Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Arbitrage Opportunities on Decentralized Exchanges

Bachelor's Thesis

Simon Wachter

siwachte@ethz.ch

Distributed Computing Group
Computer Engineering and Networks Laboratory
ETH Zürich

Supervisors:
Lioba Heimbach, Robin Fritsch
Prof. Dr. Roger Wattenhofer

August 21, 2022

Acknowledgements

I would like to thank my supervisors Lioba Heimbach and Robin Fritsch for their support and guidance. They quickly provided me with the right approach to solve problems that I was stuck with. Further, I want to thank Prof. Dr. Roger Wattenhofer and the Distributed Computing Group for making it possible for me to work on my thesis and to provide the necessary computing infrastructure.

Abstract

We analyze arbitrage opportunities on decentralized exchanges on the Ethereum blockchain with a focus on the Uniswap V3 protocol. In previous works, Uniswap V3 was not considered as it was only released in May 2021. A relation between the total value of arbitrage opportunities and ETH price changes could already be established. In our analysis we further analyze this relation and show a statistically significant correlation between the total amount of arbitrage opportunities and ETH price changes.

Contents

Chapter 1

Introduction

1.1 Background

1.1.1 Decentralized Exchanges

Decentralized exchanges (DEXes) are one of the fundamental building blocks of decentralized finance (DeFi). They enable the exchange between two assets using an underlying smart contract. They do not rely on the exchange itself or any third party to be a market maker. Rather, most employ the constant product market maker model (CPMM), which functions without the need for active market making. The model is based on keeping the product of the two asset reserves in a liquidity pool equal to a constant k. A liquidity pool is an Ethereum address where users can deposit funds and therefore provide liquidity, which is then used to facilitate the trade of the deposited assets. Liquidity providers (LPs) are incentivised by a trading fee that is paid by the users wanting to exchange assets through a pool. DEXes reach a price equilibrium through arbitrage. Meaning if the price of an asset in one pool is much lower than the price in a different pool, a profit can be generated by buying the asset in the cheap pool and selling it in the expensive one.

1.1.2 Uniswap V2 and the Constant Product Market Maker

The Uniswap protocol is the largest decentralized exchange, with an average monthly trading volume of over 50 billion in the last 12 months [1]. The large growth of trading volume in May 2021 can be attributed to the launch of Uniswap V3, which brought a lot of attention and liquidity to the protocol.

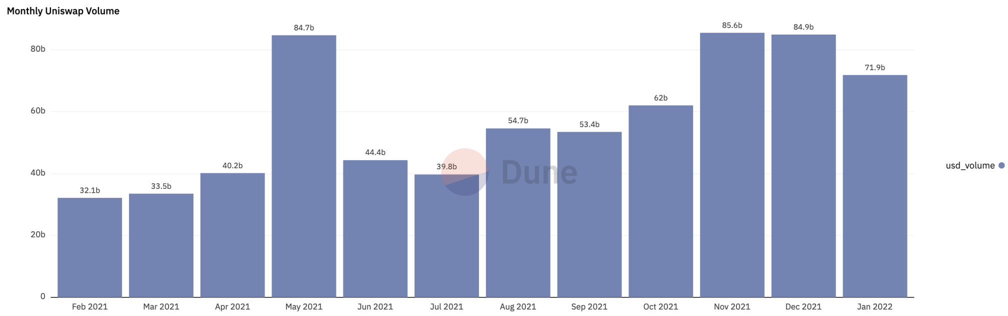

Figure 1.1: Monthly trading volume of Uniswap [1]

The exchange is built on the Ethereum blockchain and uses the CPMM model to facilitate the trade of two assets against each other. The price p of a pool is determined by the fraction of the amounts of each token res_x and res_y that are deposited [2]:

$$p = \frac{\text{res}_x}{\text{res}_y}$$

The price of the pool is the price of tok_y in terms of tok_x. The inverse of p therefore is the price of tok_x in terms of tok_y. The constant k is the product of the reserves of the two assets:

$$k = \text{res}_x \cdot \text{res}_y$$

In a zero fee pool, after the inception of the pool, k only changes when liquidity is deposited or withdrawn. It has to stay constant when a user is exchanging assets. In practice, however k increases with each trade, as the trading fee is added to the pool and not taken into account when calculating the output amount for the trade. The equation describes the curve shown in Figure 1.2:

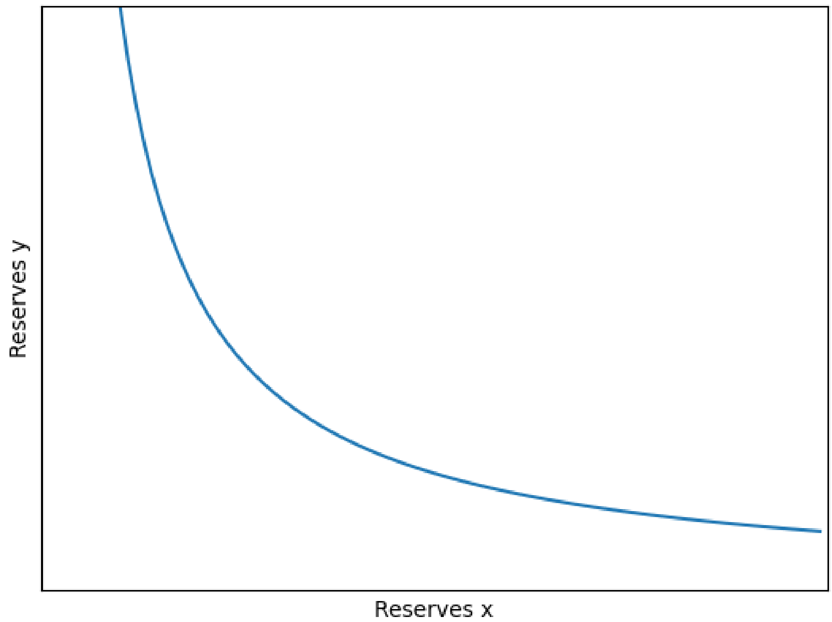

Figure 1.2: Uniswap V2 Liquidity Curve

From this simple equation, others can be derived that describe how much tok_y^{out} is returned by the pool when a certain amount tok_x^{in} is inserted and vice versa.

$$\text{tok}_y^{out} = \frac{k}{\text{res}_x + (\text{tok}_x^{in} \cdot (1 - \gamma))} - \text{res}_y \tag{1.1}$$

$$\text{tok}_x^{out} = \frac{k}{\text{res}_y + (\text{tok}_y^{in} \cdot (1 - \gamma))} - \text{res}_x \tag{1.2}$$

Here, γ is the fee as a portion of 1 that is paid to the pool. Notice that tok_x^{out} and tok_y^{out} are negative here, as the equations describe the change of the reserves from the view of the pool. The pools reserves are then adjusted according to the amount and the funds are transferred to the user who initiated the swap.

