Impact of Liquidity Pool Size on Trading Volume in BTC-ETH Pools

Team #111
Matias Vizcaino (avizcaino3) — Walter Jack Simmons (wsimmons35) — MingShen Wang (mwang709) — Vítor de Matos Castilho (vcastilho3)

09 July 2023

Problem Statement and Objective

Decentralized Finance (DeFi), hosting about 48.78 billion USD in total value (as of April 23, 2023[1]), relies heavily on liquidity pools in decentralized exchanges like Uniswap for efficient crypto trading. These pools, using an automated market maker model, earn from trading fees distributed among liquidity providers based on their pool share.

This study primarily **aims to examine the influence of pool size on trading volume, specifically in BTC-ETH liquidity pools in exchanges like Uniswap**. It hypothesizes a direct correlation between pool size and trading volume, supported by existing studies.

This understanding could optimize liquidity provision, enhance DeFi market efficiency, and benefit stakeholders. For instance, liquidity providers could improve fee returns and risk management, traders could refine strategies, and DeFi platforms could develop more effective liquidity pools.

Uniswap Liquidity Pools: An Overview

Various DeFi platforms cater to different needs. Uniswap, a popular choice, provides a user-friendly interface and a broad spectrum of token pairs. Other platforms like Curve Finance and Balancer offer low-slippage trades and customizable pools, respectively.

Uniswap, a decentralized exchange (DEX) on Ethereum, supports trading of Ethereum-native and wrapped non-native assets. It uses a Constant Product Market Maker (CPMM) model in which liquidity providers deposit two tokens, keeping a constant product of reserves.

The CPMM formula is given by:

$$x \cdot y = k \quad \text{and} \quad price = \frac{y}{x}$$

Here, x and y denote the quantities of Token X and Y in the pool, and k is the constant product.

Uniswap operates without a central authority, ensuring privacy, fund control, and a diverse range of tokens. Centralized exchanges (CEX), though offering faster transaction speeds and better customer support, may pose security risks as they hold user funds.

Liquidity pools in DEXs enable direct interaction with smart contracts, ensuring continuous liquidity but exposing liquidity providers to impermanent loss risks. Prices are influenced by trade size, market conditions, supply-demand dynamics, and external price changes. Swaps and liquidity provision events, such as token minting and burning, significantly impact trading volumes. Indirect effects from network correlations and arbitrage opportunities can also influence trading volumes across pools.

Introduction to Techniques Used to Solve the Problem

To address the problem statement and achieve our objectives, we will employ various techniques in our analysis. These techniques are commonly used in data analytics and quantitative research to explore relationships, make predictions, and gain insights from complex datasets. In this section, we provide an overview of the key techniques we will utilize in our study.

Regression Analysis

Regression analysis is a widely used statistical technique for examining the relationship between a dependent variable and one or more independent variables. In our study, we will employ regression models, specifically Ordinary Least Squares (OLS) regression, to investigate the impact of liquidity pool size on trading volume. OLS regression allows us to estimate the coefficients of the independent variables and assess their significance in explaining the variation in the dependent variable.

Feature Engineering

Feature engineering is the process of creating new features from existing data to improve the performance of machine learning models. In our analysis, we will perform feature engineering to construct additional variables

[1] *"DefiLlama"* (2023), `https://defillama.com/`

that capture relevant information about liquidity pool dynamics, spillover effects, and price divergences. These engineered features will enhance the predictive power of our models and enable us to uncover meaningful patterns and relationships in the data.

Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an essential step in understanding and summarizing the main characteristics of a dataset. We will conduct EDA to generate descriptive statistics, visualizations, and correlation analyses. EDA will help us identify trends, outliers, and relationships between variables, providing valuable insights into the behavior of liquidity pool size and trading volume in BTC-ETH pools.

Model Evaluation and Optimization

Model evaluation and optimization are crucial steps in developing accurate and robust predictive models. We will evaluate the performance of our regression models using metrics such as R-squared, which measures the proportion of variance in the dependent variable explained by the independent variables. Through model optimization, we will refine our feature selection, address multicollinearity, and explore alternative functional forms to improve the accuracy and interpretability of the models.

These techniques, along with our structured methodology, will enable us to analyze the relationship between liquidity pool size and trading volume in BTC-ETH pools effectively. By leveraging these techniques, we aim to provide valuable insights and recommendations to enhance liquidity provision, optimize trading strategies, and improve the efficiency of DeFi marketplaces.

