more of the variation in the dependent variable than the explanatory variables do (Brooks,2014).

F.i Regression Results

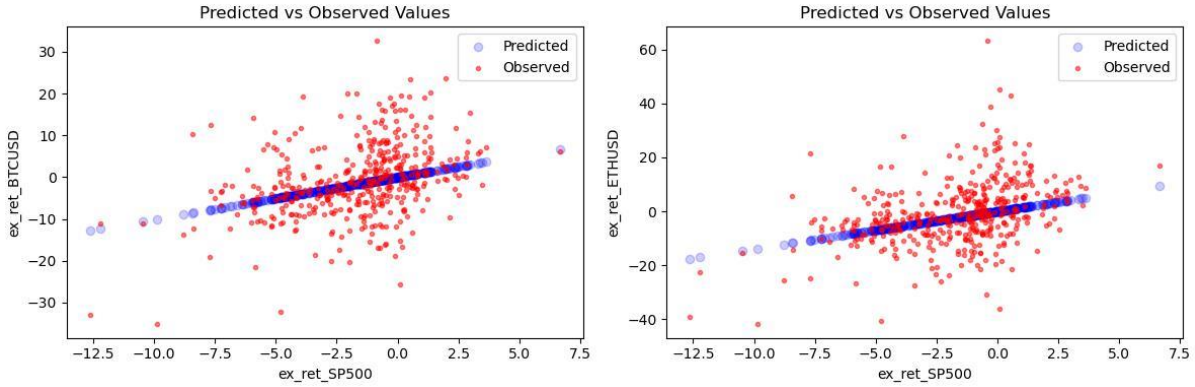


Figure 11: Regression of BTC and ETH models without constant

Model	BTC	ETH
SP500 Coefficient	1.0133	1.3937
SE (SP500)	0.126	0.173
p-value (SP500)	0	0
R-squared	0.131	0.131
Adjusted R-squared	0.129	0.129
F-statistic	64.73	64.66
Prob (F-statistic)	8.44e-15	8.72e-15
AIC	3033	3308
BIC	3037	3312
Jarque-Bera	32.868	259.329
Prob (JB)	7.29e-08	4.87e-57
Durbin-Watson	1.44	1.359
Skewness	0.199	0.683
Kurtosis	4.293	6.546

Table 12: Regression Results Summary Without Constant

F.ii Model Interpretation

Regression Coefficients:

BTC

For our Bitcoin model we obtain a $\hat{\beta}$ coefficient of 1.0133 which is statistically significant at a 1% level of significance. As a result, we would expect the excess return of Bitcoin to closely follow with slightly larger moves in returns than the market. For every percentage change in the excess return of the market we would expect the excess return of Bitcoin to move 1.0133 percentage points in the same direction.

ETH

Our Ethereum model has a $\hat{\beta}$ coefficient of 1.3937 which is statistically significant at a 1% level of significance. For every percentage change in the excess return of the market we would expect the excess return of Ethereum to move 1.3937 percentage points in the same direction.

F-Statistic

BTC

The model has an F-Statistic of 64.73 with an associated p-value close to zero. Thus, we reject the null hypothesis at the 1% significance level.

ETH

Our Ethereum model has an F-Statistic of 64.66 with an associated p-value of very close to zero. Thus, we reject the null hypothesis at the 1% significance level.

R^2

As mentioned above, the R^2 value does not carry much meaning when regressing without an intercept and so we will not use this metric to assess the fit of the model.

AIC and BIC

BTC

The model reports an AIC value of 3033 and a BIC value of 3037. Both metrics increased compared to our model in section B. At first this was surprising given that BIC penalises extra parameters more harshly I initially expected to see a decrease in the BIC. However the fact that the intercept coefficient is statistically significant in section B, explains why the model's fit has worsened slightly when we remove the constant.

ETH

The model reports an AIC value of 3308 and a BIC value of 3312. Both metrics increased compared to our model in section B with the reason for this the same as outlined above for our Bitcoin model.

Skewness

BTC

We obtain a skew of 0.199 indicating the model has fatter or longer tails to the right

ETH

We obtain a skew of 0.683 indicating the model has fatter or longer tails to the right

Kurtosis

BTC

The model has a kurtosis of 4.293, indicating the model has some outliers.

ETH

The model has a slightly larger kurtosis than our BTC model of 6.546, indicating the model has some more outliers.

Jarque-Bera

BTC

The model has a JB statistic of 32.868 with an associated p-value of very close to zero. Therefore, we reject the null hypothesis that the residuals of the model are normally distributed.

ETH

Our Ethereum model has a JB statistic of 259.329 with a corresponding p-value of very close to zero. Thus we draw the same conclusion as with our BTC model.

Omnibus

BTC

The BTC model reported an omnibus statistic of 16.895 with an associated p-value of 0. This indicates that the residuals of the model are not normally distributed.

ETH

The model has an omnibus statistic of 70.303 with an associated p-value of 0. This again indicates that the residuals of the model are not normally distributed.

G. Conclusion

CAPM is an effective framework for modelling the risk and return of cryptocurrencies for multiple reasons. The simplicity of the CAPM model gives us a straightforward methodology to estimate expected returns. The fact that CAPM only takes systematic risk into account plays into this simplicity. As previously mentioned, systematic risk is the risk inherent to the market and this is what CAPM uses to estimate expected returns. The functionality of CAPM as a benchmark is also particularly useful. This allows us to compare different assets to the market.

In the analysis of our models some benefits of using the CAPM to analyse cryptocurrencies were evident. We found that for the most part the intercept and slope coefficients were statistically significant in explaining the dependent variable. In addition we also observed that our data exhibited homoskedasticity, fulfilling the classical linear regression assumption of constant variance over time. The Chow Test was also conducted on our models to test for structural breaks in our data with no such breaks found. This indicated stability in our model parameters.

However, there are certain limitations of CAPM. The CAPM assumes that the returns of an asset follow a normal distribution however over the course of our analysis we found that this was not the case for our cryptocurrencies as we saw returns exhibit skewness and fatter tail that you would not expect to see in a normal distribution. In addition, the models failed our statistical normality tests.

Another significant limitation of CAPM is that it assumes systematic risk (β) is the main driver of returns which is an oversimplification in reality. As a result, the model fails to take unsystematic risk into account and thus does not account for total risk. This explains why we could not significantly prove that β was different to 1 in section D.i. A potential improvement to the model would be to use standard deviation (volatility) to account for risk as this captures total risk.

We can also conclude from the low R^2 values of our models that the excess return of the S&P 500 Index does little to explain the excess returns of our two cryptocurrencies, Bitcoin and Ethereum. I also believe that the use of the S&P 500 to represent the market in our CAPM

analysis was not particularly useful. The reason for this is that cryptocurrencies deviate significantly from traditional equity markets and thus I do not believe an equity index such as the S&P 500 can truly capture the variance and risk of cryptocurrencies.

In conclusion, the CAPM is an effective tool to analyse returns in conventional markets. However, its application to cryptocurrency markets has limitations as cryptocurrencies as discussed above. Cryptocurrencies are quite a new asset with unique characteristics compared to traditional markets. The non-normal returns, high volatility and other external factors unique to cryptocurrencies make it difficult for the CAPM to effectively model cryptocurrency returns.

References

Brooks, C. (2014). Introductory econometrics for finance (third edition). Cambridge, UK: Cambridge University Press

Wooldridge, J. M. (2010). *Econometric Analysis of Cross Section and Panel Data* (2nd ed.). MIT Press.

Wooldridge, J. M. (2016). *Introductory Econometrics: A Modern Approach* (6th ed.). Cengage Learning.

Durbin, J., & Watson, G. S. (1951). Testing for serial correlation in least squares regression ii.

Godfrey, L. G. (1978). Testing against general autoregressive and moving average error models when the regressors include lagged dependent variables.

Breusch, T. S. (1978). Testing for autocorrelation in dynamic linear models. Australian Economic Papers.

Goldfeld, S. M., & Quandt, R. E. (1965). "Some Tests for Homoscedasticity." *Journal of the American Statistical Association*.

Chow, G. C. (1960). Tests of equality between sets of coefficients in two linear regressions

Newey, W. K., & West, K. D. (1987). A simple, positive semi-definite, heteroskedasticity and autocorrelation consistent covariance matrix.

