Modelling the dynamics between cryptocurrency markets and traditional financial markets

Chan Xin Hung

December 2024

1 Introduction

The rapid rise of cryptocurrency markets has introduced increased levels of volatility and complexity to the global financial landscape. Unlike traditional financial markets, which are generally characterized by lower volatility and more predictable patterns, cryptocurrency markets exhibit extreme price fluctuations that existing economic models are not well suited for. This disparity is why we need new approaches to understand how these two market systems interact and influence one another. If we can develop models capable of mapping the volatility and behaviors of traditional markets to cryptocurrency markets, it would provide better understanding into their interdependence, helping market participants make more informed decisions.

To capture these dynamics, models like the Lotka-Volterra and Markov chains provide promising tools. The Lotka-Volterra model, traditionally used to describe predator-prey interactions in biological systems, can be adapted to represent competing and mutually influencing market forces. Similarly, Markov chains offer a way to analyze state transitions over time based on historical patterns, allowing for the prediction of market behavior under varying conditions.

2 Preparing the data

To compare the two markets, I decided to use the S&P 500 as an general indicator of how the traditional markets are doing. For the cryptocurrencies, I used the top 23 crypto currencies over the same period and calculated a similar index for it. Then I normalized both datasets so that they would be on the same scale, allowing for more accurate comparison and modeling.

Below is how I calculated the cryptocurrency index:

For each Cryptocurrency on each day, the market cap is calculated by:

$$\text{Market Cap}_i = \text{Price}_i \times \text{Circulating Supply}_i$$

So, the total market cap at time t is:

$$\text{Total Market Cap}(t) = \sum_{n=1}^{23} \text{Market Cap of Crypto}_i(t)$$

Then, using a market-cap weighted index:

$$\text{Weight}_i = \frac{\text{Market Cap}_i}{\sum_{j=1}^{n} \text{Market Cap}_j}$$

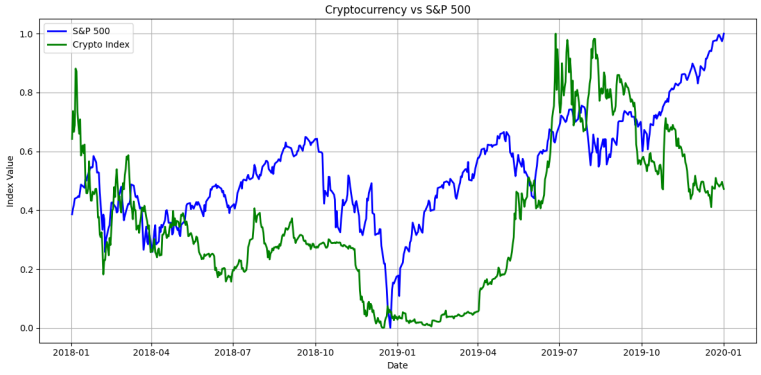
Calculating the Index Value:

$$\text{Index Value}_t = \frac{\sum_{i=1}^{n}(\text{Market Cap}_i(t) \times \text{Weight}_i)}{\sum_{i=1}^{n} \text{Market Cap}_{i,\text{Base Date}}}$$

Normalize both indexes so they go from 0 to 1:

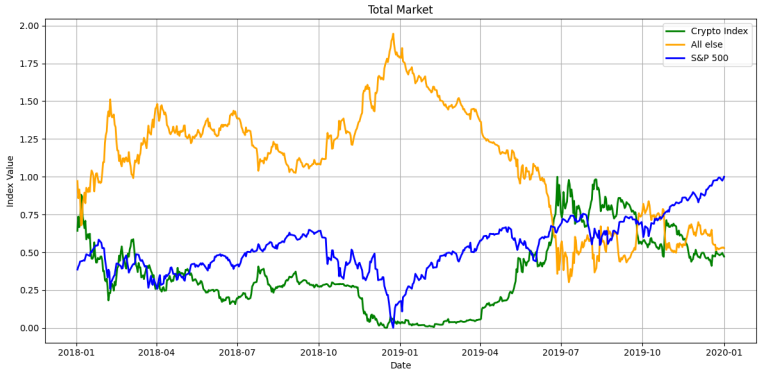
$$\text{Normalized Value} = \frac{\text{Value - Min Value}}{\text{Max Value - Min Value}}$$

Here is how both plots look like on the same graph:



For the purpose of making this a closed system, I added a new set of data points: All the capital that is not in either the cryptocurrency or traditional markets but can flow in or out of the two.