Approach and Progress

Overall, significant progress has been made in each stage of our approach. However, further analysis, evaluation, and optimization are required to successfully complete the project.

Throughout this process, we aim to maintain rigorous documentation of our methodologies and findings, which will allow us to ensure the transparency and replicability of our research.

Data Collection: Sourced data from multiple APIs (Uniswap, Binance) for liquidity pool sizes, trading volumes, and relevant variables. Data has been associated with Ethereum block numbers for consistent time measurement. Despite API rate limits, most of the data has been collected successfully.
Data Preprocessing: Cleaned, formatted, and addressed missing values in the dataset. Data consistency ensured and discrepancies from multiple sources addressed. Preprocessing completed with further examination of missing values in independent variables. Records with null values temporarily removed for continued analysis.
Feature Engineering: Constructed features to capture historical patterns, spillover effects, and price divergences utilizing a block-based time-series analysis framework. Feature engineering mostly complete, handling most direct pool features with finite lag. A review is planned to reduce the number of null values, include other feature categories, and perform feature selection.
Exploratory Data Analysis: Generated descriptive statistics, visualizations, and performed correlation analyses to uncover patterns, trends, and relationships between variables. In-depth exploratory data analysis will be performed once all features have been engineered, with various correlations and trends identified among the variables.
Model Selection and Development: Implemented a multivariate regression model (OLS) to investigate the pool size-trading volume relationship. Predicted target variables at different horizons with the application of lagged variables. OLS regression models have been developed and tested. Additional model testing and optimization in progress.
Model Evaluation and Optimization: Models evaluated using R-squared metric. Initial analyses indicate that model performance can be enhanced through further feature engineering/selection and reducing multicollinearity. Conducting correlation analysis and Variance Inflation Factor (VIF) for the final features.
Final Analysis and Conclusions: In the final stages of analysis. Commencing with drawing conclusions and communicating findings. Preparing to provide quantitative-driven insights.

Table 1: Approach and Progress

Challenges and Insights

Data collection and processing required significant effort due to the advanced engineering involved. Notably, we faced challenges with the rate limits of the Etherscan free-tier API when gathering data via Python API calls.

We've adopted two time concepts to analyze liquidity pool dynamics[2]. The "trading clock", defined by a block containing a mint operation, captures data temporal dynamics. A "time horizon" concept was introduced, defined

[2] *"DeFi: modeling and forecasting trading volume on Uniswap v3 liquidity pools"* (2023), `https://ssrn.com/abstract=444535`

every ten blocks up to the next mint operation, providing insights into the predictive power of independent variables over different timeframes.

Combined, these concepts allow a comprehensive exploration of short and long-term trends, informing decision-making and risk management in the DeFi ecosystem. Considering Uniswap operates on the Ethereum blockchain with a 14 seconds block time, these insights are invaluable.

The complexities of liquidity pool mathematics[3][4] presented hurdles, but we've captured the formulas to our best understanding in calculations.py functions. Given more time, we aim to refine these for enhanced feature derivation.

Dataset

Inspired by the *"DeFi modeling and forecasting trading volume" (2023)*[5] paper, we sourced and constructed trade information for at least 6 months[6]. Initial extraction work has already been performed, and you can refer to our GitHub repository[7] for further details and access to the code. The dataset consists of data obtained from the following sources:

Source	**Description**	**Data**
Uniswap's The Graph API[8]	Provides transaction details, trading volumes, and block information from Uniswap v3 liquidity pools.	Transaction IDs, timestamps, amounts, USD equivalents, and other related data.
Etherscan API[9] for Uniswap transaction hashes	Used to extract corresponding transaction data from Etherscan based on transaction hashes.	Block hashes, block numbers, sender addresses, gas details, transaction hashes, and other relevant information.
Binance[10] CEX Data for ETHBTC	Daily zip trades downloaded using provided scripts from the Binance GitHub repository.	Detailed information about each trade executed on the Binance platform, including trade prices, quantities, timestamps, and buyer/seller characteristics.

Key Variables

Target Variable: Liquidity pool trading volumes (amountUSD) over specific blocks. We consider different models for multiple time horizons to study the relationship change over time.

Independent Variables: The dependent variables are derived from a set of features and are categorized as follows [Mostly completed, refer to progress section]:

1. Direct pool features (41): volatility, rate, number of trades, average trade size.
2. CEX spillover effects (6): actual coin trade volume.

We will aim to include as many variables as possible, given the time constraints of our project.