Appendix

Some of my regression outputs have been cut off when exported from Jupyter so please find them here:

Non-Robust Regression:

Regression of BTCUSD on SP500:

OLS Regression Results

Dep. Variable:	ex_ret_BTCUSD	R-squared:	0.139
Model:	OLS	Adj. R-squared:	0.137
Method:	Least Squares	F-statistic:	69.24
Date:	Sun, 13 Oct 2024	Prob (F-statistic):	1.17e-15
Time:	22:07:25	Log-Likelihood:	-1511.7
No. Observations:	431	AIC:	3027.
Df Residuals:	429	BIC:	3035.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.3006	0.465	2.796	0.005	0.386	2.215
ex_ret_SP500	1.2409	0.149	8.321	0.000	0.948	1.534

Omnibus:	16.660	Durbin-Watson:	1.465
Prob(Omnibus):	0.000	Jarque-Bera (JB):	30.750
Skew:	0.219	Prob(JB):	2.10e-07
Kurtosis:	4.233	Cond. No.	3.84

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Regression of ETHUSD on SP500:

OLS Regression Results

Dep. Variable:	ex_ret_ETHUSD	R-squared:	0.155
Model:	OLS	Adj. R-squared:	0.153
Method:	Least Squares	F-statistic:	78.95
Date:	Sun, 13 Oct 2024	Prob (F-statistic):	1.77e-17
Time:	22:07:25	Log-Likelihood:	-1646.3
No. Observations:	431	AIC:	3297.
Df Residuals:	429	BIC:	3305.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.3827	0.636	3.749	0.000	1.134	3.632
ex_ret_SP500	1.8107	0.204	8.886	0.000	1.410	2.211

Omnibus:	68.837	Durbin-Watson:	1.389
Prob(Omnibus):	0.000	Jarque-Bera (JB):	248.185
Skew:	0.674	Prob(JB):	1.28e-54
Kurtosis:	6.464	Cond. No.	3.84

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Robust Regression:

```
Regression of BTCUSD on SP500 using robust standard errors:
                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_BTCUSD   R-squared:                       0.139
Model:                            OLS   Adj. R-squared:                  0.137
Method:                 Least Squares   F-statistic:                     47.48
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           1.99e-11
Time:                        22:07:25   Log-Likelihood:                -1511.7
No. Observations:                 431   AIC:                             3027.
Df Residuals:                     429   BIC:                             3035.
Df Model:                           1
Covariance Type:                  HAC
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      1.3006      0.621      2.095      0.036       0.084       2.518
ex_ret_SP500   1.2409      0.180      6.891      0.000       0.888       1.594
==============================================================================
Omnibus:                       16.660   Durbin-Watson:                   1.465
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               30.750
Skew:                           0.219   Prob(JB):                     2.10e-07
Kurtosis:                       4.233   Cond. No.                         3.84
==============================================================================

Notes:
[1] Standard Errors are heteroscedasticity and autocorrelation robust (HAC) using 12 lags and with small sample correction
Regression of ETHUSD on SP500 using robust standard errors:
                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_ETHUSD   R-squared:                       0.155
Model:                            OLS   Adj. R-squared:                  0.153
Method:                 Least Squares   F-statistic:                     54.01
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           1.02e-12
Time:                        22:07:25   Log-Likelihood:                -1646.3
No. Observations:                 431   AIC:                             3297.
Df Residuals:                     429   BIC:                             3305.
Df Model:                           1
Covariance Type:                  HAC
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      2.3827      0.939      2.537      0.011       0.542       4.224
ex_ret_SP500   1.8107      0.246      7.349      0.000       1.328       2.294
==============================================================================
Omnibus:                       68.837   Durbin-Watson:                   1.389
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              248.185
Skew:                           0.674   Prob(JB):                     1.28e-54
Kurtosis:                       6.464   Cond. No.                         3.84
==============================================================================

Notes:
[1] Standard Errors are heteroscedasticity and autocorrelation robust (HAC) using 12 lags and with small sample correction
```

Subsample 1 Regression:

Regression of BTCUSD on SP500:

OLS Regression Results

Dep. Variable:	ex_ret_BTCUSD	R-squared:	0.086
Model:	OLS	Adj. R-squared:	0.082
Method:	Least Squares	F-statistic:	20.14
Date:	Sun, 13 Oct 2024	Prob (F-statistic):	1.17e-05
Time:	22:07:25	Log-Likelihood:	-779.05
No. Observations:	216	AIC:	1562.
Df Residuals:	214	BIC:	1569.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.5032	0.693	2.171	0.031	0.138	2.868
ex_ret_SP500	1.2884	0.287	4.488	0.000	0.723	1.854

Omnibus:	5.499	Durbin-Watson:	1.431
Prob(Omnibus):	0.064	Jarque-Bera (JB):	6.077
Skew:	0.230	Prob(JB):	0.0479
Kurtosis:	3.682	Cond. No.	2.86

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Regression of ETHUSD on SP500:

OLS Regression Results

Dep. Variable:	ex_ret_ETHUSD	R-squared:	0.103
Model:	OLS	Adj. R-squared:	0.099
Method:	Least Squares	F-statistic:	24.51
Date:	Sun, 13 Oct 2024	Prob (F-statistic):	1.50e-06
Time:	22:07:25	Log-Likelihood:	-858.67
No. Observations:	216	AIC:	1721.
Df Residuals:	214	BIC:	1728.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.3746	1.001	2.372	0.019	0.401	4.348
ex_ret_SP500	2.0547	0.415	4.951	0.000	1.237	2.873

Omnibus:	41.858	Durbin-Watson:	1.284
Prob(Omnibus):	0.000	Jarque-Bera (JB):	91.222
Skew:	0.909	Prob(JB):	1.55e-20
Kurtosis:	5.613	Cond. No.	2.86

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Subsample 2 Regression:

Regression of BTCUSD on SP500:

OLS Regression Results

Dep. Variable:	ex_ret_BTCUSD	R-squared:	0.193
Model:	OLS	Adj. R-squared:	0.190
Method:	Least Squares	F-statistic:	51.28
Date:	Sun, 13 Oct 2024	Prob (F-statistic):	1.27e-11
Time:	22:07:25	Log-Likelihood:	-730.12
No. Observations:	216	AIC:	1464.
Df Residuals:	214	BIC:	1471.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.0197	0.613	1.664	0.097	-0.188	2.227
ex_ret_SP500	1.1898	0.166	7.161	0.000	0.862	1.517

Omnibus:	13.989	Durbin-Watson:	1.522
Prob(Omnibus):	0.001	Jarque-Bera (JB):	33.935
Skew:	0.178	Prob(JB):	4.28e-08
Kurtosis:	4.909	Cond. No.	4.78

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Regression of ETHUSD on SP500:

OLS Regression Results

Dep. Variable:	ex_ret_ETHUSD	R-squared:	0.246
Model:	OLS	Adj. R-squared:	0.243
Method:	Least Squares	F-statistic:	69.88
Date:	Sun, 13 Oct 2024	Prob (F-statistic):	8.00e-15
Time:	22:07:25	Log-Likelihood:	-774.71
No. Observations:	216	AIC:	1553.
Df Residuals:	214	BIC:	1560.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.4083	0.753	3.198	0.002	0.924	3.893
ex_ret_SP500	1.7073	0.204	8.360	0.000	1.305	2.110

Omnibus:	23.192	Durbin-Watson:	1.620
Prob(Omnibus):	0.000	Jarque-Bera (JB):	107.397
Skew:	-0.025	Prob(JB):	4.78e-24
Kurtosis:	6.454	Cond. No.	4.78

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Regression Without a Constant:

Regression of BTCUSD on SP500:

OLS Regression Results

Dep. Variable:	ex_ret_BTCUSD	R-squared (uncentered):	0.131
Model:	OLS	Adj. R-squared (uncentered):	0.129
Method:	Least Squares	F-statistic:	64.73
Date:	Sun, 13 Oct 2024	Prob (F-statistic):	8.44e-15
Time:	22:07:25	Log-Likelihood:	-1515.6
No. Observations:	431	AIC:	3033.
Df Residuals:	430	BIC:	3037.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
ex_ret_SP500	1.0133	0.126	8.046	0.000	0.766	1.261

Omnibus:	16.895	Durbin-Watson:	1.440
Prob(Omnibus):	0.000	Jarque-Bera (JB):	32.868
Skew:	0.199	Prob(JB):	7.29e-08
Kurtosis:	4.293	Cond. No.	1.00

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Regression of ETHUSD on SP500:

OLS Regression Results

Dep. Variable:	ex_ret_ETHUSD	R-squared (uncentered):	0.131
Model:	OLS	Adj. R-squared (uncentered):	0.129
Method:	Least Squares	F-statistic:	64.66
Date:	Sun, 13 Oct 2024	Prob (F-statistic):	8.72e-15
Time:	22:07:25	Log-Likelihood:	-1653.2
No. Observations:	431	AIC:	3308.
Df Residuals:	430	BIC:	3312.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
ex_ret_SP500	1.3937	0.173	8.041	0.000	1.053	1.734

Omnibus:	70.303	Durbin-Watson:	1.359
Prob(Omnibus):	0.000	Jarque-Bera (JB):	259.329
Skew:	0.683	Prob(JB):	4.87e-57
Kurtosis:	6.546	Cond. No.	1.00

Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Code

In [1]:
```
import os
print(os.getcwd())
```

```
C:\Users\rorym\OneDrive\Documents\Final Year\Econometrics of Financial Markets
\CAPM Assignment
```