1.1.3 Uniswap V3

In May 2021, Uniswap launched their new V3 protocol, which aimed to improve capital efficiency and slippage [3]. Slippage is a measure of the amount that the expected price of a trade differs from the actual price that the trade was executed at. These improvements were achieved by making it possible to provide liquidity in a certain price range, rather than from 0 to ∞. The concept is called concentrated liquidity. LPs can now choose a range in which they provide liquidity. The chosen range in relation with the current price defines the ratio of tok_x and tok_y that the LP has to provide. Further the LPs only receive fee payments if the price is in their specified range. The space of possible prices is divided into discrete ticks. The range chosen by LPs always has to be between two ticks. From the tick i, the price can be calculated as follows:

$$\begin{aligned} p(i) &= 1.0001^i \\ \sqrt{p(i)} &= 1.0001^{i/2} \end{aligned}$$

For technical reasons however, the pools track the square root price, rather than the normal price. The smart contract also tracks the liquidity L, which is currently active [4].

$$\begin{aligned} L &= \sqrt{\text{res}_x \cdot \text{res}_y} \\ \sqrt{P} &= \sqrt{\frac{\text{res}_y}{\text{res}_x}} \end{aligned}$$

Each time when a tick is crossed the liquidity might change. The liquidity of an LPs position will be subtracted or added depending on wether the crossed tick was an upper or lower bound. The basic calculation for Uniswap V3 is the same as for Uniswap V2, however limited to the price range between two adjacent initialized ticks. When a tick is crossed, liquidity has to be updated, which in turn might effect the price curve and the output amount. The amount of tok_x^{out} for a specific price movement can be derived from the Equations (6.14) and (6.16) from the Uniswap V3 whitepaper [4]:

$$\text{tok}_x^{out} = L \cdot (\sqrt{P_B} - \sqrt{P_A}) \qquad (\sqrt{P_B} \geq \sqrt{P_A}) \tag{1.3}$$

$$\text{tok}_y^{out} = L \cdot (\frac{1}{\sqrt{P_B}} - \frac{1}{\sqrt{P_A}}) \qquad (\sqrt{P_B} \leq \sqrt{P_A}) \tag{1.4}$$

P_A and P_B are prices of the pool at the beginning and the end of a part of the swap. Equations (1.3) and (1.4) only apply as long as P_B and P_A lie between two adjacent ticks. If a tick is crossed, the liquidity might change, which alters the liquidity and hence the output amount.

1.2 Related Work

Previous work has focused on the study of cyclic arbitrage on Uniswap V2. The Cyclic Arbitrage Model was presented and the value of exploitable arbitrage opportunities was studied by Wang et al. [5].
Further the DEX market was compared to the centralized exchange market. Berg [6] analyzed cyclic arbitrage opportunities and trade optimizations, showing a relation between the value of arbitrage opportunities over a time span and the corresponding ETH price changes.
Qin et al. [7] studied the impact of blockchain extractable value (BEV) on the security of the blockchains consensus. In their analysis of arbitrage, which is part of BEV, they found opportunities that amounted to a profit of over 270M USD from the 1st of December 2018 to the 5th of August 2021.

Chapter 2
Data Collection

Data was collected between March 8th 2021, a few days after the Uniswap V3 contracts were deployed, and January 13th 2022.
A lot of DEXes exist on the Ethereum blockchain, most of which do not have significant trading volume cf. Figure 2.1. In our analysis only Uniswap V2, Uniswap V3 and Sushiswap were considered because over the analyzed period, these three DEXes had the highest trading volume. The Curve protocol also had high trading volume, but it is an exchange for stablecoins and specifically designed to facilitate big trades with small slippage. Further selected were the biggest liquidity pools for each exchange, where most trading volume was coming from. Another criterion was a significant trading volume on multiple of the selected exchanges. Pools meeting those criteria are: DAI/ETH, DAI/USDC, DAI/USDT, ETH/USDT, SUSHI/ETH, UNI/ETH, USDC/ETH, USDC/USDT, WBTC/ETH, WBTC/USDC. On Uniswap V3, all fee tiers with significant trade volume for these pairs were analyzed. The fee tiers are either 0.01%, 0.05%, 0.3% or 1% and they describe how much of the swap input amount the user has to pay to the pool.

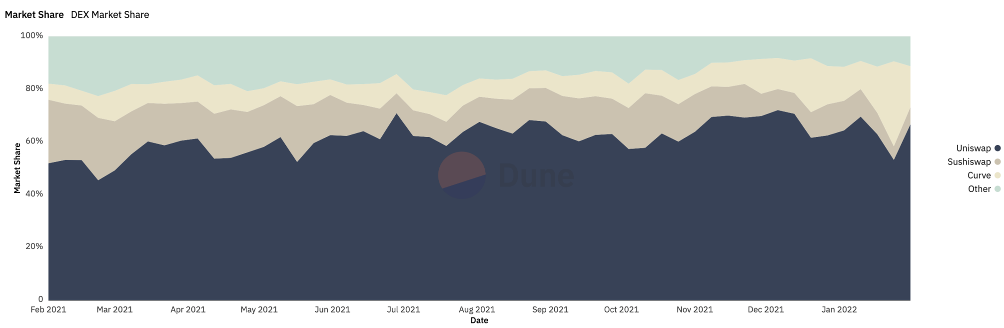

Figure 2.1: Market Share of DEX's

Because DEXes operate on the Ethereum public ledger, all transaction data is public. Erigon, an Ethereum client, was used to retrieve all the needed data. Regarding the Uniswap V2 and Sushiswap pools, the state of interest were the reserves of tok_x and tok_y ("reserve0" and "reserve1" from the contracts). For Uniswap V3 it is the liquidity, the square root price and the current tick. To be able to simulate the swaps in a Uniswap V3 pool it was also necessary to know the amount of liquidity that is added or removed at each initialized tick. We built a dataset that includes all these values for each analyzed pool and block.
In addition, we gathered ETH price data from Binance [6]. For the analyzed period we gathered Candlestick data of the ETH/USDT pair for all time frames. From the candlesticks, the price changes of ETH can be derived by calculating the percentage difference from the lowest price to the highest price in a single candle.