In-Depth: Data and Feature Engineering

Feature engineering plays a crucial role in capturing the dynamics and relationships between variables in our analysis. The focus lies on engineering features that capture historical patterns of the DEX pools and spillover effects from CEX activity.

To start with, we can first define all the blocks in scope, then move to define the Reference blocks, and finally, the interval between reference blocks (which will be all the blocks in the interval).

Let's assume B is the entire sequence of blocks within the scope of analysis and K is the total number of blocks under study. We can represent this in the following way:

$$B = b_k{}_{k=1}^{K} \text{ where each } b_k \text{ is a block within the scope of analysis} \tag{1}$$

In the construction of a time-series data model, the "Reference Block" acts as a chronological reference point. These blocks are key markers within the blockchain, identified based on Mint Operations when users provide liquidity to different pools on the decentralized exchange (DEX), such as Uniswap. The Reference Blocks are categorized into "same" and "other" pools. We position the most recent at the top and with an index of 0, so the sequence of blocks is ordered in decreasing chronological order.

$$R_s = \{b_s\}_{s=1}^{S} \text{ where each } bs \in B \text{ and is a reference block in the "same" pool} \tag{2}$$

[3] *"Liquidity Math in Uniswap V3"* (2021), `https://atiselsts.github.io/pdfs/uniswap-v3-liquidity-math.pdf`

[4] *"A Primer on Uniswap v3 Math: As Easy As 1, 2, v3"* (2023), `https://blog.uniswap.org/uniswap-v3-math-primer`

[5] *"DeFi: modeling and forecasting trading volume on Uniswap v3 liquidity pools"* (2023), `https://ssrn.com/abstract=444535`

[6] *Data Extracted and Cleansed*, `https://drive.google.com/drive/folders/1y5ZwLZK9GQYsCNYSY--4VQMg8OdnuwuU?usp=sharing`

[7] GitHub repository: `https://github.gatech.edu/MGT-6203-Summer-2023-Canvas/Team-111/tree/main/Code`

$$R_o = \{b_o\}_{o=1}^{O} \text{ where each } bo \in B \text{ and is a reference block in the "other" pool} \tag{3}$$

Here, S and O are the total number of reference blocks in the "same" and "other" pools respectively.

Finally, the intervals between these reference blocks, which contain all blocks between b_i and b_j (inclusive of b_i), can be defined as the sequence of block numbers between b_i and b_j. This can be expressed as:

$$I = b_{ij}{}_{i=0,j=i+1\ i,j\leq N}^{N} \tag{4}$$

In this formula, b_{ij} represents the sequence of blocks from b_i to b_{j-1}, inclusive of b_i and exclusive of b_j. This correctly captures the notion of an interval being all blocks "between" b_i and b_j in the blockchain sequence.

Block Reference Interval chains for the "same" and "other" pools' mint operations generate sequences of block numbers. These chains include the last N reference blocks and every block in between, where N signifies the number of lags or intervals analyzed for the direct features. Let C_s and C_o be the chains of "same" and "other" pools, which can be mathematically defined as:

$$C = \{b_i\}_{i=0}^{N} \text{ where each } bi \text{ is a block} \in R_s, R_o \text{ and } b_i > b_{i+1} \text{ for } i \in 0, 1, ..., N-1 \tag{5}$$

Time horizons, representing future periods or timeframes for predictions or analyses of target variables, are constructed by starting from the first Reference Block and increasing the block count by M until the next reference block is reached.

$$H_M = \{b_0 + kM\}_{k=0}^{K} \text{ with } K = \min\left(K_{max}, \left\lfloor \frac{\text{distance to next reference block}}{M} \right\rfloor\right) \tag{6}$$