Import Libraries

In [2]:
```
#Import relevant modules
import pandas as pd
import numpy as np
import statsmodels
import scipy
import statsmodels.formula.api as smf
import statsmodels.stats.api as sms
import statsmodels.api as smp
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib
import scipy.stats as sts
from tabulate import tabulate
from statsmodels.compat import lzip
from statsmodels.iolib.summary2 import summary_col
```

Gathering Data

a) & b)

In [3]:
```
#Read in and calculate log returns of data

#Function used for efficiency

def log_data(excel, subject : str):

    df = pd.read_excel(excel, index_col = "Date", parse_dates= True )
    df["log_ret_" + subject] = 100*np.log(df[subject]).diff()
    df.dropna(inplace = True)
    return df

spx = log_data("SP500.xls", subject = "SP500")
btc = log_data("btc.xls", subject = "BTCUSD")
eth = log_data("eth.xls", subject = "ETHUSD")
fed_funds = pd.read_excel("fedfunds.xls",index_col = "Date", parse_dates = True)
```

In [4]:
```
#Merge into a master dataframe for convenience

master_df =  pd.merge(spx,btc, how = "left", on = "Date" )
master_df = pd.merge(master_df,eth, how = "left", on ="Date")
master_df = pd.merge(master_df,fed_funds, how = "left", on ="Date")
master_df.dropna(inplace= True )

# Calculate excess returns
master_df["ex_ret_SP500"] =  master_df["log_ret_SP500"] - master_df["EFFR"]

for coin in ["BTCUSD", "ETHUSD"]:

    master_df["ex_ret_" + coin] = master_df["log_ret_"+ coin] - master_df["EFFR"
```

In [5]:
```
master_df.head()
```

Out[5]:

Date	SP500	log_ret_SP500	BTCUSD	log_ret_BTCUSD	ETHUSD	log_ret_ETHUSD	EFFR	e
2016-06-05	2100.170	0.928508	551.332857	15.087224	13.852857	11.079503	0.350	
2016-06-12	2110.442	0.487911	599.260000	8.335684	14.525714	4.742897	0.370	
2016-06-19	2075.018	-1.692758	734.215714	20.310731	16.532857	12.942926	0.374	
2016-06-26	2081.666	0.319871	655.518571	-11.337624	13.661429	-19.077331	0.386	
2016-07-03	2061.842	-0.956878	666.557143	1.669924	12.572857	-8.303613	0.388	

c)

In [6]:
```
def regression (df,crypto, constant = True, robust = False):

    if robust == True:
        model = smf.ols(formula = f"ex_ret_{crypto} ~ ex_ret_SP500",
                        data = df).fit(cov_type="HAC",cov_kwds={'maxlags':12,'us
        print(f"Regression of {crypto} on SP500 using robust standard errors:\n"
        return model

    if constant == True:
        model = smf.ols(formula = f"ex_ret_{crypto} ~ ex_ret_SP500", data = df).

    elif constant == False: # Generalise formula to be re-used in Part G
        model =  smf.ols(formula = f"ex_ret_{crypto} ~ ex_ret_SP500 -1", data =
    print(f"Regression of {crypto} on SP500:\n\n", model.summary(),"\n")


    return model

btc_regress = regression(master_df, "BTCUSD")
eth_regress = regression(master_df, "ETHUSD")
```

```
Regression of BTCUSD on SP500:

                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_BTCUSD   R-squared:                       0.139
Model:                            OLS   Adj. R-squared:                  0.137
Method:                 Least Squares   F-statistic:                     69.24
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           1.17e-15
Time:                        22:07:25   Log-Likelihood:                -1511.7
No. Observations:                 431   AIC:                             3027.
Df Residuals:                     429   BIC:                             3035.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
Intercept        1.3006      0.465      2.796      0.005       0.386       2.21
5
ex_ret_SP500     1.2409      0.149      8.321      0.000       0.948       1.53
4
==============================================================================
Omnibus:                       16.660   Durbin-Watson:                   1.465
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               30.750
Skew:                           0.219   Prob(JB):                     2.10e-07
Kurtosis:                       4.233   Cond. No.                         3.84
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctl
y specified.

Regression of ETHUSD on SP500:

                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_ETHUSD   R-squared:                       0.155
Model:                            OLS   Adj. R-squared:                  0.153
Method:                 Least Squares   F-statistic:                     78.95
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           1.77e-17
Time:                        22:07:25   Log-Likelihood:                -1646.3
No. Observations:                 431   AIC:                             3297.
Df Residuals:                     429   BIC:                             3305.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
Intercept        2.3827      0.636      3.749      0.000       1.134       3.63
2
ex_ret_SP500     1.8107      0.204      8.886      0.000       1.410       2.21
1
==============================================================================
Omnibus:                       68.837   Durbin-Watson:                   1.389
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              248.185
Skew:                           0.674   Prob(JB):                     1.28e-54
Kurtosis:                       6.464   Cond. No.                         3.84
```

```
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctl
y specified.
```

In [7]:
```
def jarque_bera(reg, crypto):

    test = sms.jarque_bera(reg.resid)
    labels = ['JB Statistic', 'Chi-Squared','Skewness','Kurtosis']
    table = tabulate(zip(labels,test),tablefmt = "fancy_grid")

    print(f"Results of Jarque-Bera normality test for {crypto} below:\n",table)

jarque_bera(btc_regress, "BTC")
jarque_bera(eth_regress, "ETH")
```

```
Results of Jarque-Bera normality test for BTC below:
╒═══════════════╤═══════════════╕
│ JB Statistic  │ 30.75         │
├───────────────┼───────────────┤
│ Chi-Squared   │  2.10239e-07  │
├───────────────┼───────────────┤
│ Skewness      │  0.218542     │
├───────────────┼───────────────┤
│ Kurtosis      │  4.23339      │
╘═══════════════╧═══════════════╛
Results of Jarque-Bera normality test for ETH below:
╒═══════════════╤═══════════════╕
│ JB Statistic  │ 248.185       │
├───────────────┼───────────────┤
│ Chi-Squared   │   1.28047e-54 │
├───────────────┼───────────────┤
│ Skewness      │   0.674068    │
├───────────────┼───────────────┤
│ Kurtosis      │   6.46447     │
╘═══════════════╧═══════════════╛
```

Correlation Testing

In [8]:
```
def breusch_godfrey(reg,crypto):

    test = sms.acorr_breusch_godfrey(reg, nlags = 12)
    rounded_results = [round(stat, 3) for stat in test]
    labels = ['LM Statistic', 'LM P-Value', 'F Statistic', 'F Statistic P-Value'
    table = tabulate(zip(labels,rounded_results),tablefmt = "fancy_grid")

    print(f"Results of the Breusch-Godfrey auto-correlation test for {crypto} be

breusch_godfrey(btc_regress, "BTC")
breusch_godfrey(eth_regress, "ETH")
```

```
Results of the Breusch-Godfrey auto-correlation test for BTC below:
```

LM Statistic	39.001
LM P-Value	0
F Statistic	3.457
F Statistic P-Value	0

```
Results of the Breusch-Godfrey auto-correlation test for ETH below:
```

LM Statistic	51.761
LM P-Value	0
F Statistic	4.743
F Statistic P-Value	0

Heteroskedasticity Testing

In [9]:
```
def white(reg,crypto):

    test = sms.het_white(reg.resid,reg.model.exog)

    labels = ['LM Statistic', 'LM P-Value', 'F Statistic', 'F Statistic P-Value'
    table = tabulate(zip(labels,test),tablefmt = "fancy_grid")

    print(f"Results of the White Heteroskedasticty test for {crypto} below:\n",t
white(btc_regress, "BTC")
white(eth_regress, "ETH")
```

```
Results of the White Heteroskedasticty test for BTC below:
```

LM Statistic	2.11392
LM P-Value	0.34751
F Statistic	1.05478
F Statistic P-Value	0.349174

```
Results of the White Heteroskedasticty test for ETH below:
```

LM Statistic	0.351172
LM P-Value	0.838965
F Statistic	0.174506
F Statistic P-Value	0.839931

In [10]:
```
def breusch_pagan(reg,crypto):

    test = sms.het_breuschpagan(reg.resid, reg.model.exog)
    labels = ['LM Statistic', 'LM P-Value', 'F Statistic', 'F Statistic P-Value'
    table = tabulate(zip(labels,test),tablefmt = "fancy_grid")

    print(f"Results of the Breusch-Pagan Heteroskedasticty test for {crypto} bel

breusch_pagan(btc_regress, "BTC")
breusch_pagan(eth_regress, "ETH")
```

```
Results of the Breusch-Pagan Heteroskedasticty test for BTC below:
╒═══════════════════╤══════════╕
│ LM Statistic      │ 0.734752 │
├───────────────────┼──────────┤
│ LM P-Value        │ 0.391347 │
├───────────────────┼──────────┤
│ F Statistic       │ 0.732591 │
├───────────────────┼──────────┤
│ F Statistic P-Value │ 0.392522 │
╘═══════════════════╧══════════╛
Results of the Breusch-Pagan Heteroskedasticty test for ETH below:
╒═══════════════════╤══════════╕
│ LM Statistic      │ 0.351153 │
├───────────────────┼──────────┤
│ LM P-Value        │ 0.553461 │
├───────────────────┼──────────┤
│ F Statistic       │ 0.349809 │
├───────────────────┼──────────┤
│ F Statistic P-Value │ 0.554533 │
╘═══════════════════╧══════════╛
```