Adding the new plot on the same graph:



3 Markov Chain

Let A represent the S&P 500
Let B represent the Cryptocurency Market
Let C represent all external markets and capital

The Markov chain transition matrix P, representing the probabilities of transitioning between the three markets A, B, and C, is defined as:

$$P = \begin{bmatrix} P_{AA} & P_{AB} & P_{AC} \\ P_{BA} & P_{BB} & P_{BC} \\ P_{CA} & P_{CB} & P_{CC} \end{bmatrix}$$

Each row represents the probabilities of transitioning from one market to another and satisfies the constraint that the rows sum to 1:

$$P_{AA} + P_{AB} + P_{AC} = 1, \quad P_{BA} + P_{BB} + P_{BC} = 1, \quad P_{CA} + P_{CB} + P_{CC} = 1$$

This matrix governs the flow of capital between the three markets over time, where P_{AA} represents the flow from A back into A, P_{AB} represents the capital flow from A into B, etc.

So, the goal is to obtain a transition matrix P given X and Y in the equation $PX = Y$, where X represents a vector $\begin{bmatrix} A_i \\ B_i \\ C_i \end{bmatrix}$ and Y is the vector $\begin{bmatrix} A_{i+1} \\ B_{i+1} \\ C_{i+1} \end{bmatrix}$.

However, its not possible to solve for a 3×3 matrix P given only two 3×1 matrices X and Y, and it would be difficult to approximate the values well.

So, I created more data points in my dataset by interpolating the data in between the data points that I already have. This way I can increase the size of my dataset manyfold.

By increasing the size of my dataset, the distance between each point decreases, and therefore instead of making X the vector $\begin{bmatrix} A_i \\ B_i \\ C_i \end{bmatrix}$, we can stack three of these vectors together to make a 3×3 matrix since the datapoints are so close to each other its essentially the same as the previously defined vector X. Now we can let X be $\begin{bmatrix} A_i & A_{i+1} & A_{i+2} \\ B_i & B_{i+1} & B_{i+2} \\ C_i & C_{i+1} & C_{i+2} \end{bmatrix}$ and Y be $\begin{bmatrix} A_{i+3} & A_{i+4} & A_{i+5} \\ B_{i+3} & B_{i+4} & B_{i+5} \\ C_{i+3} & C_{i+4} & C_{i+5} \end{bmatrix}$. This way we can solve for P, or at least estimate it given some constraints.

1. Setup

Given the data for the two matrices:

- $\mathbf{X}$: A 3×3 matrix representing the current values of all three markets.
- $\mathbf{Y}$: A 3×3 matrix representing the next value of all three markets.

The goal is to solve for $\mathbf{P}$, a 3×3 transition matrix, such that:

$$\text{Minimize: } \|\mathbf{PX} - \mathbf{Y}\|_F^2$$

where $\| \cdot \|_F$ is the **Frobenius norm**.

2. Objective Function

The Frobenius norm of the difference between $\mathbf{PX}$ and $\mathbf{Y}$ is computed as:

$$\|\mathbf{PX} - \mathbf{Y}\|_F^2 = \sum_{i,j} \left((\mathbf{PX})_{ij} - \mathbf{Y}_{ij} \right)^2$$

This measures how well $\mathbf{P}$ transforms $\mathbf{X}$ into $\mathbf{Y}$.

3. Constraints

Each value of $\mathbf{P}$ must satisfy: $0 < P_{ij} < 1$

4. Optimization Process

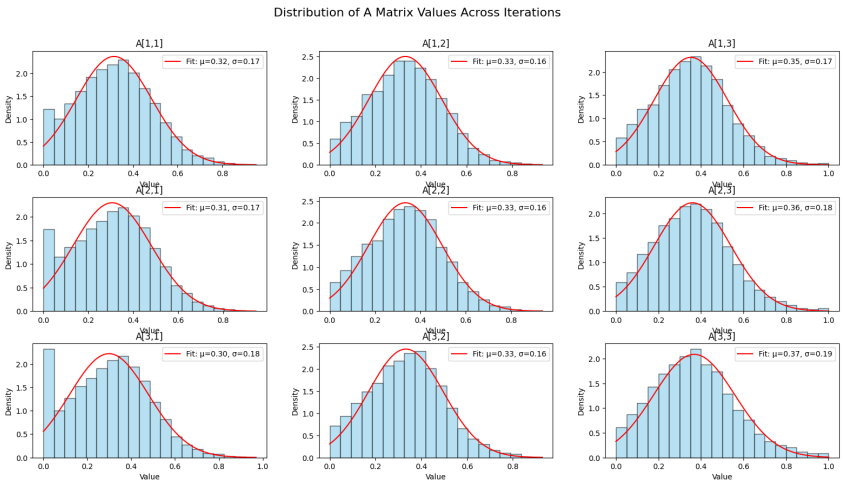
For each 6-row chunk of the data:

- An initial guess for $\mathbf{P}$ is generated randomly ($\mathbf{P}_{\text{initial}}$).
- The optimization problem is solved using the **L-BFGS-B** method under the constraints $0 < P_{ij} < 1$.

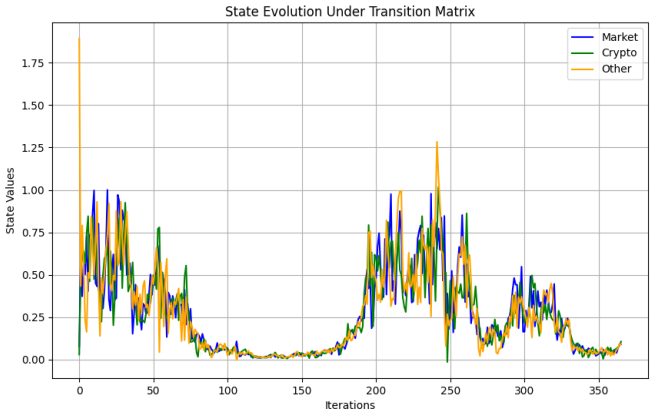
The resulting optimized $\mathbf{P}$ matrix is stored, and each value is tracked across each iteration.

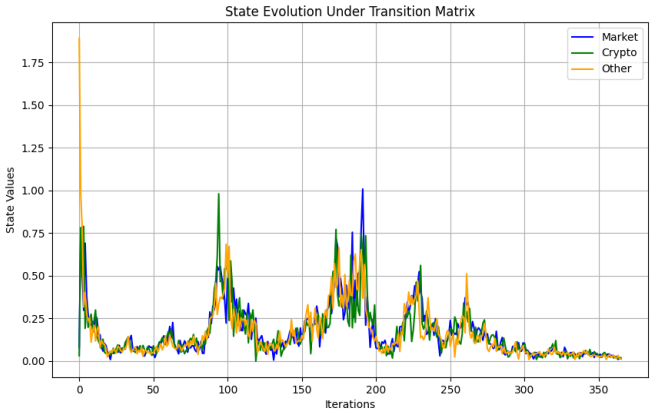
5. Results

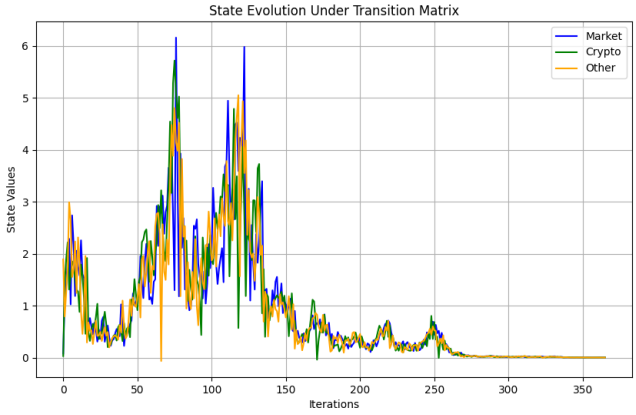
After obtaining the $\mathbf{P}$ matrices across each iteration, we can then plot the distribution of each value in $\mathbf{P}$.