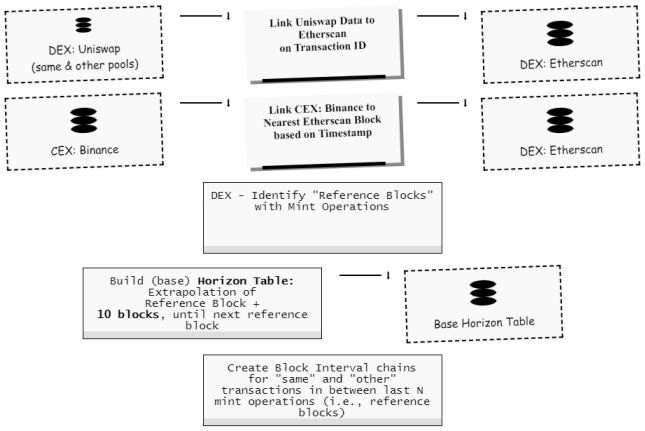

Figure 1: Data Engineering

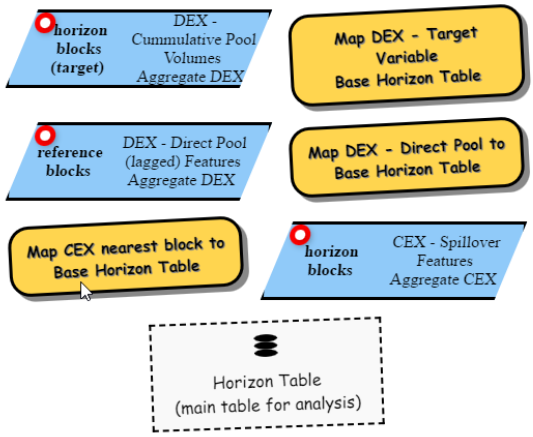

Figure 2: Feature Engineering

The Horizon Block Table captures the dynamics and relationships between the same/other/both pools by evaluating their respective target variables and features at different horizons and lags. The granularity level varies for each component within the Horizon Block Table, as illustrated in the following table:

$$HBT = \{(H_M, F_{DEX}, F_{DP}, F_{CEX})\} \text{ for each horizon } H_M \tag{7}$$

Component	Granularity Level
DEX target variable	Horizon block level
DEX direct pool features	Reference block level
CEX spillover features	Horizon block level

Table 2: Granularity Level of Components in Horizon Block Table

This granularity distinction allows for comprehensive examination of the behavior and interactions of the "same" and "other" pools at different levels within the time-series data model. The Horizon Block Table provides insights into the dynamics of the analyzed data, revealing the collaborative or competitive dynamics between the "same" and "other" pools throughout the time series.

Data Profiling: Highlights

In terms of merging Uniswap transactions (txt) with Etherscan, we've observed a high match rate of transactions for the analysis period of six months. The transaction breakdown is as follows:

- Uniswap txs: ((500, 'burns'), 4000), ((500, 'mints'), 3406), ((500, 'swaps'), 217821), ((3000, 'burns'), 5918), ((3000, 'mints'), 5046), ((3000, 'swaps'), 48592)
- Etherscan txs: ((500, 'burns'), 4000), ((500, 'mints'), 3242), ((500, 'swaps'), 217821), ((3000, 'burns'), 5918), ((3000, 'mints'), 4986), ((3000, 'swaps'), 48592)

From the transaction data, we created 8,227 reference blocks that correspond to each mint operation, excluding one. These blocks were then used to engineer features to calculate metrics for the same pool and other pools. These features capture data about liquidity pools and calculate metrics such as volatility, traded volume rate, trades count, and average volume. To expand the analysis, we have also included lagged features for the previous three mint operations. Table 3 serves as our block reference table.

Before initiating the modeling phase, we need to address records with missing values in the independent variables. Notably, some null values have been identified in the volatility (same pool) and avg-USD/rate-USD (other pool) metrics. We have opted to remove these records temporarily to proceed with the analysis, with plans to reassess the calculations and mitigate the occurrence of null values likely resulting from lagged variables.

pool	blockNumber	blockNumberChain	other_blockNumberChain
500	14498564	[14498564, nan, nan, nan]	[14498564, nan, nan, nan]
500	14498699	[14498699, 14498564, nan, nan]	[14498699, nan, nan, nan]
500	14499597	[14499597, 14498699, 14498564, nan]	[14499597, 14499560, 14499457, 14499198]
500	14499836	[14499836, 14499597, 14498699, 14498564]	[14499836, 14499560, 14499457, 14499198]
500	14500355	[14500355, 14499836, 14499597, 14498699]	[14500355, 14500043, 14499560, 14499457]
...	...	...	...
3000	15648981	[15648981, 15648330, 15648305, 15648187]	[15648981, 15648887, 15648536, 15646933]
3000	15649246	[15649246, 15648981, 15648330, 15648305]	[15649246, 15649243, 15648887, 15648536]
3000	15649545	[15649545, 15649246, 15648981, 15648330]	[15649545, 15649522, 15649347, 15649269]
3000	15649565	[15649565, 15649545, 15649246, 15648981]	[15649565, 15649522, 15649347, 15649269]
3000	15649578	[15649578, 15649565, 15649545, 15649246]	[15649578, 15649522, 15649347, 15649269]