Chapter 3

Processing

Rather than simulating the contracts on an Ethereum node, we used our own implementation that follows the same underlying equations as the contracts deployed to the Ethereum blockchain. The whole data collecting and processing was done in Python. The code is accessible on Github [8].

3.1 Simulating Uniswap V2

To simulate the swaps for Uniswap V2 and Sushiswap pools, Equations (1.1) and (1.2) need to be implemented. The procedure first calculates the constant k by multiplying the two reserves and then returns the output amount. The boolean variable zeroForOne states whether a user wants to exchange asset x for asset y or asset y for asset x. Notice, that the output is positive here, unlike in Equations (1.1) and (1.2) where the left side of the equation is negative. This is because we are interested in the difference of res_x or res_y before and after the swap.

Algorithm 1 Uniswap V2 Swap

procedure SWAPV2(tok^{in}, zeroForOne, res_x, res_y)
    $k = \text{res}_x \cdot \text{res}_y$ ▷ Calculate k
    $\text{tok}^{in} = \text{tok}^{in} \cdot (1 - 0.003)$ ▷ Subtract the fee
    if zeroForOne **then**
        return $\text{res}_y - (k/(\text{res}_x + \text{tok}^{in}))$
    else
        return $\text{res}_x - (k/(\text{res}_y + \text{tok}^{in}))$

3.2 Simulating Uniswap V3

In contrast to the calculations of the Uniswap V2 contracts, for which all calculations can be done with two equations, the Uniswap V3 contracts require an iterative method. One iteration is needed for each crossed tick and each iteration then does similar computations as for a Uniswap V2 swap. The main part of the algorithm is a while loop that runs until the input amount of tokens is depleted, analog to the implementation in Solidity. In the loop, it is calculated how many tok_x are needed to move the price to the next tick and how many tok_y get out when moving the price to the next tick or vice versa. This is done by the procedure shown in Algorithm 2. Here P_A and P_B are the current price and the price to move to respectively.

Algorithm 2 Uniswap V3 getAmountDelta

procedure GETAMOUNTDELTA(P_A, P_B, liquidity, zeroForOne)
    if zeroForOne **then**
        if $P_A > P_B$ **then**
            $P_A, P_B = P_B, P_A$
        return liquidity $* ((1/\sqrt{P_A}) - (1/\sqrt{P_B}))$ ▷ Calculates Equation (1.4)
    else
        if $P_A < P_B$ **then**
            $P_A, P_B = P_B, P_A$
        return liquidity $* (\sqrt{P_A} - \sqrt{P_B})$ ▷ Calculates Equation (1.3)

3.3 Optimal Swap Size

To find the optimal swap size for two Uniswap V2 pools, Equations (1.1) and (1.2) can be combined. We define k_i, $\text{res}_{x,i}$, $\text{res}_{y,i}$ as the constant k and reserves for pool i where $i \in \{0, 1\}$. Pool 0 is used to swap from asset x to some asset y and in pool 1 the asset y is swapped back to asset x. Resulting in the following equations:

$$\text{tok}_x^{out} = \frac{k_1}{\text{res}_{y,1} + \frac{k_0}{\text{res}_{x,0} + \text{tok}_x^{in}} - \text{res}_{y,0}} - \text{res}_{x,1}$$

$$\text{profit}(\text{tok}_x^{in}) = \text{tok}_x^{out} - \text{tok}_x^{in}$$

$$\text{profit}'(\text{tok}_x^{in}) = \frac{k_0 \cdot k_1}{(k_0 - (\text{res}_{y,0} - \text{res}_{y,1})(\text{res}_{x,0} + \text{tok}_x^{in}))^2} - 1 \tag{3.1}$$

$$\text{tok}_x^{in} = \frac{k_0 - \sqrt{k_0 \cdot k_1}}{(\text{res}_{y,0} - \text{res}_{y,1})} - \text{res}_{x,0} \tag{3.2}$$

This model describes the simplest cyclic arbitrage opportunity, with two different pools with the same two tokens. The optimal input amount can be derived by setting the derivative of the profit in Equation (3.1) to zero and solving for tok_x^{in}, see Equation (3.2). For cycles that only contain Uniswap V2 or Sushiswap pools it is therefore possible to solve for the best input amount by deriving an extended version of Equation (3.2), one containing more pools.
If there is a Uniswap V3 pool in the cycle, then another approach is needed. Due to the implementation of the V3 contracts, it is not possible to solve for the optimal input amount with one equation, rather we need to use an iterative scheme as described in Section 3.2.
By definition of the CPMM model, it is clear that for each additional token that we swap in a pool, the price worsens. This applies to Uniswap V2 and Uniswap V3 pools. The rate of the price change depends on the amount of liquidity in the pool. The slippage of a trade measures how much the expected price (at the start of the trade) differs from the actual price, the one the trade was executed at. Because the price changes for each input token that is swapped in a pool, the price before and after the trade are not the same. The actual price that the user paid is between the price before and the price after the trade. When we know the input and output amounts of the trade, the executed price can easily be calculated as:

$$\text{executed price} = \frac{\text{tok}^{out}}{\text{tok}^{in}}$$

The slippage is calculated as:

$$s = \frac{P_{\text{expected}} - P_{\text{actual}}}{P_{\text{expected}}}$$

The slippage increases when our expected price decreases, which happens when the input amount gets bigger. Hence, the slippage is proportional to the input amount. The slippage is also dependent on the liquidity in the pool, as the liquidity impacts the rate of the price change.

$$\text{tok}_x^{out} = \text{tok}_x^{in} \cdot (P_0 - s_0) \cdot (P_1 - s_1) \cdot \ldots \cdot (P_n - s_n) \quad s_i \geq 0 | i \in \{0, ..., n\} \tag{3.3}$$

$$\text{tok}_x^{out} = \text{tok}_x^{in} \cdot (P_c - s_c) \tag{3.4}$$

$$\text{profit}_x = \text{tok}_x^{out} - \text{tok}_x^{in} \tag{3.5}$$

Where P_i and s_i are the price and slippage of pool i, the fee is directly subtracted from the price and not shown for readability. P_c and s_c are the aggregated price and slippage over the whole cycle respectively.