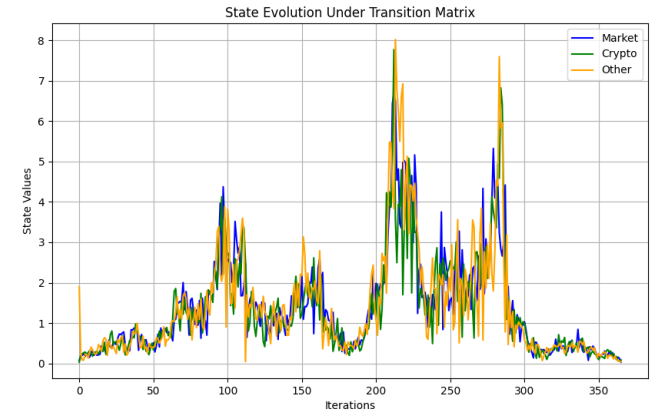


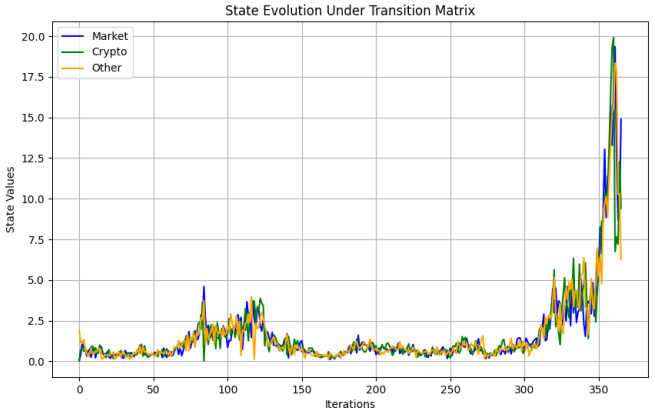
Then, to visualize the transition matrix to see if it accurately models the given data, I provide the initial values of all three markets, iterate over the transition matrix (picking the values for each cell randomly based on the distribution), and plot each resulting value.


State Evolution Under Transition Matrix
Market
Crypto
Other
State Values
Iterations


State Evolution Under Transition Matrix
Market
Crypto
Other
State Values
Iterations


State Evolution Under Transition Matrix
Market
Crypto
Other
State Values
Iterations





Since at each iteration, the transition matrix is different, the resulting plots vary quite a bit. However, after plotting the graphs, it seems evident that the complexity of capital flow cannot be simply captured and estimated by a probability distribution transition matrix.

4 Modified Lotka-Volterra Model

The Lotka-Volterra model describes predator-prey dynamics using two equations:

1. For the prey population (A):

$$\frac{dA}{dt} = \alpha A - \beta AB$$

2. For the predator population (B):

$$\frac{dB}{dt} = \delta AB - \gamma B$$

Where:

- A = prey population
- B = predator population
- α = prey growth rate
- β = the effect of the population of B on A
- δ = the effect of the population of A on B
- γ = predator death rate

The model predicts oscillations in both populations: as prey increase, predators increase, but eventually, the prey population decreases, leading to a decline in predators.

The traditional lotka volterra model does not account for external effects on the population of either populations, for this reason, in order to more accurately model the cryptocurrency vs traditional market dynamics (which is not solely affected by each other), I propose adding a new parameter to the equation that represents external effects on both markets.

Modified Lotka-Volterra Model:

$$\frac{dA}{dt} = \alpha A - \beta AB + k(2 - (A + B))$$
$$\frac{dB}{dt} = \delta AB - \gamma B - k(2 - (A + B))$$

Where:

- k = external effects on both markets

The parameter k can represent the effects of all other variables on both markets. This includes government policies and regulations, international markets, etc.

They are opposite signs on the two equations due to the assumption that government policies generally tend to be against cryptocurrencies and but supportive of traditional markets.

In this equation, I used the constant "2" because that is the total capital that can flow into or out of either markets, as defined above as the additional set of data apart from the cryptocurrency and S&P 500.

1. Approach

The approach for creating the model is as follows:

1. Input observed data: A_{obs} (cryptocurrency index) and B_{obs} (traditional market index).
2. Use an initial guess for parameters $\alpha, \beta, \delta, \gamma$, and k.
3. Minimize the objective function to find the best-fitting parameters.
4. Solve the Lotka-Volterra equations using the fitted parameters.
5. Calculate the error metrics (MSE for A and B).
6. Compare the observed data to the model predictions, and plot the curves.
7. Compare which model (Traditional or Modified Lotka-Volterra) fits better.