In [11]:
```
def goldfeld_quandt(reg,crypto):

    test = sms.het_goldfeldquandt(reg.resid, reg.model.exog,split=0.5)
    labels = ['F-statistic', 'p-value']
    table = tabulate(zip(labels,test),tablefmt = "fancy_grid")

    print(f"Results of the Goldfeld-Quandt test for {crypto} below:\n",table)

goldfeld_quandt(btc_regress,"BTCUSD")
goldfeld_quandt(eth_regress,"ETHUSD")
```

```
Results of the Goldfeld-Quandt test for BTCUSD below:
╒═══════════════╤══════════╕
│ F-statistic   │ 0.633351 │
├───────────────┼──────────┤
│ p-value       │ 0.999547 │
╘═══════════════╧══════════╛
Results of the Goldfeld-Quandt test for ETHUSD below:
╒═══════════════╤══════════╕
│ F-statistic   │ 0.458033 │
├───────────────┼──────────┤
│ p-value       │ 1        │
╘═══════════════╧══════════╛
```

D)

In [12]:
```
robust_BTC_result = regression(master_df, "BTCUSD", robust= True)
robust_ETH_result = regression(master_df, "ETHUSD", robust = True)
```

```
Regression of BTCUSD on SP500 using robust standard errors:
                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_BTCUSD   R-squared:                       0.139
Model:                            OLS   Adj. R-squared:                  0.137
Method:                 Least Squares   F-statistic:                     47.48
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           1.99e-11
Time:                        22:07:25   Log-Likelihood:                -1511.7
No. Observations:                 431   AIC:                             3027.
Df Residuals:                     429   BIC:                             3035.
Df Model:                           1
Covariance Type:                  HAC
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      1.3006      0.621      2.095      0.036       0.084       2.518
ex_ret_SP500   1.2409      0.180      6.891      0.000       0.888       1.594
==============================================================================
Omnibus:                       16.660   Durbin-Watson:                   1.465
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               30.750
Skew:                           0.219   Prob(JB):                     2.10e-07
Kurtosis:                       4.233   Cond. No.                         3.84
==============================================================================

Notes:
[1] Standard Errors are heteroscedasticity and autocorrelation robust (HAC) using 12 lags and with small sample correction
Regression of ETHUSD on SP500 using robust standard errors:
                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_ETHUSD   R-squared:                       0.155
Model:                            OLS   Adj. R-squared:                  0.153
Method:                 Least Squares   F-statistic:                     54.01
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           1.02e-12
Time:                        22:07:25   Log-Likelihood:                -1646.3
No. Observations:                 431   AIC:                             3297.
Df Residuals:                     429   BIC:                             3305.
Df Model:                           1
Covariance Type:                  HAC
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      2.3827      0.939      2.537      0.011       0.542       4.224
ex_ret_SP500   1.8107      0.246      7.349      0.000       1.328       2.294
==============================================================================
Omnibus:                       68.837   Durbin-Watson:                   1.389
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              248.185
Skew:                           0.674   Prob(JB):                     1.28e-54
Kurtosis:                       6.464   Cond. No.                         3.84
==============================================================================

Notes:
```

```
[1] Standard Errors are heteroscedasticity and autocorrelation robust (HAC) usi
ng 12 lags and with small sample correction
```

In [13]:
```
jarque_bera(robust_BTC_result, "BTC")
jarque_bera(robust_ETH_result, "ETH")
```

Results of Jarque-Bera normality test for BTC below:

JB Statistic	30.75
Chi-Squared	2.10239e-07
Skewness	0.218542
Kurtosis	4.23339

Results of Jarque-Bera normality test for ETH below:

JB Statistic	248.185
Chi-Squared	1.28047e-54
Skewness	0.674068
Kurtosis	6.46447

In [14]:
```
breusch_godfrey(robust_BTC_result, "BTC")
breusch_godfrey(robust_ETH_result, "ETH")
```

Results of the Breusch-Godfrey auto-correlation test for BTC below:

LM Statistic	39.001
LM P-Value	0
F Statistic	3.457
F Statistic P-Value	0

Results of the Breusch-Godfrey auto-correlation test for ETH below:

LM Statistic	51.761
LM P-Value	0
F Statistic	4.743
F Statistic P-Value	0

In [15]:
```
white(robust_BTC_result, "BTC")
white(robust_ETH_result, "ETH")
```

Results of the White Heteroskedasticty test for BTC below:

LM Statistic	2.11392
LM P-Value	0.34751
F Statistic	1.05478
F Statistic P-Value	0.349174

Results of the White Heteroskedasticty test for ETH below:

LM Statistic	0.351172
LM P-Value	0.838965
F Statistic	0.174506
F Statistic P-Value	0.839931

In [16]:
```
breusch_pagan(robust_BTC_result, "BTC")
breusch_pagan(robust_ETH_result, "ETH")
```

Results of the Breusch-Pagan Heteroskedasticty test for BTC below:

LM Statistic	0.734752
LM P-Value	0.391347
F Statistic	0.732591
F Statistic P-Value	0.392522

Results of the Breusch-Pagan Heteroskedasticty test for ETH below:

LM Statistic	0.351153
LM P-Value	0.553461
F Statistic	0.349809
F Statistic P-Value	0.554533

In [17]:
```
goldfeld_quandt(robust_BTC_result, "BTC")
goldfeld_quandt(robust_ETH_result, "ETH")
```

```
Results of the Goldfeld-Quandt test for BTC below:
╒═══════════════╤═══════════╕
│ F-statistic   │ 0.633351  │
├───────────────┼───────────┤
│ p-value       │ 0.999547  │
╘═══════════════╧═══════════╛
Results of the Goldfeld-Quandt test for ETH below:
╒═══════════════╤═══════════╕
│ F-statistic   │ 0.458033  │
├───────────────┼───────────┤
│ p-value       │ 1         │
╘═══════════════╧═══════════╛
```

E)

In [18]:
```
hypotheses = "ex_ret_SP500 = 1"

for model in [btc_regress, eth_regress]:
    t_stat = model.t_test(hypotheses)
    print(t_stat)
```

```
                             Test for Constraints
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
c0             1.2409      0.149      1.616      0.107       0.948       1.534
==============================================================================
                             Test for Constraints
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
c0             1.8107      0.204      3.978      0.000       1.410       2.211
==============================================================================
```

F)

In [19]:
```
observations = len(master_df)
midpoint = observations// 2
subset_1= master_df.loc[:master_df.index[midpoint]]
subset_2 = master_df.loc[master_df.index[midpoint]:]
subset_2.head()
```

Out[19]:

Date	SP500	log_ret_SP500	BTCUSD	log_ret_BTCUSD	ETHUSD	log_ret_ETHUSD	EFFR
2020-07-19	3203.920	1.177534	9193.980000	-1.159865	237.065714	-2.070070	0.094
2020-07-26	3247.290	1.344574	9560.450000	3.908588	274.094286	14.513478	0.094
2020-08-02	3246.726	-0.017370	11218.565714	15.993526	343.425714	22.549867	0.100
2020-08-09	3325.866	2.408299	11572.792857	3.108684	391.801429	13.178432	0.100
2020-08-16	3364.158	1.144762	11747.844286	1.501286	413.475714	5.384363	0.100

In [20]:
```
subset_1_BTC = regression(subset_1, "BTCUSD")
subset_1_ETH = regression(subset_1, "ETHUSD")
```

```
Regression of BTCUSD on SP500:

                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_BTCUSD   R-squared:                       0.086
Model:                            OLS   Adj. R-squared:                  0.082
Method:                 Least Squares   F-statistic:                     20.14
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           1.17e-05
Time:                        22:07:25   Log-Likelihood:                -779.05
No. Observations:                 216   AIC:                             1562.
Df Residuals:                     214   BIC:                             1569.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
Intercept      1.5032      0.693      2.171      0.031       0.138       2.86
8
ex_ret_SP500   1.2884      0.287      4.488      0.000       0.723       1.85
4
==============================================================================
Omnibus:                        5.499   Durbin-Watson:                   1.431
Prob(Omnibus):                  0.064   Jarque-Bera (JB):                6.077
Skew:                           0.230   Prob(JB):                       0.0479
Kurtosis:                       3.682   Cond. No.                         2.86
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctl
y specified.

Regression of ETHUSD on SP500:

                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_ETHUSD   R-squared:                       0.103
Model:                            OLS   Adj. R-squared:                  0.099
Method:                 Least Squares   F-statistic:                     24.51
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           1.50e-06
Time:                        22:07:25   Log-Likelihood:                -858.67
No. Observations:                 216   AIC:                             1721.
Df Residuals:                     214   BIC:                             1728.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
Intercept      2.3746      1.001      2.372      0.019       0.401       4.34
8
ex_ret_SP500   2.0547      0.415      4.951      0.000       1.237       2.87
3
==============================================================================
Omnibus:                       41.858   Durbin-Watson:                   1.284
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               91.222
Skew:                           0.909   Prob(JB):                     1.55e-20
Kurtosis:                       5.613   Cond. No.                         2.86
```

```
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctl
y specified.
```