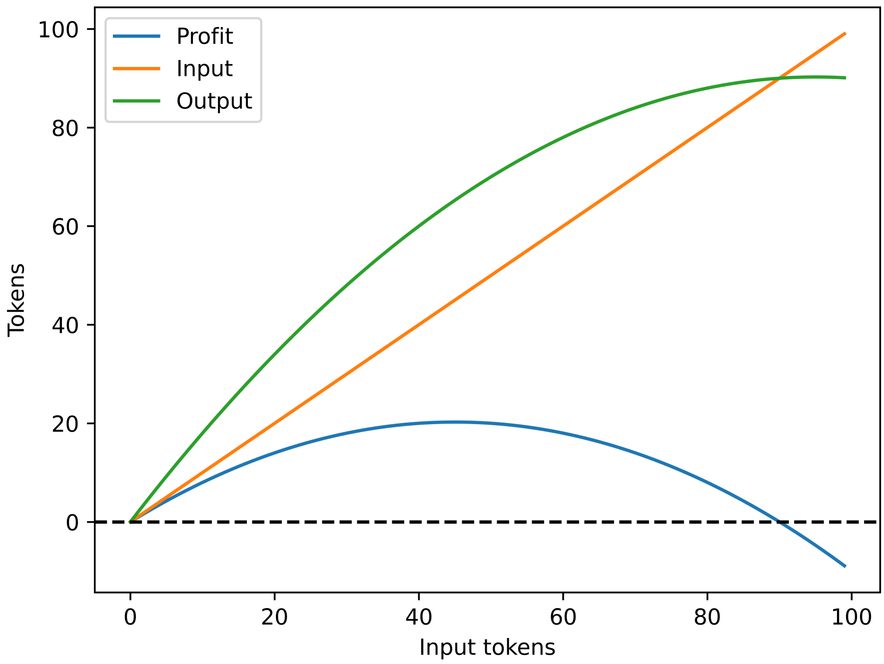

Figure 3.1: Profit and token output curve. The output curve has the largest slope at an input of 0. At the same point where the output curve crosses the input curve, the profit curve crosses into the negative.

Figure 3.1 depicts Equations (3.4) and (3.5). The curvature of the output function relates to the amount of slippage the trade is generating. As more input is given, the ratio of $\frac{\text{output}}{\text{input}}$ begins to decrease. For infinitely large inputs, the $\frac{\text{output}}{\text{input}}$ ratio converges to 0. As long as the output amount is greater than the input amount, the profit function will be positive. For infinitely large input amounts, the profit function diverged to $-\infty$. Both, the output and profit function behave unimodal. A unimodal function must have an m such that for all $x \leq m$ the function is monotonically increasing and monotonically increasing for all $y \geq m$. Further, $f(m)$ is the maximum of the function and no other local maximum exists. Because of this unimodal property, ternary search can be used to find the optimal size for a cyclic trade. Ternary search runs in $\Theta(log(n))$ time. Optimizing the input amount for a cycle, starting with 0.1 of the starting token as the upper limit and 0 as the lower limit proved to be the fastest. Other approaches, such as starting with an upper bound of an amount equivalent to 1, 10 or 100 USD or 1 ETH, proved to be slower. In each iteration of the ternary search algorithm the swap is simulated for the whole cycle with the given input amount. The profit amount of each simulation is compared and the algorithm determines which part of the search space can be discarded. This process is repeated until a search space smaller than 10% of the size of the initial space was reached. Setting this parameter to a lower value would have been ideal, but was not possible due to a drastic increase in the runtime of the analysis. The maximum of the profit function for the given cycle must lie in this space. If the found space is close to the upper limit, then the algorithm is restarted with an upper limit 10 times bigger than the previous one.

3.4 Graph Modelling

The current state of the whole market on all exchanges can be modeled as a graph cf. Figure 3.2. Each coin is represented by a vertex and the pools are the edges connecting two of the coins. Our set of coins consists of ETH, DAI, SUSHI, UNI, USDC, USDT and WBTC. The weights of the edges represent the negative logarithm of the price of the pool.

$$w_{xy} = -log_2(P_{xy} - f_{xy})$$

w_{xy} is the weight of edge (x, y) and P_{xy} is the best price across all exchanges from tok_x to tok_y and f_{xy} is the fee of the pool with the best price. Hence, for each edge $e = (x, y)$ exists another edge e' with $e' = (y, x)$. For a coin pair (x, y) it is possible that edge (x, y) has its weight from a different pool than (y, x). The fee is subtracted from the price because when analyzing the graph model, at first, no swaps are simulated and therefore the fee cannot be subtracted from a given input amount. By including it directly with the price, we discard some negative cycles that would otherwise be present in the graph. These discarded cycles however do not represent arbitrage opportunities as they are only viable if no fees have to be paid.

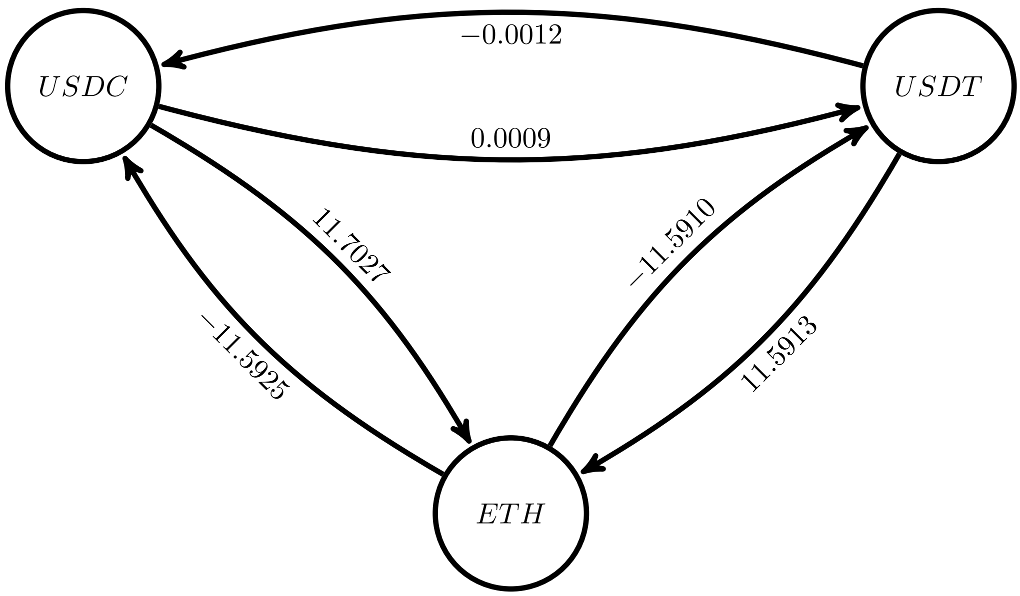

Figure 3.2: Example subgraph of our model.