2. Goal

The goal is to fit the Lotka-Volterra model and the Modified Lotka-Volterra model to the observed data A_{obs} (cryptocurrency index) and B_{obs} (traditional market index). We can achieve this by minimizing the sum of squared errors between the observed data and the model predictions:

$$\text{Error} = \sum_{i=1}^{N} \left[(A_{\text{obs},i} - A_{\text{model},i})^2 + (B_{\text{obs},i} - B_{\text{model},i})^2 \right].$$

Where:

- $A_{\text{obs},i}$ and $B_{\text{obs},i}$: Observed data points at time t_i,
- $A_{\text{model},i}$ and $B_{\text{model},i}$: Predicted values from the Lotka-Volterra model.

The optimization is performed using the `minimize` function with the Powell method in the SciPy python library.

3. Error Calculation

To judge the quality of the model fit, we calculate the Mean Squared Error (MSE) for both markets:

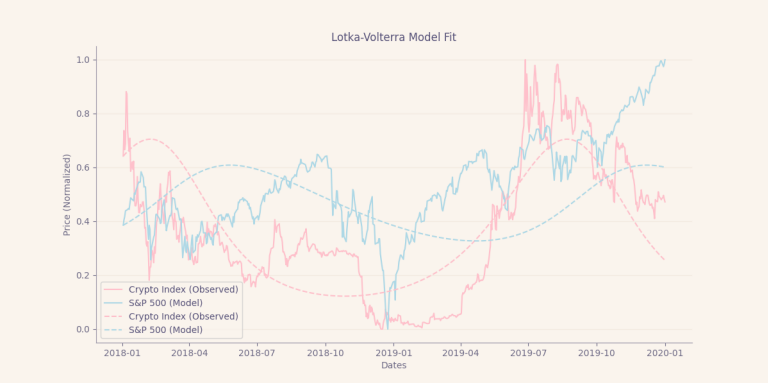
$$\text{MSE}_A = \frac{1}{N}\sum_{i=1}^{N}(A_{\text{obs},i} - A_{\text{model},i})^2, \tag{1}$$

$$\text{MSE}_B = \frac{1}{N}\sum_{i=1}^{N}(B_{\text{obs},i} - B_{\text{model},i})^2. \tag{2}$$

4. Results

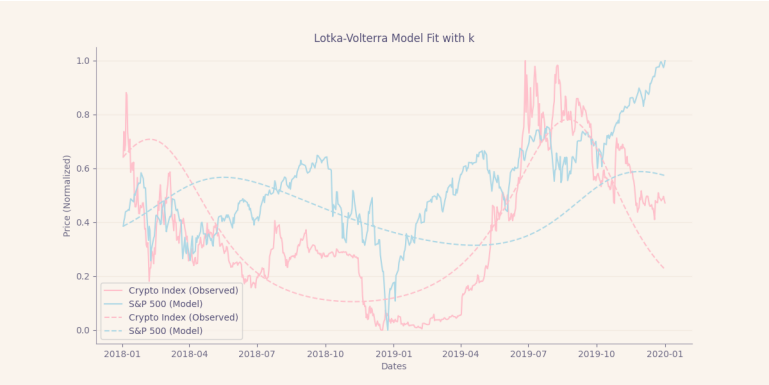
Depending on the date range, the Lotka-Volterra model would fit better sometimes, while other times the Modified Lotka-Volterra model would be a better fit.

For date ranges 2018-2020, the modified lotka-volterra model fitted better:



Error A: 0.0269008
Error B: 0.0373255

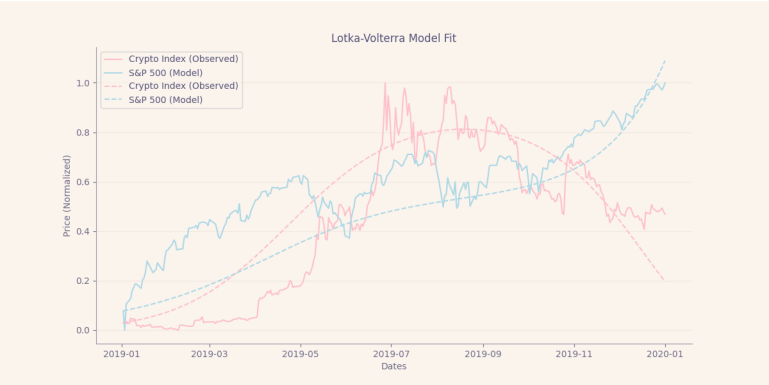
Fitted parameters: [0.03210855 , 0.07077064 , 0.01255331 , 0.0041793]