In [21]:
```
subset_2_BTC = regression(subset_2, "BTCUSD")
subset_2_ETH = regression(subset_2, "ETHUSD")
```

```
Regression of BTCUSD on SP500:

                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_BTCUSD   R-squared:                       0.193
Model:                            OLS   Adj. R-squared:                  0.190
Method:                 Least Squares   F-statistic:                     51.28
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           1.27e-11
Time:                        22:07:25   Log-Likelihood:                -730.12
No. Observations:                 216   AIC:                             1464.
Df Residuals:                     214   BIC:                             1471.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
Intercept        1.0197      0.613      1.664      0.097      -0.188       2.22
7
ex_ret_SP500     1.1898      0.166      7.161      0.000       0.862       1.51
7
==============================================================================
Omnibus:                       13.989   Durbin-Watson:                   1.522
Prob(Omnibus):                  0.001   Jarque-Bera (JB):               33.935
Skew:                           0.178   Prob(JB):                     4.28e-08
Kurtosis:                       4.909   Cond. No.                         4.78
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctl
y specified.

Regression of ETHUSD on SP500:

                            OLS Regression Results
==============================================================================
Dep. Variable:          ex_ret_ETHUSD   R-squared:                       0.246
Model:                            OLS   Adj. R-squared:                  0.243
Method:                 Least Squares   F-statistic:                     69.88
Date:                Sun, 13 Oct 2024   Prob (F-statistic):           8.00e-15
Time:                        22:07:25   Log-Likelihood:                -774.71
No. Observations:                 216   AIC:                             1553.
Df Residuals:                     214   BIC:                             1560.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
Intercept        2.4083      0.753      3.198      0.002       0.924       3.89
3
ex_ret_SP500     1.7073      0.204      8.360      0.000       1.305       2.11
0
==============================================================================
Omnibus:                       23.192   Durbin-Watson:                   1.620
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              107.397
Skew:                          -0.025   Prob(JB):                     4.78e-24
Kurtosis:                       6.454   Cond. No.                         4.78
```

```
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctl
y specified.
```

In [22]:
```
def chow(model_1, model_2,orig_model):

    param_s1 = model_1.params
    param_s2 = model_2.params

    n1 = len(subset_1)
    n2 = len(subset_2)
    k=len(param_s1)

    residuals0 = orig_model.resid
    residuals1 = model_1.resid
    residuals2 = model_2.resid

    ssr0 = (residuals0**2).sum()
    ssr1 = (residuals1**2).sum()
    ssr2 = (residuals2**2).sum()

    numerator = (ssr0 - (ssr1 + ssr2))/  k
    denominator = (ssr1 + ssr2) / (n1 + n2 - 2*k)
    chow_statistic = numerator / denominator


   # Calculate the p-value for the Chow test
    p_value = 1 - sts.f.cdf(chow_statistic, k, n1 + n2 - 2*k)

    return chow_statistic, p_value

chow_statistic, p_value = chow(subset_1_BTC, subset_2_BTC, btc_regress)

if p_value < 0.05:
    result = "Reject the null hypothesis: There is a significant difference in c
else:
    result = "Fail to reject the null hypothesis: No evidence of a significant d

labels = ["Chow Test Statistic", "p-value:"]
table = tabulate(zip(labels, [chow_statistic, p_value]), tablefmt="fancy_grid")

print(table)
print(result)
```

```
╒═══════════════════════╤═══════════╕
│ Chow Test Statistic   │ 0.0192839 │
├───────────────────────┼───────────┤
│ p-value:              │ 0.980902  │
╘═══════════════════════╧═══════════╛
Fail to reject the null hypothesis: No evidence of a significant difference in
coefficients between the two periods, indicating stability.
```

In [23]:
```
chow_statistic, p_value = chow(subset_1_ETH, subset_2_ETH, eth_regress)

if p_value < 0.05:
    result = "Reject the null hypothesis: There is a significant difference in c
else:
    result = "Fail to reject the null hypothesis: No evidence of a significant d

labels = ["Chow Test Statistic", "p-value:"]
table = tabulate(zip(labels, [chow_statistic, p_value]), tablefmt="fancy_grid")

print(table)
print(result)
```

```
╒═══════════════════╤══════════╕
│ Chow Test Statistic │ 0.276131 │
├───────────────────┼──────────┤
│ p-value:          │ 0.758848 │
╘═══════════════════╧══════════╛
Fail to reject the null hypothesis: No evidence of a significant difference in
coefficients between the two periods, indicating stability.
```

G)

In [24]:
```
btc_regress_no_const = regression(master_df, "BTCUSD", constant = False)
eth_regress_no_const = regression(master_df, "ETHUSD", constant = False)

#Add tests
```

```
Regression of BTCUSD on SP500:

                                 OLS Regression Results
==============================================================================
========
Dep. Variable:          ex_ret_BTCUSD   R-squared (uncentered):
0.131
Model:                            OLS   Adj. R-squared (uncentered):
0.129
Method:                 Least Squares   F-statistic:
64.73
Date:                Sun, 13 Oct 2024   Prob (F-statistic):
8.44e-15
Time:                        22:07:25   Log-Likelihood:
-1515.6
No. Observations:                 431   AIC:
3033.
Df Residuals:                     430   BIC:
3037.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
=
                    coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
ex_ret_SP500     1.0133      0.126      8.046      0.000       0.766       1.26
1
==============================================================================
Omnibus:                       16.895   Durbin-Watson:                   1.440
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               32.868
Skew:                           0.199   Prob(JB):                     7.29e-08
Kurtosis:                       4.293   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not cont
ain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctl
y specified.

Regression of ETHUSD on SP500:

                                 OLS Regression Results
==============================================================================
========
Dep. Variable:          ex_ret_ETHUSD   R-squared (uncentered):
0.131
Model:                            OLS   Adj. R-squared (uncentered):
0.129
Method:                 Least Squares   F-statistic:
64.66
Date:                Sun, 13 Oct 2024   Prob (F-statistic):
8.72e-15
Time:                        22:07:25   Log-Likelihood:
-1653.2
No. Observations:                 431   AIC:
3308.
Df Residuals:                     430   BIC:
3312.
Df Model:                           1
```

```
Covariance Type:            nonrobust
==============================================================================
=
                   coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
ex_ret_SP500     1.3937      0.173      8.041      0.000       1.053       1.73
4
==============================================================================
Omnibus:                       70.303   Durbin-Watson:                   1.359
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              259.329
Skew:                           0.683   Prob(JB):                     4.87e-57
Kurtosis:                       6.546   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not cont
ain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctl
y specified.
```

In [25]:
```
goldfeld_quandt(btc_regress_no_const,"BTCUSD")
goldfeld_quandt(eth_regress_no_const,"ETHUSD")
```

```
Results of the Goldfeld-Quandt test for BTCUSD below:
```

F-statistic	0.6273
p-value	0.999657

```
Results of the Goldfeld-Quandt test for ETHUSD below:
```