Table 3: BlockNumber Chains

Finally, we've constructed a base table 4 with the `start_blockNumber` of each horizon and the `reference_blockNumber`, defined by mint operations in the block. This base table is joined with the mint aggregated transaction data to generate our target variables and independent variables that are lagged.

blockNumber	min_flag	reference_blockNumber	horizon_label	cum_volume_500
108757	0	15552674	9	423485.346309
108758	0	15552674	10	423485.346309
108759	1	15552772	1	328338.732259
108760	0	15552772	2	406084.780730
108761	0	15552772	3	536640.714920
...	...	...	...	...
109047	0	15555464	12	122730.731534
109048	0	15555464	13	123650.594764

Table 4: Reference and Horizon blocksn

In-Depth: Modelling Techniques

In our analysis, we will employ regression models to investigate the relationship between liquidity pool size and trading volume. Specifically, we will use an Ordinary Least Squares (OLS) regression approach.

We will develop and test OLS regression models for different target variables and horizons. Lagged variables will be incorporated into the models to predict trading volumes at different time horizons. The performance of the models will be evaluated using metrics like R-squared.

Additional modelling techniques and considerations, such as addressing autocorrelation, endogeneity, structural breaks, and multicollinearity, will be applied to enhance the accuracy and interpretability of the models.

We have commenced the development of a framework for our prediction models and have constructed several Proof of Concept (PoC) models. For the modeling phase, we're employing an Ordinary Least Squares (OLS) regression approach to predict multiple target variables across different horizons using a set of independent lagged

variables for each reference mint. This process involves iterating over the horizons, generating data subsets, fitting an OLS model for each horizon, and retrieving the R-squared value as a performance indicator. This approach allows for efficient and consistent prediction across various horizons.

$$\text{cum\_volume\_500}_{\text{horizon}} = \beta_0 + \beta_1 \cdot \widetilde{01}X + \beta_2 \cdot \widetilde{12}X + \beta_3 \cdot \widetilde{23}X + \ldots + \epsilon$$

In this formula, β_0, β_1, β_2, β_3, etc., represent the coefficients associated with each spot lagged variable, while ϵ denotes the error term or residual.

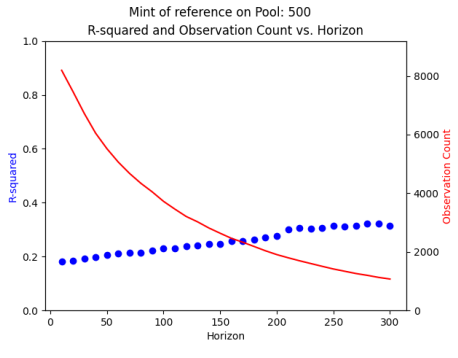

Figure 3: R2 best fit horizon

Observations indicate that around the horizon 20 (200 blocks, approximately 13 minutes), the R2 value ceases to increase significantly. Preliminary analysis of the relevant model (at horizon 20) reveals that the OLS regression model explains approximately 27.6% of the variation in the dependent variable cum_volume_3000_ref500. The significant variables, such as rate-count-isame_01, rate-count-isame_12, rate-count-isame_23, and wlother_3, have a considerable impact on the dependent variable. However, the potential collinearity among the independent variables could affect the interpretation of individual coefficients, suggesting a need for further analysis.

Model selection will be based on the optimal combination of features for each horizon. As we refine the feature and model development, considerations like autocorrelation, endogeneity, structural breaks, or multicollinearity will be addressed. These will be managed through additional sensitivity analyses, cross-validations, and the application of appropriate statistical techniques to diagnose and mitigate any complexities.

Model Optimisation and Selection

Once the initial models are developed, we will evaluate their performance and optimize them for better accuracy and predictive power. Model optimization may involve refining the feature selection, reducing multicollinearity, and exploring alternative functional forms or transformations of variables.

Model selection will be based on the optimal combination of features, horizons, and target variables that provide accurate predictions and meaningful insights.

Analysis and Discussion

In the final stages of our analysis, we will conduct an in-depth examination of the results and draw conclusions based on the findings. We will analyze the relationships between liquidity pool size and trading volume in BTC-ETH pools, considering other factors such as spillover effects and price divergences.

We will discuss the implications of our findings for liquidity provision, DeFi market efficiency, and various stakeholders. The quantitative-driven insights from our analysis can help optimize liquidity provision, refine trading strategies, and improve the effectiveness of liquidity pools in the DeFi ecosystem.