Error A: 0.024537
Error B: 0.038428

Fitted parameters:[0.03996874 , 0.08915267 , 0.01161101 , 0.00450942 , -0.00024287]

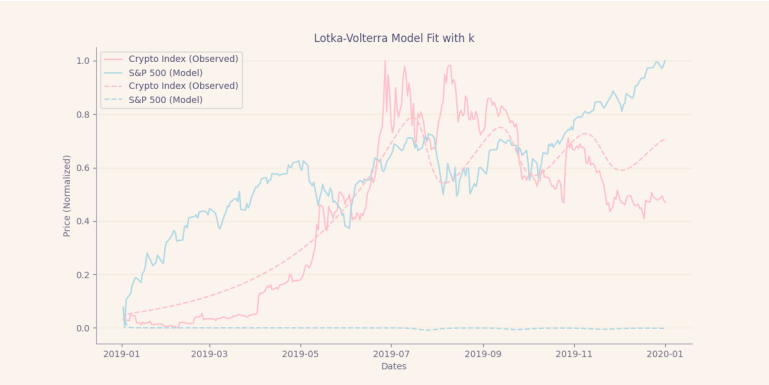
Whereas for date ranges 2019-2020, the lotka-volterra model was a better fit as compared to the modified lotka-volterra model.



Let this model be Model 1

Error A: 0.021327
Error B: 0.024439

Fitted parameters: [0.03651224 , 0.06833389 , -0.01765822 , -0.01608745]

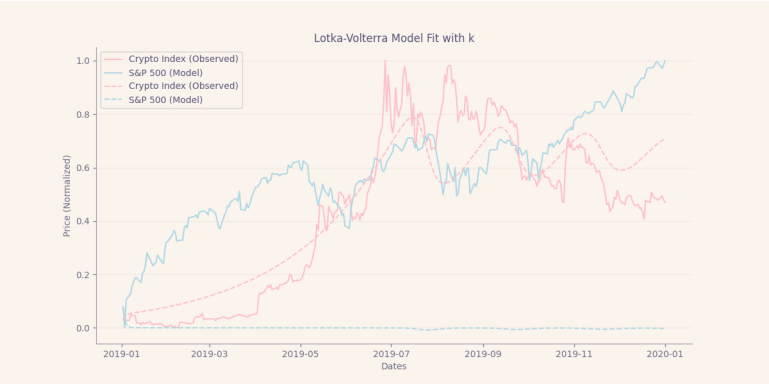


Error A: 0.014979
Error B: 0.383336

Fitted parameters: [0.0146923282 , -5.21395814 , 1.31945944 0.869535173 , 0.0000153266]

However, while running the trials for the models, I found that the initial guess for the parameters were heavily influencing how to model turned out.

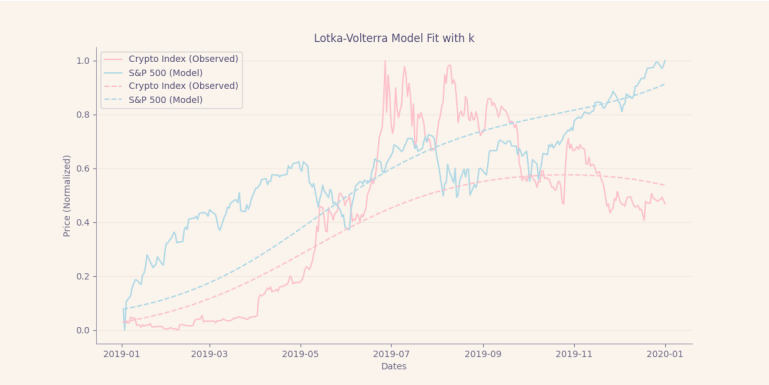
Below is a plot of two modified lotka-volterra models with slightly varying initial guesses for the parameters:



Initial Guess: [0.15, 0.008, 0.02, 0.1,0.000000000001]

Error A: 0.014979
Error B: 0.383336

Fitted parameters: [0.01469233 , -5.2139581 , 1.3194594 , 0.869535173 , 0.0000153266]



Let this be Model 2

Initial Guess: [0.15, 0.008, 0.02, 0.001,0.000000000001]