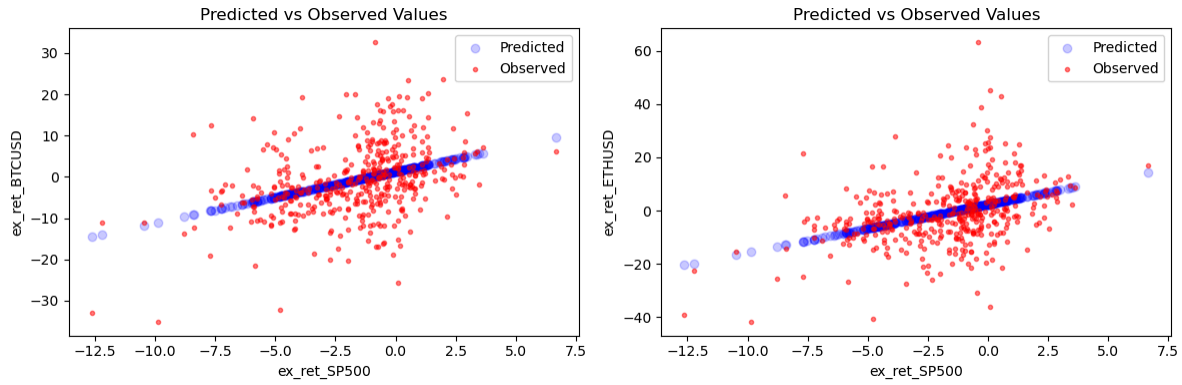
F-statistic	0.467231
p-value	1

Plotting

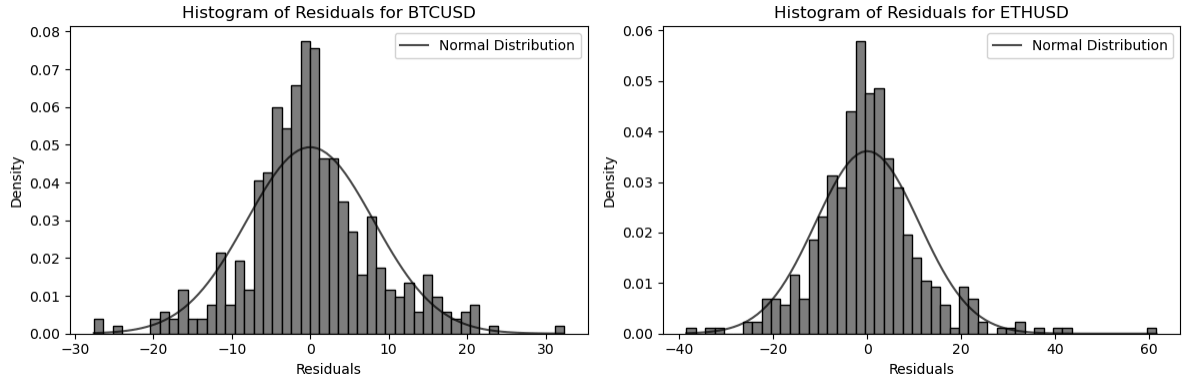
In [26]:
```
#Excess Returns
matplotlib.rcParams['figure.figsize']=[14,6]

plt.xlabel('Date')
plt.ylabel('Excess Returns')
plt.plot(master_df['ex_ret_SP500'],'k-',label='Excess Return of SP500')
plt.plot(master_df['ex_ret_BTCUSD'],'r-', alpha = 0.5, label='Excess Return of B
plt.plot(master_df['ex_ret_ETHUSD'],'y-', alpha = 0.5, label='Excess Return of E
plt.title("Excess Returns of SP500, BTC and ETH")
# Set the legend
plt.legend(loc='lower right')

# Save the figure
plt.savefig('Excess returns.jpg',bbox_inches='tight')

# Display the figure
plt.show()
```

In [27]:
```
def plot_residuals(df,reg : list, plot_type,crypto:list = ["BTCUSD","ETHUSD"]):

    # Set up 2 subplots
    fig, ax = plt.subplots(nrows = 1, ncols= 2, figsize = (12,4))


    col = 0
    for coin in crypto:

        df["resid_"+coin] = reg[col].resid
        if plot_type == "scatter":
            ax[col].scatter(df["ex_ret_SP500"],reg[col].predict(), alpha = 0.2,c
            ax[col].scatter(df["ex_ret_SP500"], df["ex_ret_"+ coin], alpha= 0.5,

            ax[col].legend()
            ax[col].set_title("Predicted vs Observed Values")
            ax[col].set_xlabel("ex_ret_SP500")
            ax[col].set_ylabel("ex_ret_"+coin)

        elif plot_type == "hist":

            ax[col].hist(df["resid_"+coin], bins=50,color = "gray",edgecolor = "

            mu = df["resid_"+coin].mean()
            std = df["resid_"+coin].std()
            x = np.linspace(df["resid_"+coin].min(), df["resid_"+coin].max(), 10
            dens_fct = sts.norm.pdf(x, mu, std)
            ax[ col].plot(x, dens_fct, color='black', alpha =0.7, label='Normal

            ax[ col].set_title(f'Histogram of Residuals for {coin}')
            ax[ col].set_xlabel('Residuals')
            ax[ col].set_ylabel('Density')
            ax[ col].legend()

        # Indexing to change col value from 0 to 1
        col = col +1

    plt.tight_layout()

    plt.savefig('Residuals.jpg',bbox_inches='tight')
    plt.show()
```

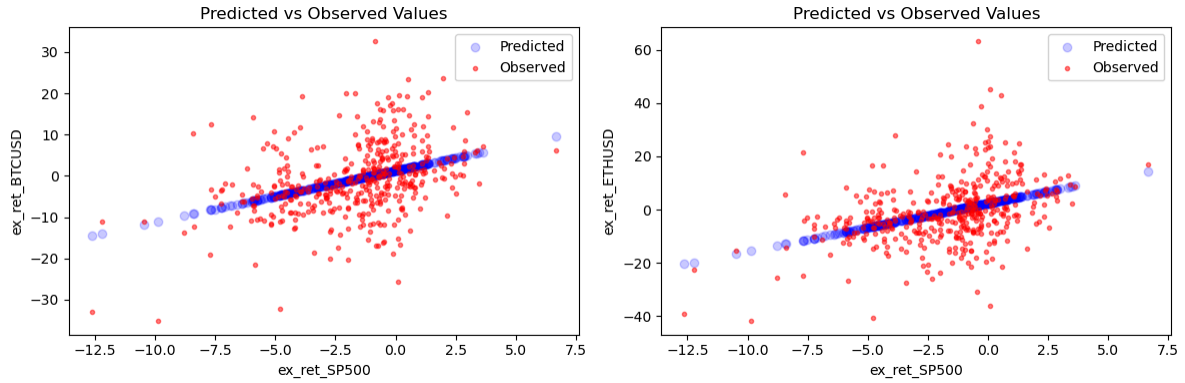
In [28]:
```
# Non - Robust Regression
plot_residuals(master_df, plot_type = "scatter",reg = [btc_regress,eth_regress])
```



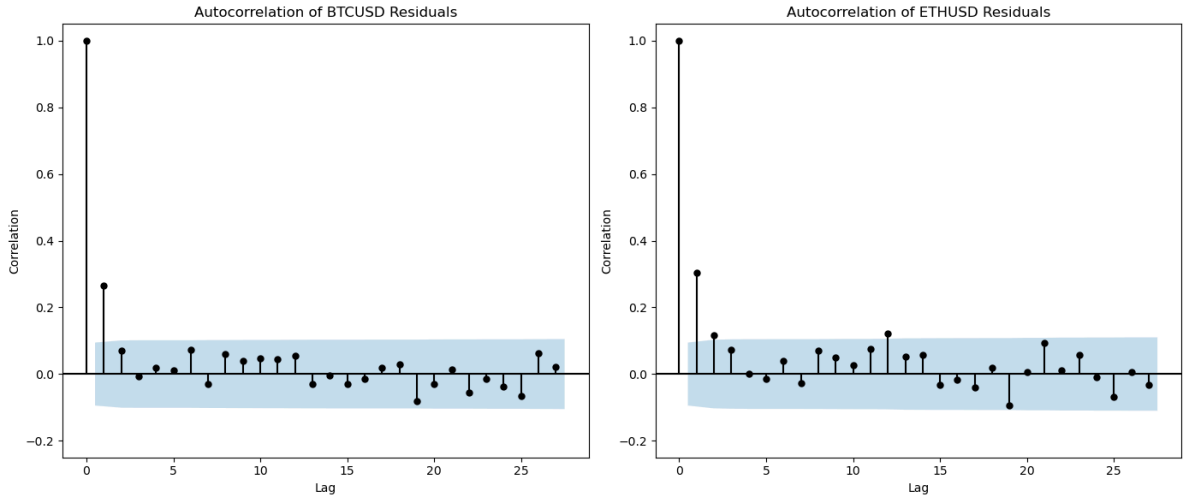
In [29]:
```
plot_residuals(master_df, plot_type = "hist",reg = [btc_regress,eth_regress])
```



In [30]:
```
# Robust Regression
plot_residuals(master_df, plot_type = "scatter",reg = [robust_BTC_result,robust_
```