The graph models arbitrage opportunities as negative weight cycles. For any cyclic arbitrage opportunity the price over the whole cycle P_c must be greater than 1. P_c is the product of the price P_i of all pools i on the cycle. For an arbitrage opportunity to be profitable the output amount after the cycle needs to be greater than the input amount. This is only the case if the product of all prices P_c (including fees) is greater than 1.

$$\begin{aligned} P_c &> 1 \\ P_0 \cdot ... \cdot P_n &> 1 \\ \log_2(P_0 \cdot ... \cdot P_n) &> \log_2(1) \\ \log_2(P_0) + ... + \log_2(P_n) &> 0 \\ -\log_2(P_0) - ... - \log_2(P_n) &< 0 \end{aligned}$$

The Bellman-Ford algorithm could be used to find just any negative cycle, however to find the best arbitrage opportunities, all negative cycles need to be found. This is because the price difference between two exchanges is only enough to determine if a profitable arbitrage trade across the cycle is possible, but not how profitable this opportunity will be. The reason being that in pools with more liquidity the slippage increases less than in smaller pools. There can exist two cycles where one has a much bigger price difference P_c but still is a worse opportunity because the pools are too illiquid. Following the same reasoning it would also not be possible to only look at the most negative cycle.
In our case, the graph is small enough to be able to enumerate all cycles and sum up the weights along them. If a cycle has negative weight, the optimal swap size is determined as described in Section 3.3.

3.5 Example Cycle

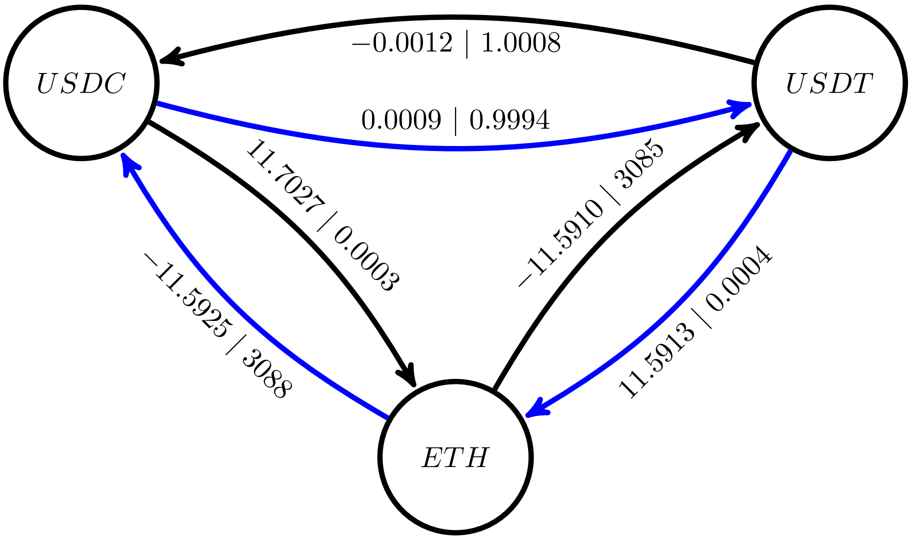

Figure 3.3: Negative weight cycle for ETH → USDC → USDT → ETH. Edges are labeled as $w_{xy} \mid P_{xy}$

Figure 3.3 shows a negative weight cycle in the graph. The price over the whole cycle is 1.23, which means that there is a 23% price difference. To use this opportunity one would first swap ETH for USDC, then USDC for USDT and finally USDT back to ETH. The graph is nearly identical to the state on the blockchain at block 14'447'743. This cycle was used by an arbitrageur. The input was 5'243 ETH and the output was 5'518 ETH [9]. A profit of 275 ETH was made, which corresponds to about 855K USD at the time of the trade.

Chapter 4

Evaluation

The evaluated period spans 2M blocks, namely from block 12M on March 8th 2021 to block 14M on January 13th 2022. During this period every block was analyzed and 540M negative cycles occurred. The evaluation of the cycles yielded 125M arbitrage trade opportunities. Over the whole period a total of over 90M USD in arbitrage opportunities were found, only counting the most profitable one in each block. Figures 4.1 and 4.2 show the total value of arbitrage opportunities (TVAO) over the whole analyzed period. Both periods account for about half of the observed TVAO. In the second period the TVAO is much more distributed but there is still a period where a big part of the total TVAO is concentrated. In the second period, the TVAO is more consistent, in the last three months there was a sustained TVAO of over 200K USD every day. Further, instead of a spike that lasted a few days, like in May 2021, in late November and throughout December 2021 the TVAO was consistently around 1M USD per day.

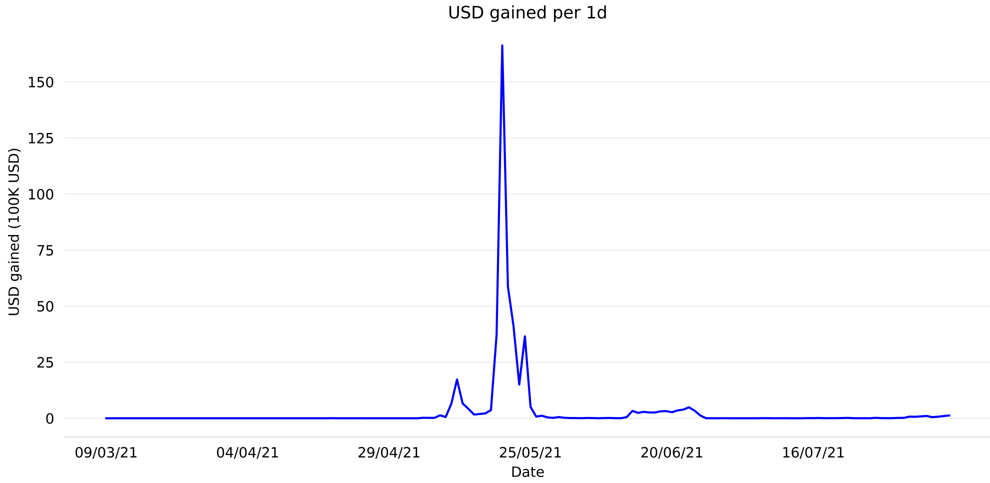

Figure 4.1: TVAO over a period of 1M blocks, which is roughly half a year. The TVAO is very concentrated around May 2021 where ETH experienced a period of high volatility (price changes of $> 15\%$).

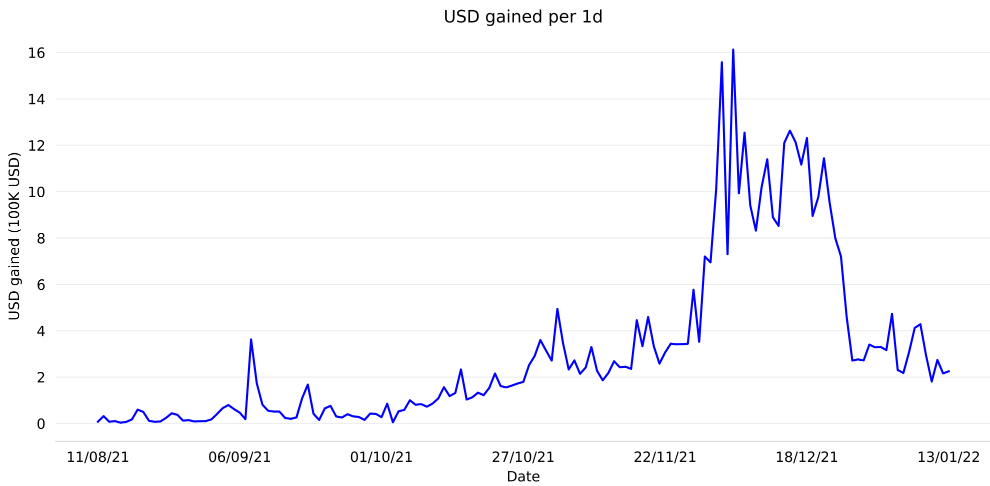

Figure 4.2: TVAO over a period of 1M blocks, representing the second period of our analysis. TVAO is more consistent but still experiences some periods where it is significantly higher than in others.

4.1 Arbitrage opportunities and ETH price changes

In this section we will focus on the correlation between the TVAO and ETH price changes. Arbitrage opportunities rely on price differences between different pools. These price differences are often caused by large changes in the price of ETH, high trading volume or large swaps that move the price by themselves. Additionally, irrational behavior of traders, caused by large ETH or BTC price changes, can lead to selloffs of a particular asset, which in turn can lead to price differences.
One of the most interesting periods is in May 2021, where the price of ETH reached new highs and then plummeted again. This led to daily price changes of over 60% and was the first period of high volatility since Uniswap V3 was launched.

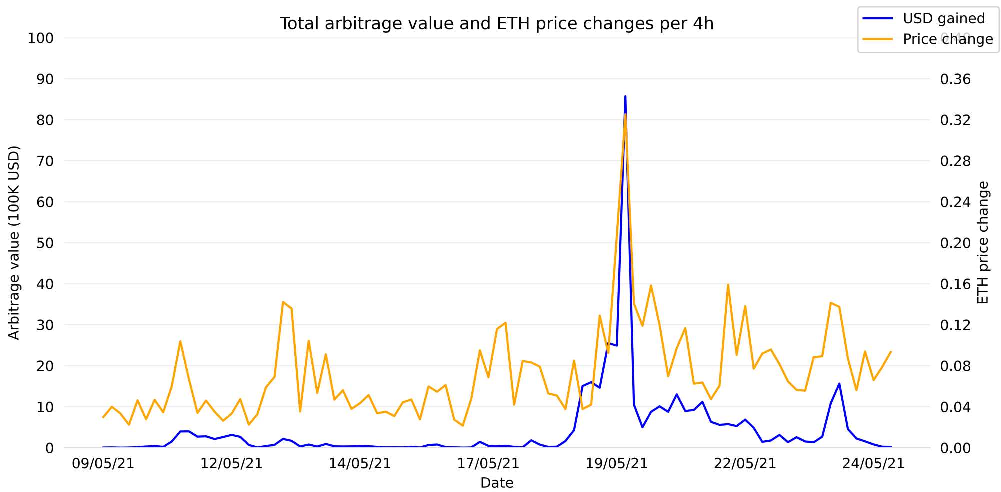

Figure 4.3: ETH price changes and TVAO over a period of 100K blocks (~ 17 days). The two data series have a pearson correlation coefficient of $r = 0.71$. A statistically significant correlation is observed.

During the highest volatility, namely on May 19th, a peak of TVAO of over 8M USD was found. These findings reappear in other periods of high volatility, however not in such a short time period. Reason might be the fact that Uniswap V3 was only introduced a few weeks before and arbitrageurs were not yet in the position to take full advantage of the price changes, causing prices to diverge more and thus creating bigger opportunities. Other notable periods with a correlation between the price change and the TVAO are shown in Figure 4.4. Although the ETH price changes are not as high as in May, a spike in TVAO is notable.

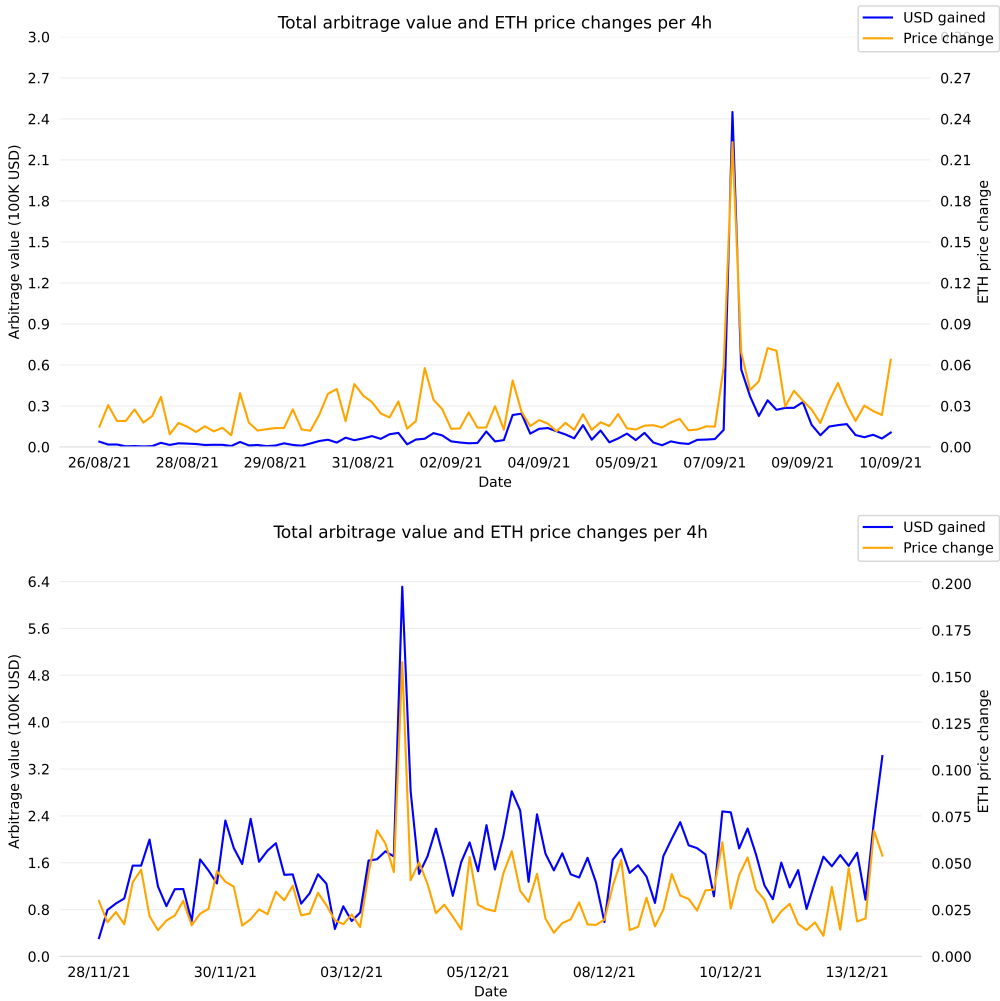

Figure 4.4: ETH price changes and TVAO over periods of 100K blocks. Both periods have a statistically significant correlated between the two data series.

Comparing the three shown periods from Figures 4.3 and 4.4 to the overall ETH price changes, shows that those periods of high correlation occur in time periods where the ETH price changes the most.

4.2 Arbirage opportunities and ETH price

Most arbitrage opportunities involve ETH pools or even start and end with ETH. This is because ETH is the base currency in the DEX system and most of the liquidity is in pools where ETH is one of the two assets. This led to the assumption, that the TVAO would increase with an increase in the price of ETH.
This assumption turned out to be false.In the first half of the analyzed period, no significant correlation between the ETH price and the TVAO could be found cf. Figure 4.5. Moreover, the spike in TVAO happened during a period of a strong decrease in the price of ETH. The weak correlation can also be attributed to the fact that this spike creates a large outlier, which has a large impact on the overall correlation.

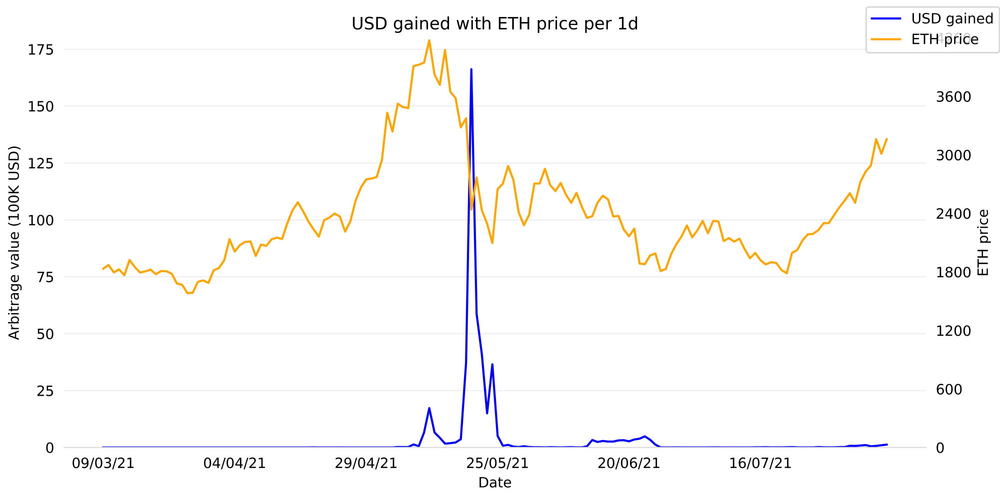

Figure 4.5: ETH price and TVAO over a period of 100K block. No significant correlation between the two data series can be found.

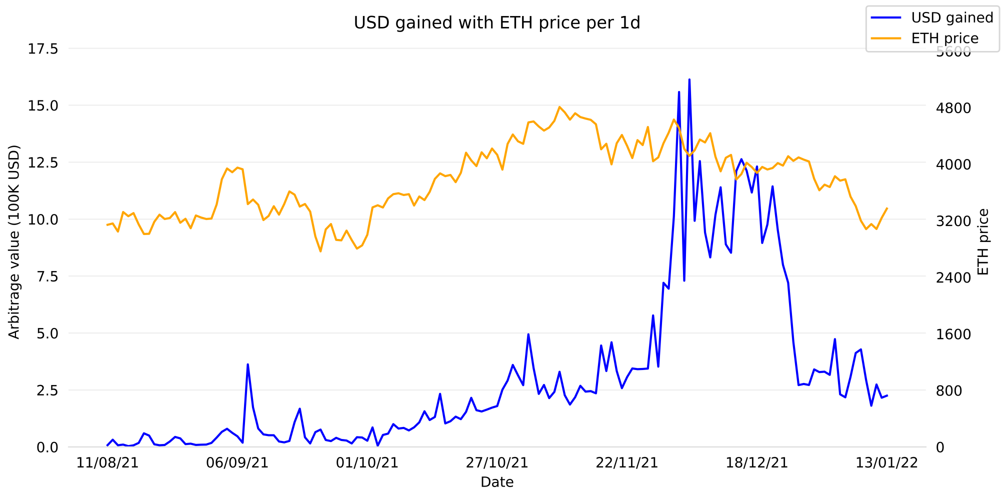

Figure 4.6: ETH price and TVAO over a period of 100K block. The pearson coefficient for the two data series is $r = 0.48$ and the correlation is statistically significant.

In the second period, where the price was generally higher than in the first one, there is a stronger correlation but still much weaker than we have observed between TVAO and ETH price changes.

Chapter 5

Conclusion

In this paper we provided insight into cyclic arbitrage opportunities. We analysed the largest DEXes Uniswap V2, Uniswap V3 and Sushiswap. Our data showed that the TVAO is correlated with price changes of ETH. Specifically, in periods of high volatility, larger and more arbitrage opportunities exist. Further, we could not find a significant correlation between the TVAO and the ETH price. Overall the TVAO got more consistent during the second half of 2021.

Bibliography

[1] S. Wachter, "Dune dashboard," 2022. [Online]. Available: https://dune.com/niedo/market-efficiency-on-dex

[2] "Uniswap v2 whitepaper," Mar 2020. [Online]. Available: https://uniswap.org/whitepaper.pdf

[3] U. Team, "Introducing uniswap v3," Mar 2021. [Online]. Available: https://uniswap.org/blog/uniswap-v3

[4] "Uniswap v3 whitepaper." Uniswap, Mar 2021. [Online]. Available: https://uniswap.org/whitepaper-v3.pdf

[5] Y. Wang, Y. Chen, H. Wu, L. Zhou, S. Deng, and R. Wattenhofer, "Cyclic arbitrage in decentralized exchanges," 2021. [Online]. Available: https://arxiv.org/abs/2105.02784

[6] J. A. Berg, "Optimal trading and arbitrage in a network of deventralized exchanges," 2021.

[7] K. Qin, L. Zhou, and A. Gervais, "Quantifying blockchain extractable value: How dark is the forest?" in *2022 IEEE Symposium on Security and Privacy (SP)*, May 2022, pp. 198–214.

[8] S. Wachter, "Market efficinecy on decentralized exchanges," August 2022. [Online]. Available: https://github.com/ni3do/arb_opp_dex

[9] Etherscan, "Large arbitrage transaction," 2022. [Online]. Available: https://etherscan.io/tx/0x631a4941eb8d0903c1c0073784423f87019cddd7c3822c77258172bd8d1a862c

Appendix A
Additional Figures

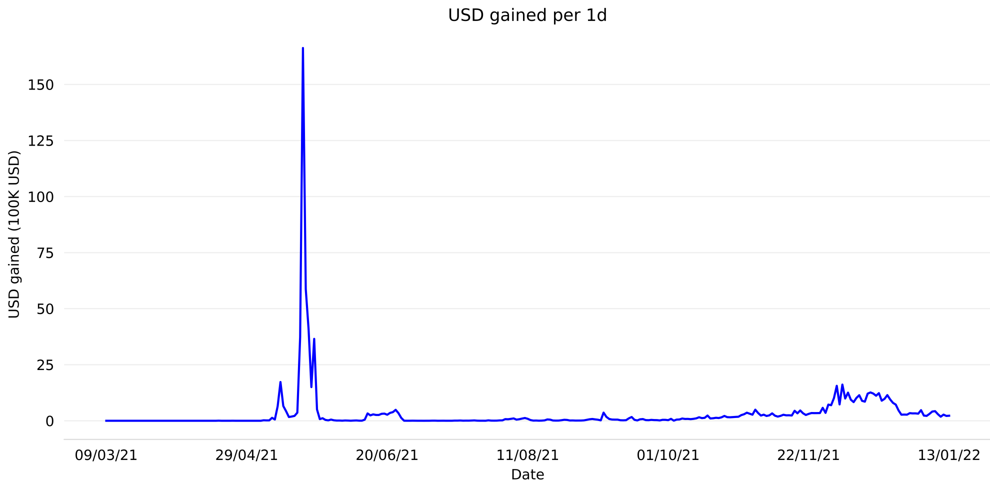

Figure A.1: TVAO over whole period. The May period was clearly the most active over the whole period. The TVAO is 10 times the amount of the peak of the second period.

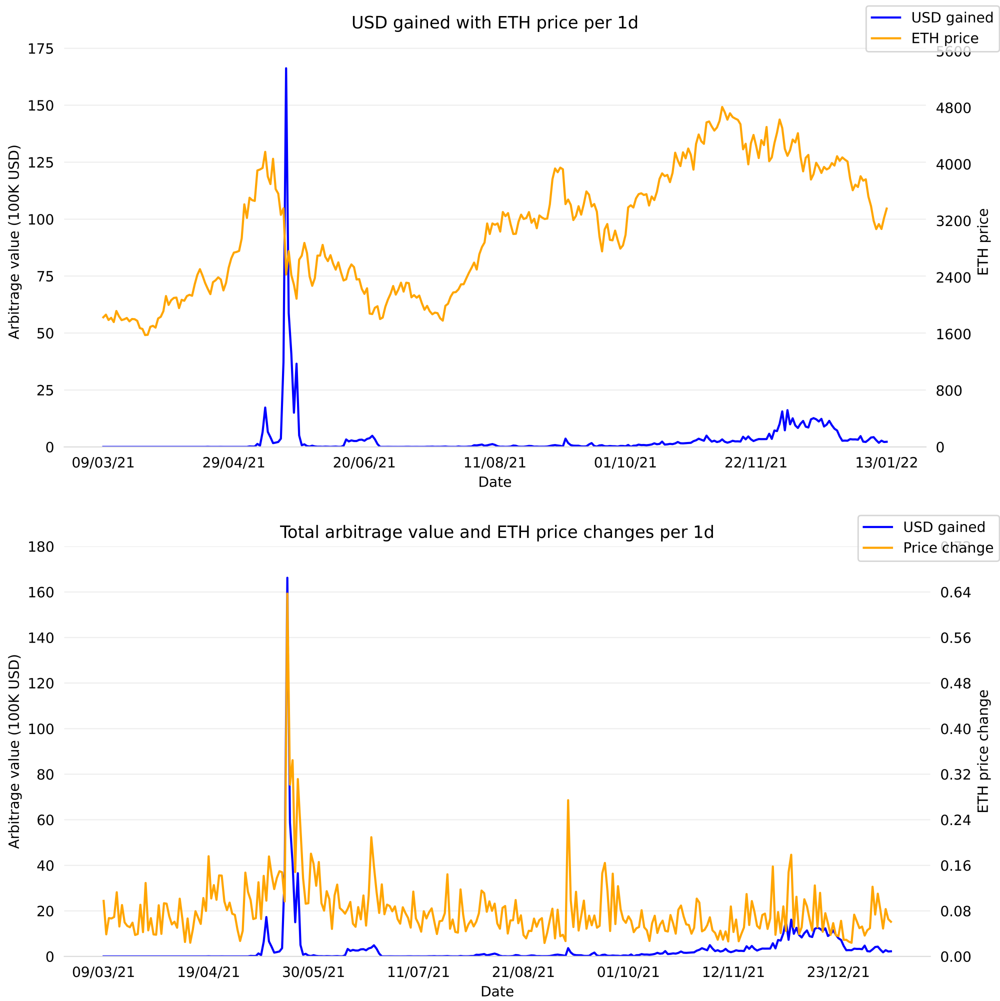

Figure A.2: TVAO and price changes/price over the whole analyzed period. The overall pearson coefficient for the TVAOand the price changes is $r = 0.69$. The pearson coefficient with the ETH price is only $r = 0.09$