Error A: 0.029058
Error B: 0.022275

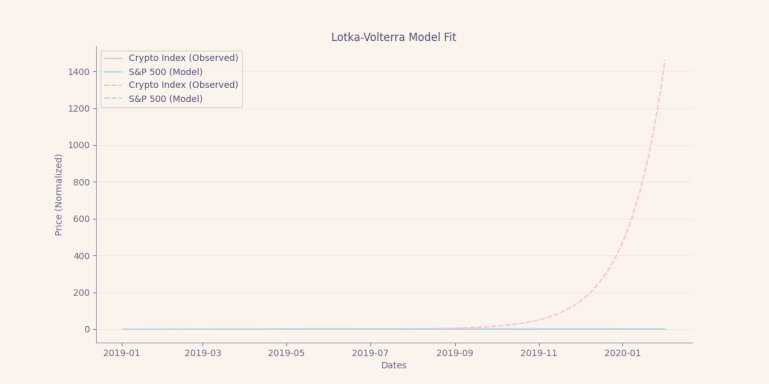
Fitted parameters: [0.01838526 , 0.0230223 , -0.0321707 , -0.02011675 , 0.00024394]

I had only changed my initial guess of gamma (γ) from 0.001 to 0.1, and all other parameters remained unchanged and yet the models had drastically changed and there is a significant drop in the error for the S&P 500 model.

5. Model Validation

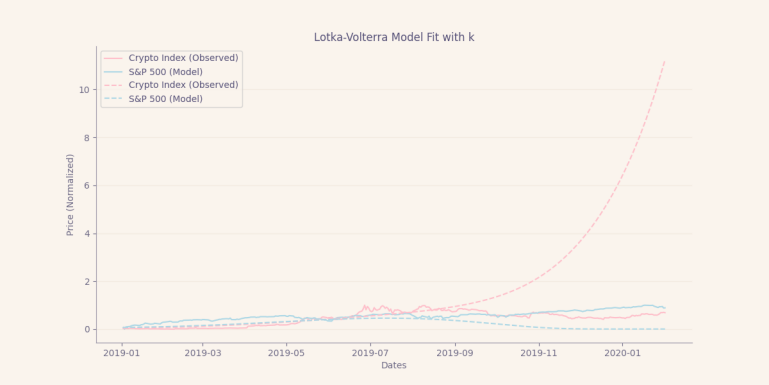
I chose the models generated from the date range 2019-2020 to validate as those two models were the ones with the least mean squared error across all the other generated models.

Testing **Model 1**, one month of data after what it was trained on.



Error A: 76489.5118 ; Error B: 0.2613

Testing **Model 2**, one month of data after what it was trained on.



Error A: 7.4348; Error B: 0.2046

From this we can see that even though both models were very inaacurate in predicting the market dynamics past the data it was trained on, the modified lotka-volterra model was significantly less erroneous than the traditional lotka-volterra model.

6. Analysis

Adding the additional k parameter to the lotka-volterra model allowed for more flexible fitting of the model and it was not constrained to a solely predator and prey relationship. Even though the mean squared error for the modified-lotka volterra model is generally higher than that of the traditional lotka-volterra model, I would argue that the shape of the modified lotka-volterra model seems to fit better despite the higher error margins. And this is further proven by the fact that the modified lotka-volterra model is less erroneous over the long run, when validating the model.

5 Conclusion

In conclusion, both the Markov Chain model and the Lotka-Volterra Models did not perform as well as expected. However, there was valuable data obtained during the process of creating the models. In the case of the Markov Chain, even though the model was not fitting well, obtaining the transition matrix is insight that we would otherwise not have had we not created the model. The distribution of values in the transition matrix, along with the mean and standard deviation allows us to understand how the capital was flowing in and out of each market over that date range. The parameters of the lotka-volterra models allow us to understand the growth and decline rate in relation to each other. Therefore, despite the fact that the models were inaccurate, there are still pieces of data and information that we obtained along the way that provides a better understanding of how traditional and cryptocurrency markets interact.

References

Kumal, Kaushal. “Machine learning in parameter estimation of nonlinear systems” arXiv. August, 2023.

Garcia, P. “Modeling systems with machine learning based differential equations” Science Direct. 2022.

https://www.kaggle.com/datasets/kaushiksuresh147/top-10-cryptocurrencies-historical-dataset

https://www.kaggle.com/datasets/paultimothymooney/stock-market-data