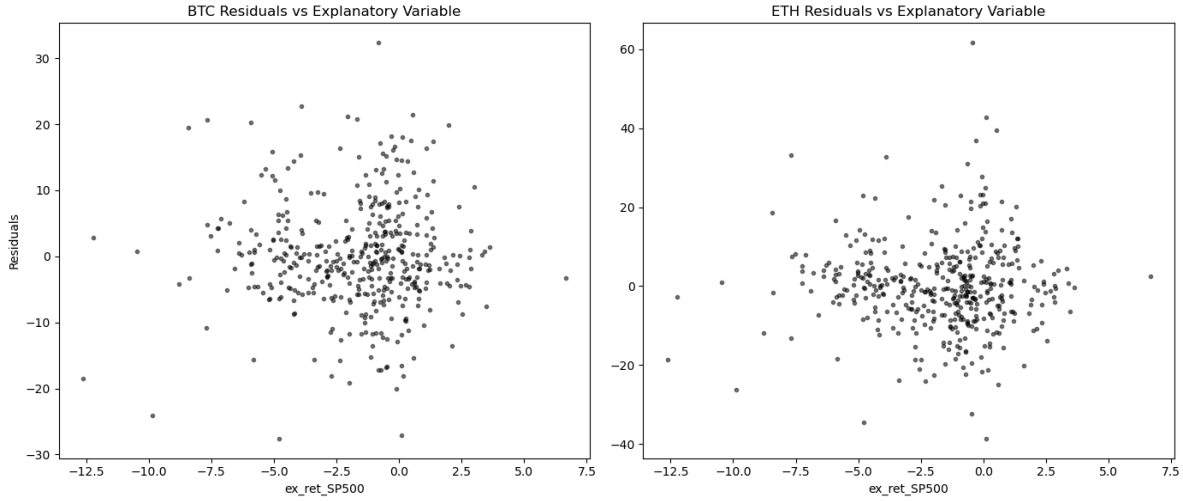
In [31]:
```
def ac_plot(crypto_1, crypto_2, reg_1,reg_2,colour):

    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 6))

    resid_1 = reg_1.resid
    smp.graphics.tsa.plot_acf(resid_1,
                                  vlines_kwargs={'colors': colour},
                                  ax=ax1,
                              color=colour)
    ax1.set_ylim((-0.25, 1.05))
    ax1.set_title(f'Autocorrelation of {crypto_1} Residuals')
    ax1.set_ylabel('Correlation')
    ax1.set_xlabel('Lag')

    resid_2 = reg_2.resid

    smp.graphics.tsa.plot_acf(resid_2,
                              vlines_kwargs={'colors': colour},
                                  ax=ax2,
                              color=colour)
    ax2.set_ylim((-0.25, 1.05))
    ax2.set_title(f'Autocorrelation of {crypto_2} Residuals')
    ax2.set_ylabel('Correlation')
    ax2.set_xlabel('Lag')



    plt.tight_layout()

    plt.savefig('Autocorrelation.jpg',bbox_inches='tight')
    plt.show()

ac_plot("BTCUSD","ETHUSD", btc_regress, eth_regress, "black")
```



In [32]:
```
def scatter(reg, x_vals,title,df = master_df):

    matplotlib.rcParams['figure.figsize']=[10,6]
    # Linearity between dependent & independent variable
    plt.scatter(df[x_vals], reg.resid,marker=".", c = "black",alpha=0.5)
    # Save the figure
    plt.savefig(f'{title}.jpg',bbox_inches='tight')
    plt.title(title)
    plt.xlabel(x_vals)
    plt.ylabel("Residuals")
```

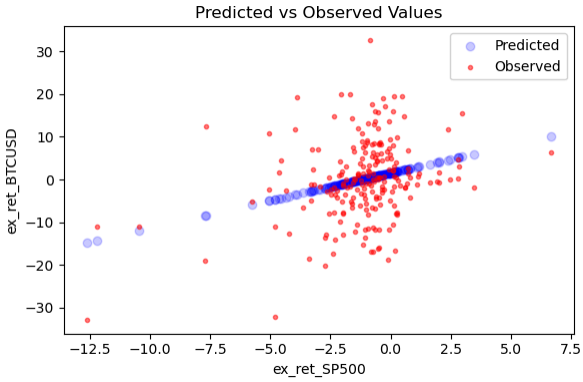
In [33]:
```
def scatter(reg1, reg2, x_vals, title1, title2, df = master_df):

    # Create the subplots, 1 row and 2 columns
    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 6))

    # First scatter plot (left subplot)
    ax1.scatter(df[x_vals], reg1.resid, marker=".", c="black", alpha=0.5)
    ax1.set_title(title1)
    ax1.set_xlabel(x_vals)
    ax1.set_ylabel("Residuals")

    # Second scatter plot (right subplot)
    ax2.scatter(df[x_vals], reg2.resid, marker=".", c="black", alpha=0.5)
    ax2.set_title(title2)
    ax2.set_xlabel(x_vals)

    # Save the figure
    plt.savefig(f'{title1}_{title2}.jpg', bbox_inches='tight')

    # Show the plot
    plt.tight_layout()
    plt.show()
```

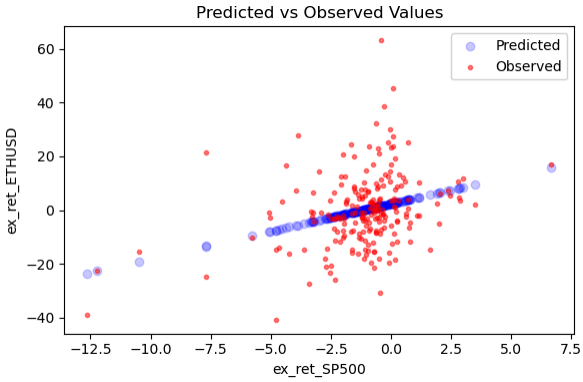
In [34]:
```
scatter(btc_regress,eth_regress, "ex_ret_SP500", "BTC Residuals vs Explanatory V
        "ETH Residuals vs Explanatory Variable")
```



In [35]:
```
# Subset 1 plotting
plot_residuals(subset_1, plot_type = "scatter",reg = [subset_1_BTC,subset_1_ETH]
```

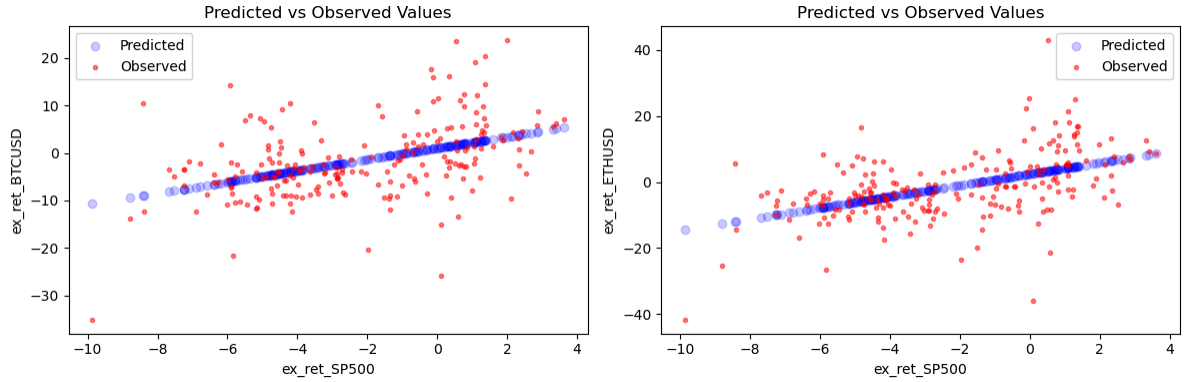
```
C:\Users\rorym\AppData\Local\Temp\ipykernel_22740\3118363465.py:10: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  df["resid_"+coin] = reg[col].resid
C:\Users\rorym\AppData\Local\Temp\ipykernel_22740\3118363465.py:10: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  df["resid_"+coin] = reg[col].resid
```



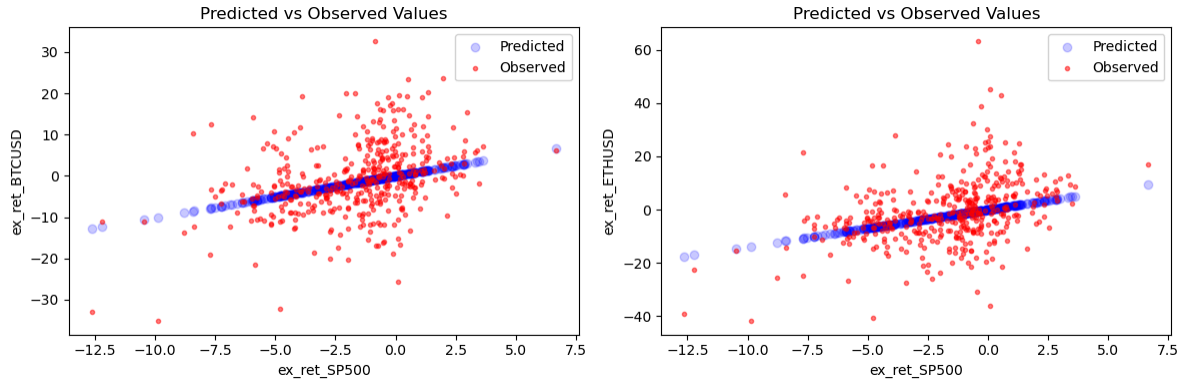


In [36]:
```
#Subset 2 plotting
plot_residuals(subset_2, plot_type = "scatter",reg = [subset_2_BTC,subset_2_ETH]
```

```
C:\Users\rorym\AppData\Local\Temp\ipykernel_22740\3118363465.py:10: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  df["resid_"+coin] = reg[col].resid
C:\Users\rorym\AppData\Local\Temp\ipykernel_22740\3118363465.py:10: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  df["resid_"+coin] = reg[col].resid
```



In [37]:
```
# No Constant
plot_residuals(master_df, plot_type = "scatter",reg = [btc_regress_no_const,eth_
```



Regression Summary Tables

In [38]:
```
def get_regression_summary(model):
    summary = model.summary()
    table1 = summary.tables[0]
    table2 = summary.tables[1]
    table3 = summary.tables[2]
    return {
        "Intercept": table2.data[1][1],
        "SE (intercept)":table2.data[1][2],
        "SP500 Coefficient": table2.data[2][1],
        "SE (SP500)":table2.data[2][2],
        "p-value (SP500)": table2.data[2][4],
        "R-squared": table1.data[0][3],
        "Adjusted R-squared":table1.data[1][3],
        "F-statistic": table1.data[2][3],
        "Prob (F-statistic)": table1.data[3][3],
        "AIC": table1.data[5][3],
        "BIC": table1.data[6][3],
        "Jarque-Bera" : table3.data[1][3],
        "Prob (JB)" : table3.data[2][3],
        "Durbin-Watson": table3.data[0][3],
        "Skewness": table3.data[2][1],
        "Kurtosis": table3.data[3][1],
    }

def get_no_const_regression_summary(model):
    summary = model.summary()
    table1 = summary.tables[0]
    table2 = summary.tables[1]
    table3 = summary.tables[2]
    return {
        "SP500 Coefficient": table2.data[1][1],
        "SE (SP500)":table2.data[1][2],
        "p-value (SP500)": table2.data[1][4],
        "R-squared": table1.data[0][3],
        "Adjusted R-squared":table1.data[1][3],
        "F-statistic": table1.data[2][3],
        "Prob (F-statistic)": table1.data[3][3],
        "AIC": table1.data[5][3],
        "BIC": table1.data[6][3],
        "Jarque-Bera" : table3.data[1][3],
        "Prob (JB)" : table3.data[2][3],
        "Durbin-Watson": table3.data[0][3],
        "Skewness": table3.data[2][1],
        "Kurtosis": table3.data[3][1],
    }
```

In [39]:
```
# Summary Table for Non-Robust regression
summary_eth = get_regression_summary(btc_regress)
summary_btc = get_regression_summary(eth_regress)

# Combine summaries into a DataFrame
summary_df = pd.DataFrame([summary_eth, summary_btc])
summary_df = summary_df.transpose()

summary_df.reset_index(inplace=True)
summary_df.columns = ['Model', 'BTC','ETH']

# Set model as the index
summary_df.set_index('Model', inplace=True)

# Create Table
table = tabulate(summary_df, headers='keys', tablefmt='fancy_grid',stralign='cen
print(table)
```

Model	BTC	ETH
Intercept	1.3006	2.3827
SE (intercept)	0.465	0.636
SP500 Coefficient	1.2409	1.8107
SE (SP500)	0.149	0.204
p-value (SP500)	0	0
R-squared	0.139	0.155
Adjusted R-squared	0.137	0.153
F-statistic	69.24	78.95
Prob (F-statistic)	1.17e-15	1.77e-17
AIC	3027	3297
BIC	3035	3305
Jarque-Bera	30.75	248.185
Prob (JB)	2.1e-07	1.28e-54
Durbin-Watson	1.465	1.389
Skewness	0.219	0.674
Kurtosis	4.233	6.464

In [40]:
```
# Summary Table for Robust regression
summary_eth = get_regression_summary(robust_BTC_result)
summary_btc = get_regression_summary(robust_ETH_result)

# Combine summaries into a DataFrame
summary_df = pd.DataFrame([summary_eth, summary_btc])
summary_df = summary_df.transpose()
# Set the "Model" as the index
summary_df.reset_index(inplace=True)

# Rename the columns for clarity
summary_df.columns = ['Robust Model', 'BTC','ETH']

# Now set 'Model' as the index
summary_df.set_index('Robust Model', inplace=True)

# Create a formal table using 'tabulate'
table = tabulate(summary_df, headers='keys', tablefmt='fancy_grid',stralign='cen

# Print the formatted table
print(table)
```

Robust Model	BTC	ETH
Intercept	1.3006	2.3827
SE (intercept)	0.621	0.939
SP500 Coefficient	1.2409	1.8107
SE (SP500)	0.18	0.246
p-value (SP500)	0	0
R-squared	0.139	0.155
Adjusted R-squared	0.137	0.153
F-statistic	47.48	54.01
Prob (F-statistic)	1.99e-11	1.02e-12
AIC	3027	3297
BIC	3035	3305
Jarque-Bera	30.75	248.185
Prob (JB)	2.1e-07	1.28e-54
Durbin-Watson	1.465	1.389
Skewness	0.219	0.674
Kurtosis	4.233	6.464

In [41]:
```
#BTC Subset Regression Table
summary_btc_sample_1 = get_regression_summary(subset_1_BTC)
summary_btc_sample_2 = get_regression_summary(subset_2_BTC)

# Combine summaries into a DataFrame
summary_df = pd.DataFrame([summary_btc_sample_1, summary_btc_sample_2])
summary_df = summary_df.transpose()

summary_df.reset_index(inplace=True)
summary_df.columns = ["BTC","Subset 1","Subset 2"]

# Set model as the index
summary_df.set_index('BTC', inplace=True)

# Create Table
table = tabulate(summary_df, headers='keys', tablefmt='fancy_grid',stralign='cen
print(table)
```

BTC	Subset 1	Subset 2
Intercept	1.5032	1.0197
SE (intercept)	0.693	0.613
SP500 Coefficient	1.2884	1.1898
SE (SP500)	0.287	0.166
p-value (SP500)	0	0
R-squared	0.086	0.193
Adjusted R-squared	0.082	0.19
F-statistic	20.14	51.28
Prob (F-statistic)	1.17e-05	1.27e-11
AIC	1562	1464
BIC	1569	1471
Jarque-Bera	6.077	33.935
Prob (JB)	0.0479	4.28e-08
Durbin-Watson	1.431	1.522
Skewness	0.23	0.178
Kurtosis	3.682	4.909

In [42]:
```
#ETH Subset Regression Table
summary_eth_sample_1 = get_regression_summary(subset_1_ETH)
summary_eth_sample_2 = get_regression_summary(subset_2_ETH)

# Combine summaries into a DataFrame
summary_df = pd.DataFrame([summary_eth_sample_1, summary_eth_sample_2])
summary_df = summary_df.transpose()

summary_df.reset_index(inplace=True)

summary_df.columns = ["ETH",'Subset 1','Subset 2']

# Set model as the index
summary_df.set_index('ETH', inplace=True)

# Create Table
table = tabulate(summary_df, headers='keys', tablefmt='fancy_grid',stralign='cen
print(table)
```

ETH	Subset 1	Subset 2
Intercept	2.3746	2.4083
SE (intercept)	1.001	0.753
SP500 Coefficient	2.0547	1.7073
SE (SP500)	0.415	0.204
p-value (SP500)	0	0
R-squared	0.103	0.246
Adjusted R-squared	0.099	0.243
F-statistic	24.51	69.88
Prob (F-statistic)	1.5e-06	8e-15
AIC	1721	1553
BIC	1728	1560
Jarque-Bera	91.222	107.397
Prob (JB)	1.55e-20	4.78e-24
Durbin-Watson	1.284	1.62
Skewness	0.909	-0.025
Kurtosis	5.613	6.454

In [43]:
```
# Summary Table for No constant
summary_btc = get_no_const_regression_summary(btc_regress_no_const)
summary_eth = get_no_const_regression_summary(eth_regress_no_const)

# Combine summaries into a DataFrame
summary_df = pd.DataFrame([summary_btc, summary_eth])
summary_df = summary_df.transpose()

summary_df.reset_index(inplace=True)
summary_df.columns = ["Model",'BTC','ETH']

# Set model as the index
summary_df.set_index("Model", inplace=True)

# Create Table
table = tabulate(summary_df, headers='keys', tablefmt='fancy_grid',stralign='cen
print(table)
```

Model	BTC	ETH
SP500 Coefficient	1.0133	1.3937
SE (SP500)	0.126	0.173
p-value (SP500)	0	0
R-squared	0.131	0.131
Adjusted R-squared	0.129	0.129
F-statistic	64.73	64.66
Prob (F-statistic)	8.44e-15	8.72e-15
AIC	3033	3308
BIC	3037	3312
Jarque-Bera	32.868	259.329
Prob (JB)	7.29e-08	4.87e-57
Durbin-Watson	1.44	1.359
Skewness	0.199	0.683
Kurtosis	4.293	6.546

In [ ]: