KirtuSwap

Automated market maker based on virtual pools

Abstract

KirtuSwap mitigates the problem of high trading costs on constant-function-based automated market makers in pairs for which liquidity pools are small or inexistent. KirtuSwap solution relies on virtual liquidity pools, trades on which are executed by creating limited (in size and time) reserves of tokens that are not native to a pool. The gains to liquidity providers for taking the resulting limited risk of temporarily holding non-native tokens are large, as demonstrated by examples and simulations using real data parameters. Overall, KirtuSwap financial technology allows trading on an automated-market-maker-based decentralized exchange at significantly lower costs, while simultaneously increasing expected profits to liquidity provision.

1 The basics of KirtuSwap

Decentralized crypto exchanges (DEXes) based on automated market making (AMM) are a cornerstone of DeFi. DEXes solve several problems associated with centralized crypto exchanges (CEXes), such as fake (wash) trading (e.g., Aloosh and Li (2021), Amiram, Lyandres and Rabetti (2021), and Cong, Li, Tang and Yang (2021)) and outright fraud.

The benefits of AMM-based DEXes notwithstanding, there are also drawbacks. Perhaps the most important one is that trading on them tends to be more expensive than on CEXes, both in terms of transaction costs and price impact or slippage (e.g., Lehar and Parlour (2021)). This is perhaps the reason that the overall trading volume on DEXes is currently dwarfed by the overall trading volume on CEXes (around $100 billion and $US 2 trillion, respectively, in September, 2021).

An important reason for relatively high trading costs on DEXes is that in the vast majority of trading pairs liquidity pools are either small or absent. On existing DEXes, a trade in a token pair with small/nonexistent pool is routed through two or more other pools, vastly increasing both trading fees and slippage. This problem cannot be solved by adding sufficient liquidity to all pools, even on DEXes with large total value locked (TVL), as the number of trading pairs is close-to quadratic in the number of currencies.

A solution that KirtuSwap proposes is creation of "virtual pools".
Trading using a virtual pool is akin to using liquidity on other pools,
without resorting to indirect routing through multiple pools. Put differently, trading in small/
nonexistent pools amounts to executing only the buy portion of the trade
on a real pool and placing a reserve of non-native currencies sold in the trade (up to a limit) in that real pool. While the presence of reserve currencies exposes liquidity providers to the risk of currencies other than those in which they have provided liquidity,
KirtuSwap algorithm ensures that this risk is relatively small and short-lived.
The rewards for taking this risk are large. Simulations using real data parameters, such as estimates of cross-currency exchange rate volatilities and of trading volumes in various pairs, demonstrate that the gains from using KirtuSwap financial technology amount to: i) a signifi
cant reduction in liquidity traders' costs – fees, but more importantly, slippage (around 20%
for currency pair involving medium-sized currencies), and ii)
higher returns and Sharpe ratios of liquidity providers, especially in trading
pairs associated with smaller pools.
A crucial reason for the ability of KirtuSwap algorithm to reduce trading costs and simultaneously incre
ase liquidity providers' returns is significant reduction in arbitrageurs' profits. In theory,
in the absence of trading fees and (external) exchange rate volatility,
every liquidity trade on any existing constant-function-market-maker (CFMM)
is followed by an arbitrage trade of the same size in the opposite direction (see,e.g., Wang, Chen, Deng
, and Wattenhofer (2021) for an analysis of arbitrageurs' strategies on a Uniswap CFMM). Thus,
on existing CFMMs, in the absence of trading fees, liquidity traders' cumulative costs of
trading equal arbitrageurs' cumulative profits. An implication is that the only way for liquidity providers to obtain positive expected profits and compensate them for impermanent losses that occur in the presence of exchange rate volatility is to charge trading fees.
This is not the case on KirtuSwap, which is able to break the link between liquidity traders'
costs and arbitrageurs' profits even in the absence of trading fees and exchange rate volatility.
On KirtuSwap,arbitrageurs' profits are significantly smaller than liquidity traders' costs of trading.
This implies that liquidity providers on KirtuSwap could, in principle,
obtain positive expected returns even in the absence

of trading fees! (In reality, in the presence of exchange rate volatility, differential fees between real and virtual pools are still required on KirtuSwap to provide correct incentives for arbitrageurs; more on this below.)

The reason behind the ability of KirtuSwap to decouple liquidity traders' costs of trading from arbi-trageurs' profi ts is that liquidity trades through virtual pools tend to be followed by arbitrage trades on real pools. KirtuSwap algorithm provides incentives (in the form of lower trading fees) to arbitrageurs for executing arbitrage trades on real pools, regardless of whether an arbitrage trade follows a liquidity trade or an exogenous change in (external) exchange rate.
The example in the next section illustrates the basics of how KirtuSwap works.
Section 3 presents detailed KirtuSwap algorithm.
Simulations in Section 4 provide realistic estimates of cost savings to traders and gains to liquidity providers. Section 5 illustrates the promise of data-driven approach to setting AMM parameters, which will further reduce trading costs and raise liquidity providers' returns.

2 Gains from trading on KirtuSwap: Example

In this example, we illustrate the workings of KirtuSwap by examining two transactions by liquidity traders, optimal responses by arbitrageurs, and the resulting outcomes – i.e. cumulative gains to liquidity providers and cumulative trading costs of liquidity traders. This example is purposefully simplified with the goal of explaining the main ideas and driving forces behind KirtuSwap fi nancial technology in the most straightforward manner.
This example relies on several assumptions that ease the exposition:

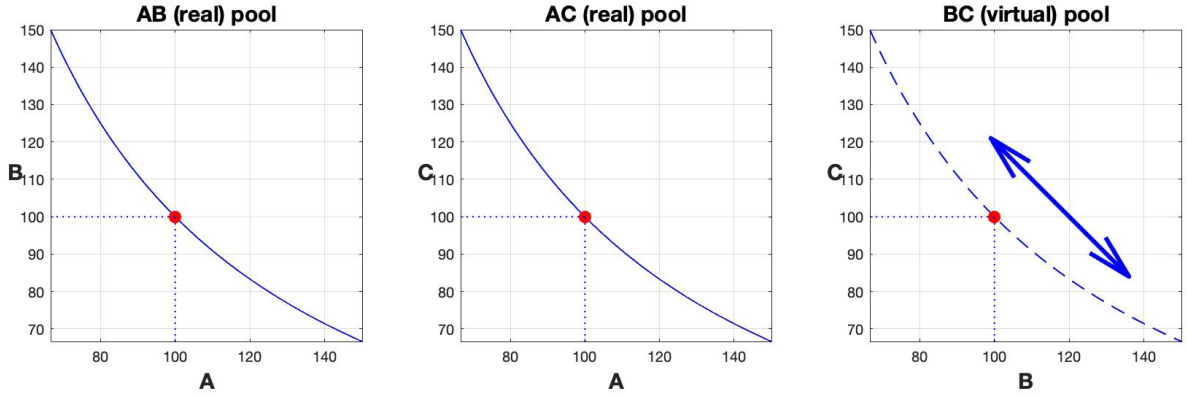
- There are 3 tokens in the AMM: A, B, and C.
- External prices of tokens (used by arbitrageurs) remain constant throughout the example and are normalized to $1 without lack of generality.
- Liquidity providers have provided equal liquidity into 2 pools: AB and AC, of $200 each, i.e. 100 tokes of A and B each into pool AB, and 100 tokens of A and C each into pool AC.
- Trading fees equal zero.

These assumptions are removed and/or generalized in the KirtuSwap algorithm in Section 3 and in numerical simulations in Sections 4 and 5. The example proceeds in several steps.

Step 1: Initial liquidity provision

Initial liquidity provision into AB and AC pools, along with constant product trading curves, are depicted in the first two subplots of Figure 1.

Figure 1: Initial liquidity provision

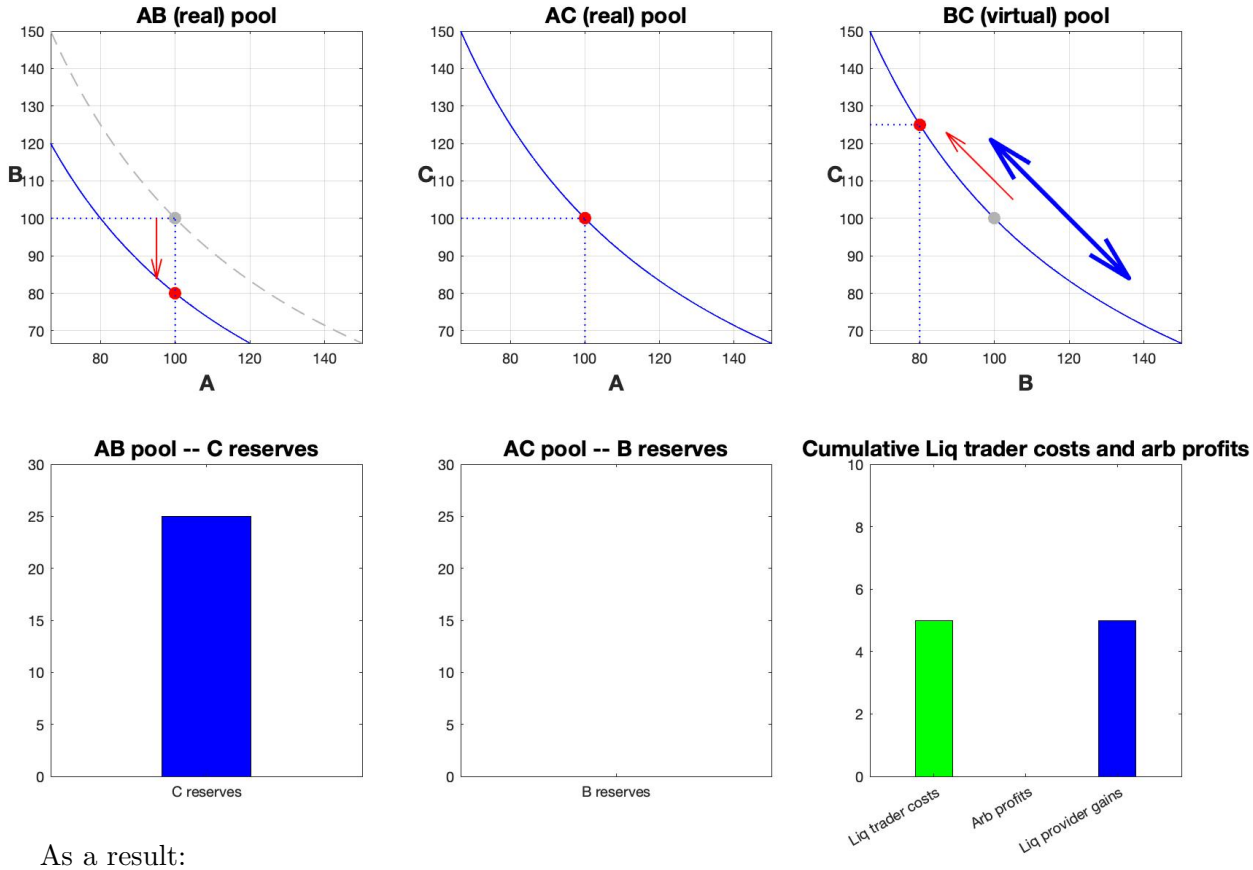


The third subplot presents BC virtual pool, whose composition is based on 1) determining the "common denominator" in real pools (100 A tokens in this case) and 2) determining the amounts of B and C tokens corresponding to this common denominator (100 B and 100 C tokens in this case). The bi-directional blue arrow denotes that trading is possible in virtual pool BC in both directions.

Step 2: Liquidity trade on pool BC (buy 20 B)

Assume that a trader purchases 20 B for C on virtual pool BC, as illustrated in the third subplot of Figure 2.

Figure 2: Liquidity trade -- buy 20 B for C



As a result:

- 20 B are taken from pool AB, as illustrated in the first subplot.
- 25 C are placed in the reserve of C on pool AB, as shown in the fourth subplot (the quantity of C based on BC trading curve: $\Delta C = \frac{B \times C}{C - \Delta C}$).

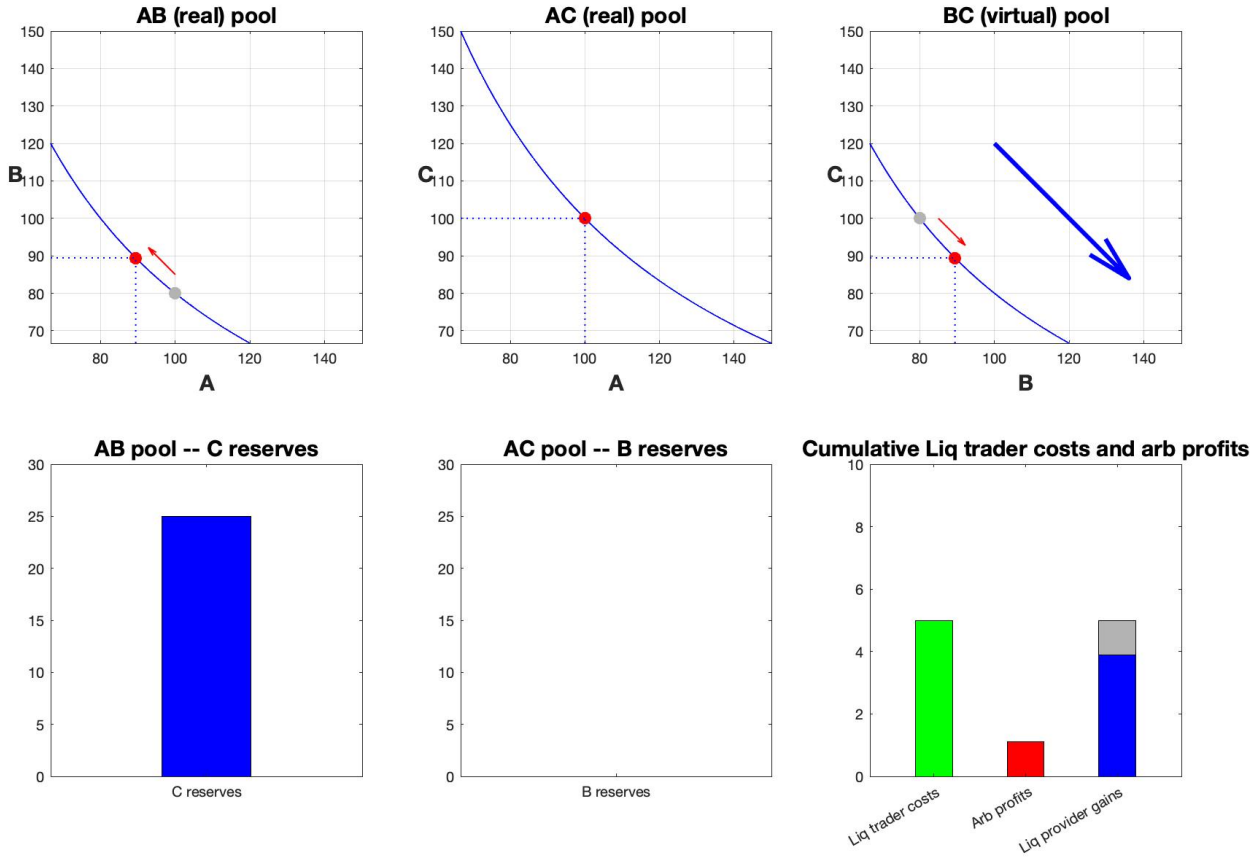
Note that pool AC as well as its reserves are not affected, as shown in the second and fifth subplots, respectively. The sixth subplot shows, in green, liquidity trader's trading costs due to slippage, which equal gains to the liquidity provider to pool AB, $\Delta B \times \$1 + \Delta C \times \$1$.

The fourth subplot in Figure 2 illustrates the inherent risk to liquidity providers on KirtuSwap: the liquidity provider on pool AB is (temporarily) exposed to potential price movements of token C, which is non-native to pool AB. While i) the expected rewards for taking this risk are large, as shown below and ii) the risk is short-lived, it is important to set limits to this risk. This is done by imposing the condition that once the ratio of the value of reserves on non-native tokens in a pool to the value of native tokens exceeds a certain threshold, the pool cannot be used in construction of virtual pools trading on which will result in increasing this ratio. In this case, this ratio is $\frac{\$1 \times C}{\$1 \times A + \$1 \times B} = 13.89\%$. Assume that this reserve ratio exceeds the threshold (say 10%). Thus, until the time when the reserve ratio on pool AB drops below 10%, pool AB cannot be used to facilitate trades on virtual pool BC. As a result, temporarily, only trades in which C is bought and B is sold are now allowed on pool BC.

Step 3: Arbitrage trade on pool AB

The liquidity trade in step 2 created price imbalances (between prices implied on KirtuSwap pools and external prices, which are constant in this example), inviting arbitrage trades. In principle, arbitrage trades can happen on either real pool AB or virtual pool BC (which is not balanced, as evident from the grey dot in the third subplot in Figure 3 – and in which trading in the direction of buying B and selling C is allowed, as shown by the thick blue arrow).

Figure 3: Arbitrage on pool AB and virtual pool adjustment



It is easy to show that an arbitrage trade on virtual pool BC does not entirely eliminate the price imbalance on real pool AB, inviting further arbitrage trades, increasing arbitrageur's profits and reducing liquidity providers' returns in the process. Thus, it is very important that the arbitrage trades happen on real pools and not on virtual ones. While in this example, this is an assumption, in reality this will be achieved by setting differential fees for trading on real vs virtual pools. It is crucial that fees on virtual pools are higher than those on real pools – sufficiently so as to make arbitrage on virtual pools unprofitable relative to arbitrage on real pools. However, the difference in fees should not be too large to make trading on virtual pools more expensive than trading using an indirect route through multiple real pools. In this example, any ratio of virtual pool fees to real pool fees between 1 and 2 satisfies both restrictions.

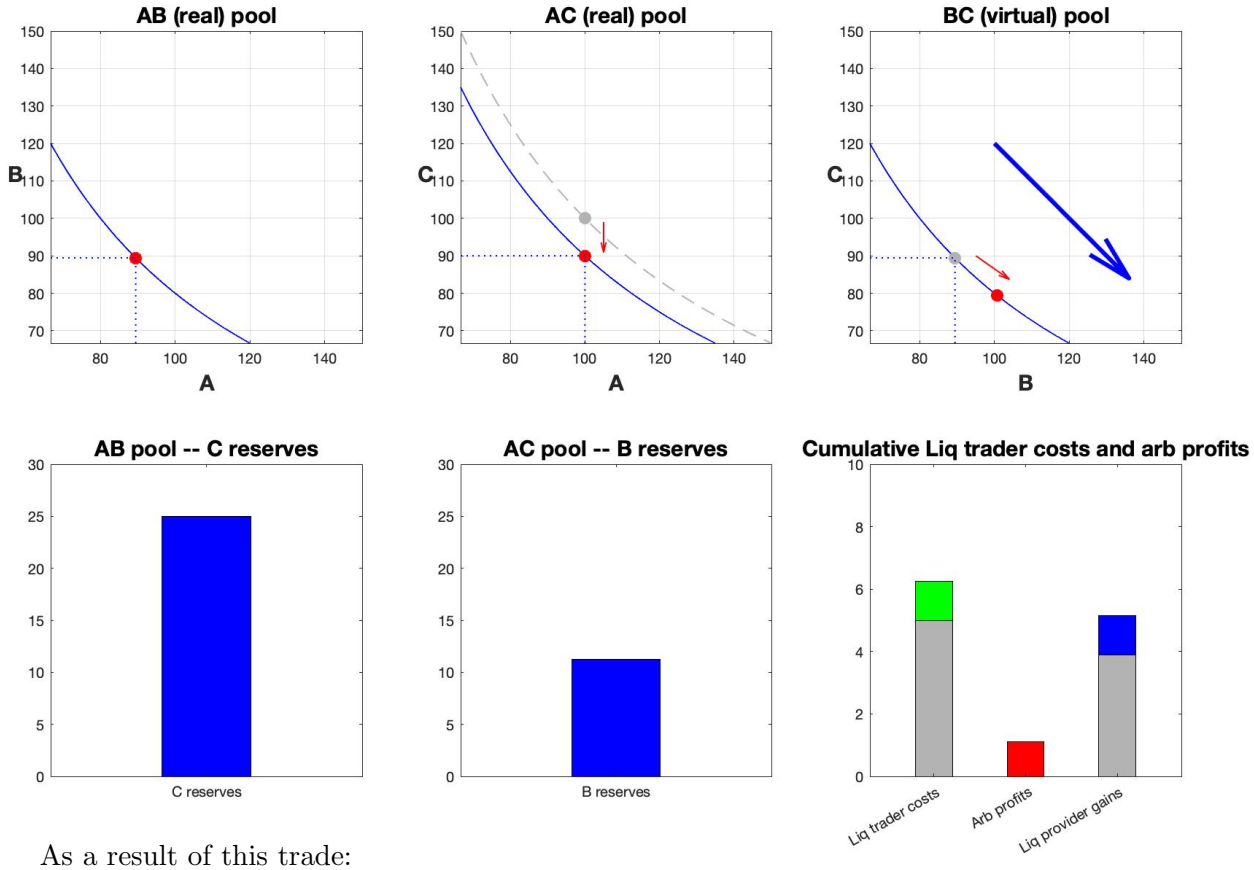
In the remainder of this example, we assume that incentives are in place such that arbitrageurs execute arbitrage trades on real pools. The first subplot of Figure 3 illustrates the arbitrage trade on pool AB.

Pool AC is not updated, whereas virtual pool BC is, as shown in subplot 3. The arbitrageur profits from the trade, as shown in the red bar in the sixth subplot. This reduces the net cumulative gain to liquidity providers, as shown by the blue bar in the sixth subplot.

Step 4: Liquidity trade on pool BC (buy 10 C)

As the pool BC is now temporarily unidirectional, i.e. only buying B and selling C is allowed, the next liquidity trade will be in this direction. Assume that a trader purchases 10 C for B on virtual pool BC, as illustrated in the third subplot of Figure 4.

Figure 4: Liquidity trade -- buy 10 C for B



As a result of this trade:

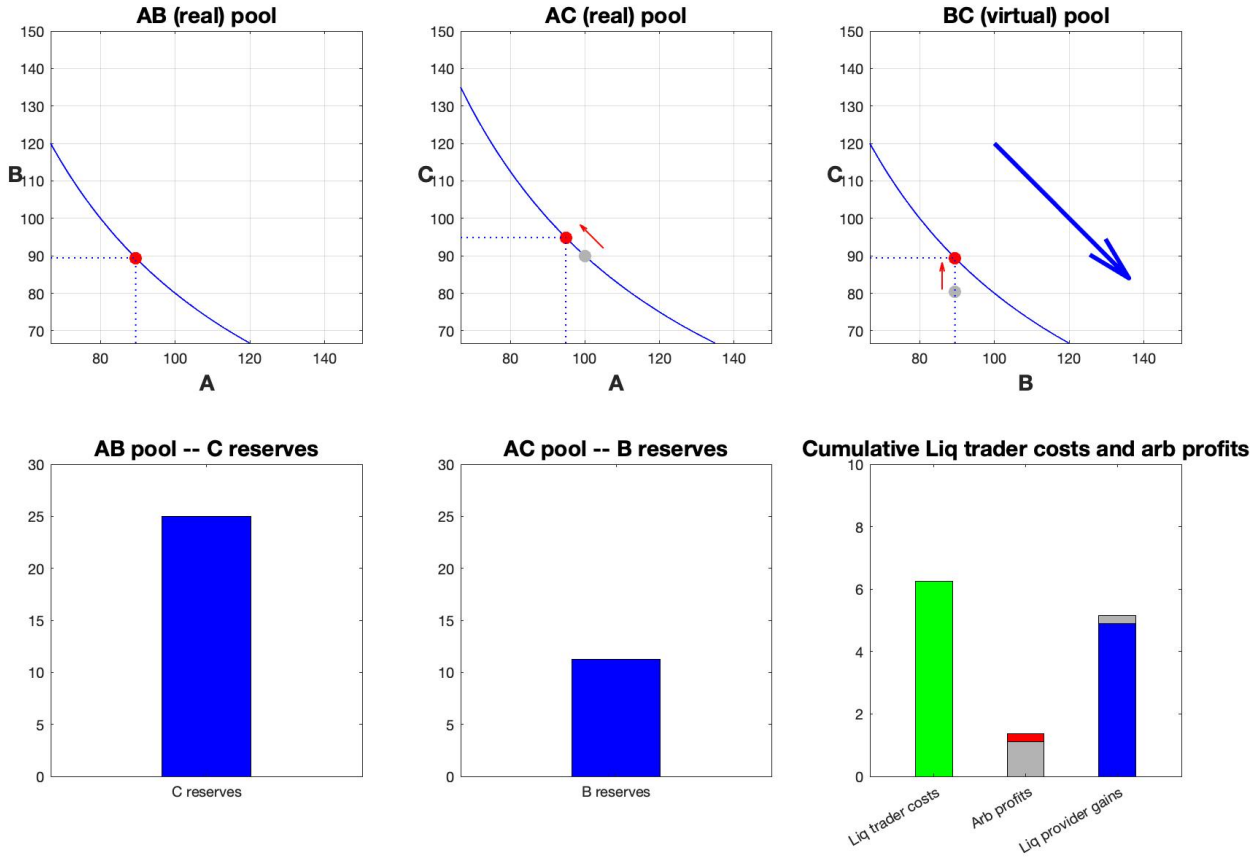
- 10 C are taken from pool AC, as illustrated in the second subplot.
- 11.26 B are placed in the B reserve of pool AC (based on BC trading curve, as shown in the fifth subplot).

Pool AB as well as its reserves are not affected, as shown in the first and fourth subplots, respectively. The sixth subplot shows, in green, the increase in total cumulative costs of liquidity traders and the increase in the cumulative net gains of liquidity providers. Note that the value of B reserves on pool AC ($\frac{\$1 \times B}{\$1 \times A + \$1 \times C}$=5.92%) does not exceed the reserve threshold of pool AC, thus, trading on virtual pool BC in the direction of buying C and selling B is still allowed, as shown by the blue arrow in the third subplot.

Step 5: Arbitrage trade on pool AC

The first subplot of Figure 5 illustrates the arbitrage trade on pool AC.

Figure 5: Arbitrage on pool AC and virtual pool adjustment

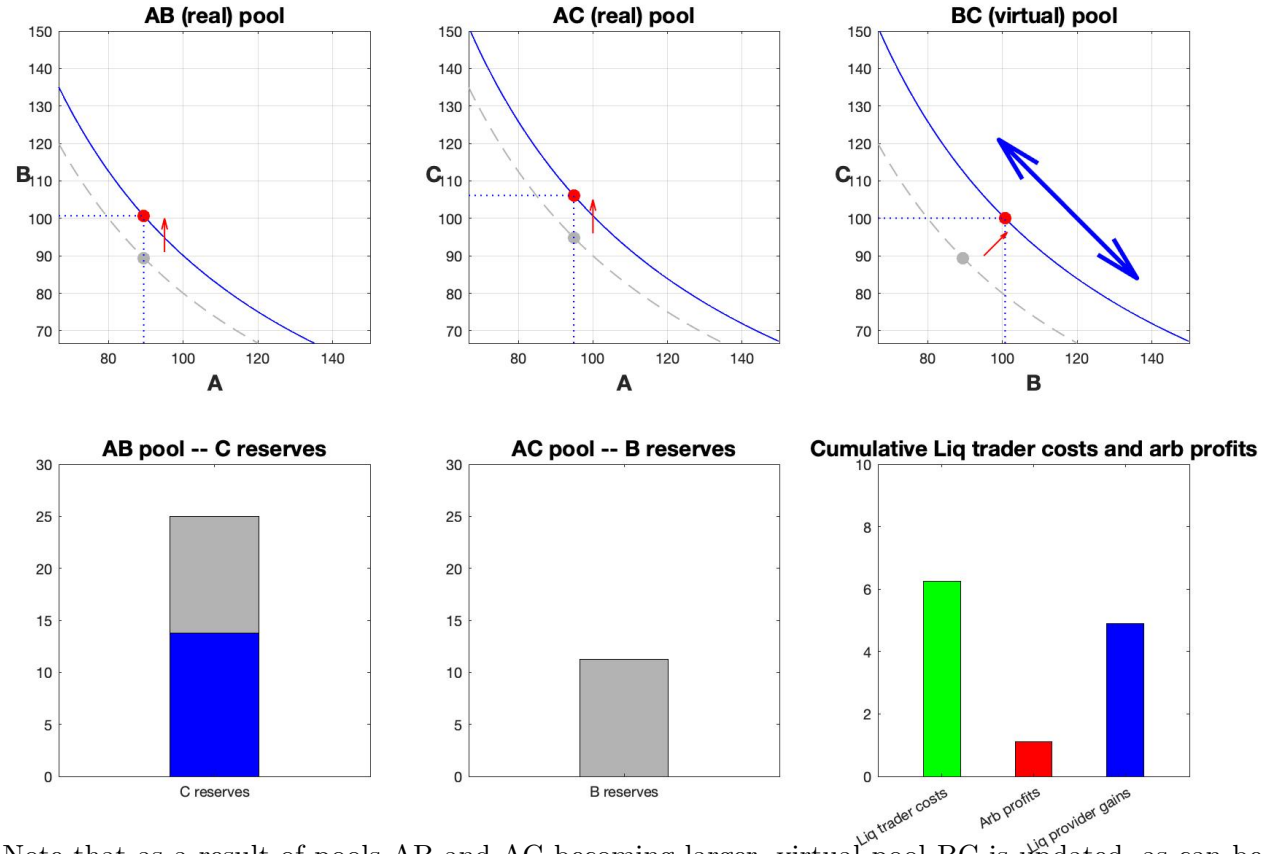


Pool AB is not updated, whereas virtual pool BC is, as shown in the third subplot. The arbitrageurs profit from the trade, as shown in the red bar in the sixth subplot. Arbitrageurs' profit reduces the net cumulative gain to liquidity providers, as shown in the blue bar in the sixth subplot.

Step 6: Exchange between reserves of pools B and C

Pool AB contains reserve of C, whereas pool AC contains reserve of B. Whenever this happens, (part) of the reserves is returned to their native pools. The exchange rate between B and C can be determined from quantities of B and C in the virtual pool BC. In this case, the quantities of B and C in virtual pool BC are identical, thus the exchange rate is 1:1. As a result the entire reserve of B moves to pool AB and an identical quantity of C from its reserve on pool AB moves to pool AC, as illustrated in the first two subplots in Figure 6.

Figure 6: Exchange of reserves between pools AB and AC

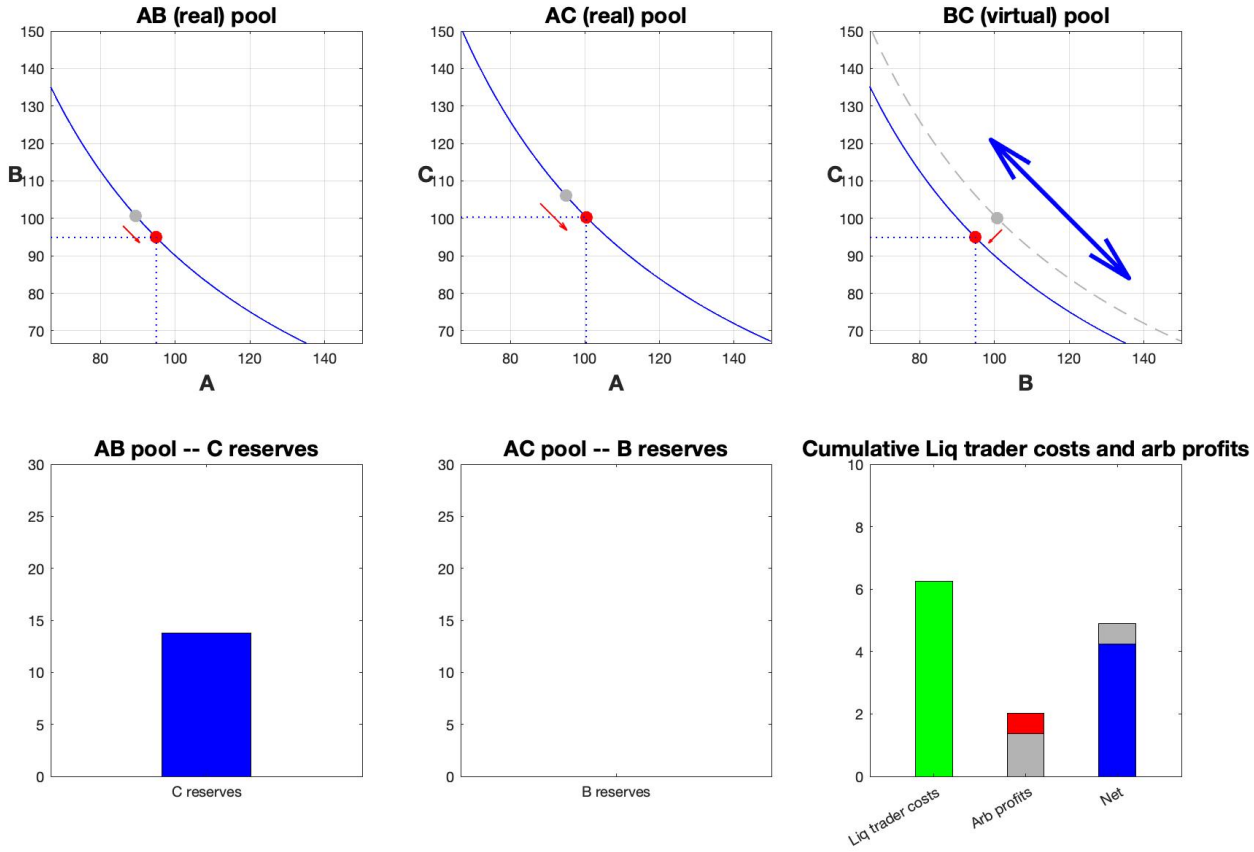


Note that as a result of pools AB and AC becoming larger, virtual pool BC is updated, as can be seen in the third subplot. In addition, since the reserve ratio of both real pools is now below the reserve limit, trades in both directions in virtual pool BC are now allowed, as shown in the third subplot.

Step 7: Arbitrage trades on pools AB and AC

As a result of the exchange of reserves, both real pools AB and AC are now unbalanced, inviting arbitrage trades on both. Arbitrage trades in AB and AC, their effects on virtual pool BC and on arbitrageurs' cumulative profits and liquidity providers' cumulative gains are illustrated in Figure 7.

Figure 7: Arbitrage on pools AB and AC, and virtual pool adjustment

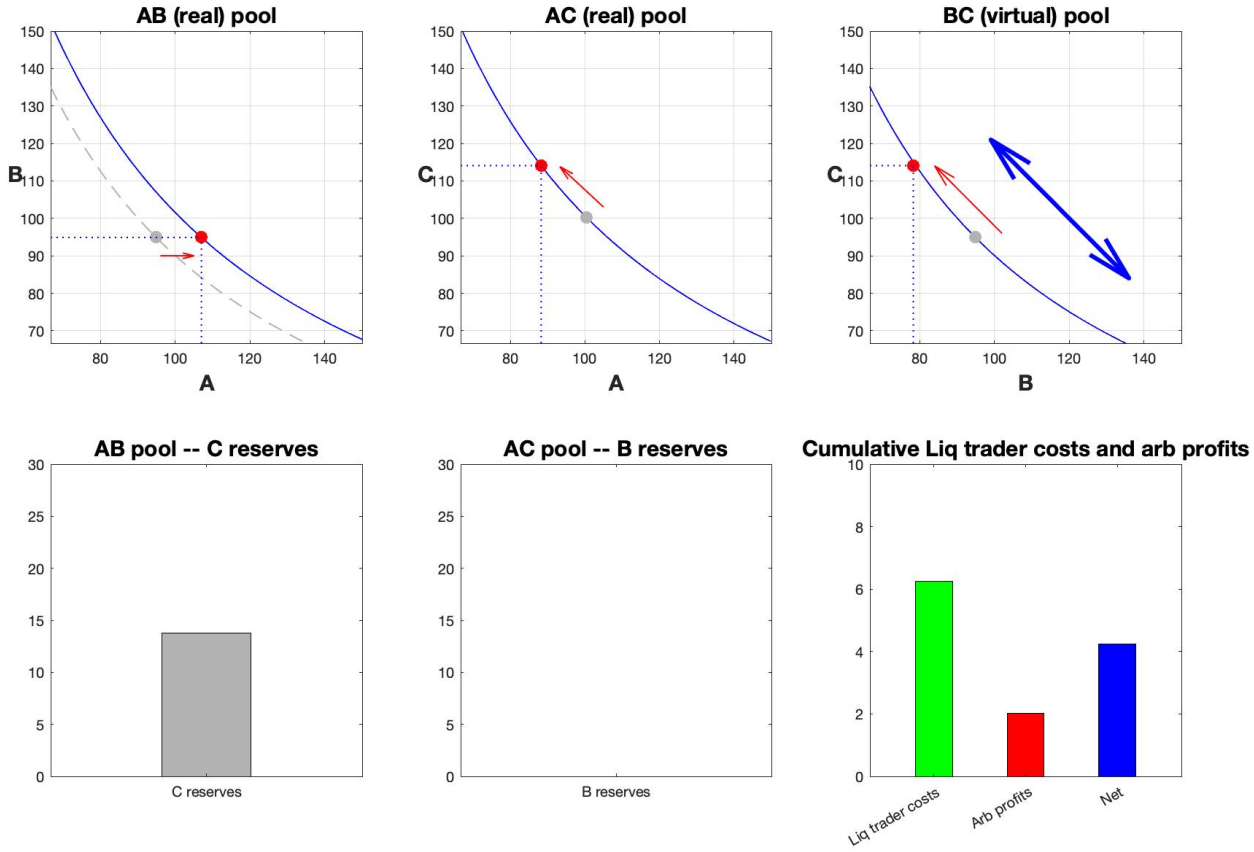


Step 8: Emptying of reserves on pool AB

As we would like to limit liquidity providers' exposure to price changes of tokens non-native to their pools, at the end of a pre-determined time interval, all remaining reserves in pools are emptied and exchanged for the pools' native tokens. Assuming no other trades occur prior to reaching the reserve emptying time, the reserve of C on pool AB is emptied. The process is as follows:

- The entire reserve of C is moved to pool AC, as illustrated in the fourth subplot in Figure 8.
- A is withdrawn from pool AC based on the trading curve on AC, as illustrated in the second subplot.
- A tokens withdrawn from pool AC are moved to pool AB, as illustrated in the first subplot.
- Virtual pool BC is updated as a result, as illustrated in the third subplot.

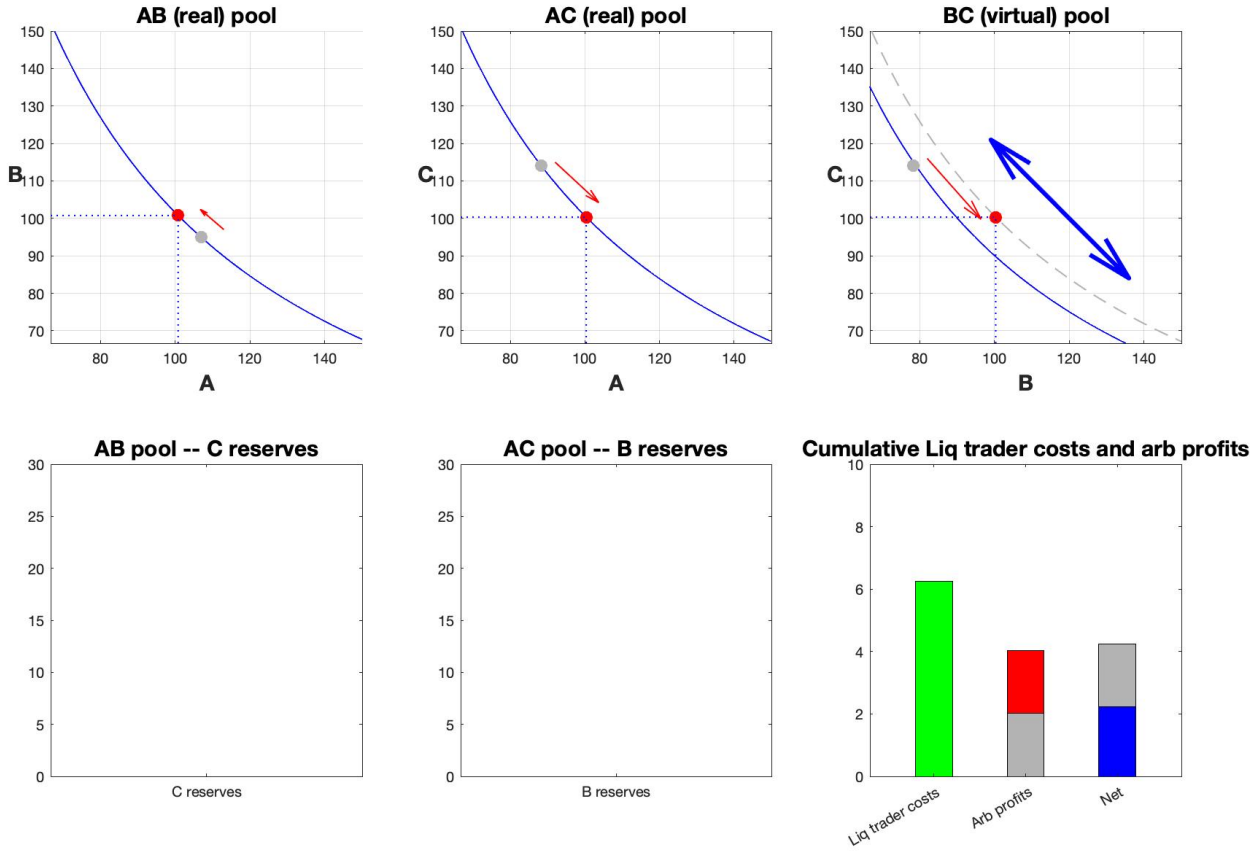
Figure 8: Emptying of C reserves on pool AB



Step 9: Arbitrage trades on pools AB and AC

As a result of emptying the reserves of C on pool AB, both real pools AB and AC are now unbalanced, inviting arbitrage trades on both. Arbitrage trades, their effects on virtual pool BC and on arbitrageurs' cumulative profits and liquidity providers' cumulative gains are illustrated in Figure 9.

Figure 9: Arbitrage on pools AB and AC, and virtual pool adjustment



Note the following outcomes. First, both real pools AB and AC contain only tokens native to them. Second, both real pools AB and AC and virtual pool BC are balanced, inviting no more arbitrage trades.

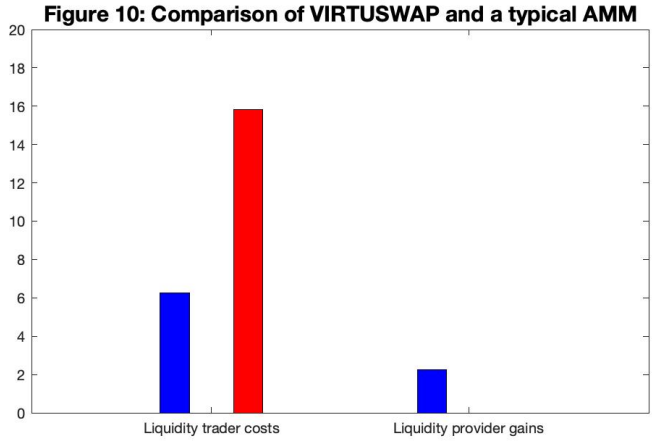
Comparison of outcomes of KirtuSwap and a traditional AMM

Figure 10 compares cumulative costs to liquidity traders and liquidity providers' cumulative gains on KirtuSwap and the corresponding quantities on a traditional AMM,
in which trades between B and C are executed using indirect routes (through A in this example).
Blue bars correspond to KirtuSwap, whereas red bars correspond to a traditional AMM.
Figure 10 illustrates the following main results:

- Due to direct trading on virtual pool, the overall trading costs due to slippage, paid by liquidity traders, are significantly lower on KirtuSwap.
- Since arbitrage following liquidity trades on a virtual pool happens on real pools, arbitrageurs' profits on KirtuSwap are lower than the costs of trading of liquidity traders,
  resulting in positive
  net cumulative gains to liquidity providers even in the absence of trading fees. This is not the case for a traditional AMM, on which in the absence of trading fees and exchange rate volatility, arbitrageurs' profits always equal liquidity traders' costs of trading, resulting in zero net gains to liquidity providers.

Relaxing assumptions in the example

While the example above is illustrative, it is based on several assumptions, which are restrictive to varying degrees. In the next two sections (Algorithm and Simulations), we focus on the general, real-life situation,



characterized by:

- An arbitrary number of tokens;
- Volatile external token prices and resulting external exchange rates;
- Unequal real pools;
- Pools having both real and virtual components;
- Reserves of multiple non-native tokens on each pool;
- Non-zero differential trading fees on both real and virtual components of pools.

3 KirtuSwap algorithm

3.1 Notation

- There are N tokens and resulting $\frac{N(N-1)}{2}$ possible trading pairs ("pools").
- Each pool is denoted by R_{ij} where i and j denote any two tokens. R stands for "real pool", as we explain below. For notation purposes, $R_{ij} = R_{ji}$ and either can be used interchangeably.
- The amount of token k in real pool R_{ij} is denoted by $k(R_{ij})$. As we explain below, each real pool will contain tokens native to this pool, i.e. $i(R_{ij})$ and $j(R_{ij})$ for pool R_{ij}; each real pool can also contain reserves of other tokens (non-native to the pool), i.e. $k \notin \{i, j\}(R_{ij})$.
- $\rho_{ij} = \rho_{ji}$ is the reserve ratio of pool R_{ij}, equaling the combined value of tokens $k \notin \{i, j\}(R_{ij})$ not native to pool R_{ij} to the combined value of tokens native to the pool, $i(R_{ij})$ and $j(R_{ij})$.
- $\overline{\rho_{ij}} = \overline{\rho_{ji}}$ is the upper limit of reserve ratio of pool R_{ij}.
- Below, we will also introduce "virtual pools", denoted by V_{ij} for buying token i and selling token j. Note that V_{ji} does not necessarily equal V_{ij}, i.e. virtual pools are direction-specific.
- We will also introduce "total pools", denoted by T_{ij} (which is not necessarily equal to $= T_{ji}$).

- $\eta_{ij} = \frac{1}{\eta_{ji}}$ is the ratio of the exchange rate between i and j implied from the composition of R_{ij}, $\frac{j(R_{ij})}{i(R_{ij})}$, and the exchange rate implied from the composition of V_{ij}, $\frac{j(V_{ij})}{i(V_{ij})}$: $\eta_{ij} = \frac{1}{\eta_{ji}} = \frac{j(R_{ij})/i(R_{ij})}{j(V_{ij})/i(V_{ij})}$ if $i(R_{ij}) > 0$, $j(R_{ij}) > 0$, $i(V_{ij}) > 0$, and $j(V_{ij}) > 0$, and $\eta_{ij} = 0$ otherwise.
- $\overline{\eta_{ij}}$ is the upper limit of the proportional deviation of η_{ij} from one.
- There are n liquidity providers (LPs), indexed by X.
- The number of tokens of LP X, defining her ownership of real pool R_{ij}, is denoted by $t_{X,ij}$.
- The proportional trading fee in the real part of T_{ij} is denoted by $f_{R_{ij}}$ and the proportional trading fee in the virtual part of T_{ij} is denoted by $f_{V_{ij}}$.
- H_i is a temporary holding tank containing token i.
- Throughout, $\tilde{z}$ denotes the updated value of variable z.

3.2 Pool composition and availability of virtual pools

Virtual pools are created under certain conditions outlined in this subsection.

3.2.1 General availability of virtual pools

Pool R_{ij} may be defined as available to contribute liquidity to virtual pools V_{ik} where $k \notin \{i, j\}$ or unavailable to contribute liquidity to virtual pools. An indicator $\mathbb{I}_{ij}^{provide}$ equals one in the former case and equals zero in the latter.

3.2.2 Availability of reserve pools based on reserve ratios

At any point in time, pool $R_{ij} = R_{ji}$ is composed of native tokens, $i(R_{ij})$ and $j(R_{ij})$, as well as, potentially, reserves of non-native tokens, $k \notin \{i, j\}(R_{ij})$. The ratio of the total value of reserves of non-native tokens to the total value of native tokens, i.e. the reserve ratio of pool R_{ij}, is given by:

$$\rho_{ij} = \frac{\sum_{k \notin \{i,j\}} \left(k(R_{ij}) \max\left(\frac{i(R_{ik})}{\max(k(R_{ik}),\epsilon)}, \frac{j(R_{jk})}{\max(k(R_{jk}),\epsilon)} \times \frac{i(R_{ij})}{\max(j(R_{ij}),\epsilon)} \right) \right)}{2 \max(i(R_{ij}), \epsilon)}, \tag{1}$$

where $\epsilon \to 0$ from above (in order to allow division in cases in which $k(R_{ik}) = 0$ or $k(R_{jk}) = 0$ (i.e. real pool R_{ik} or R_{jk} does not exist); note that in such cases this division does not matter as the numerator, $i(R_{ij}$ ir $j(R_{ij})$ equals zero). The expression in the maximand in the numerator is the exchange rate of k to i, obtained via the composition of either R_{ik} or R_{jk}.

If the reserve ratio of pool R_{ij}, ρ_{ij} exceeds its upper limit, $\overline{\rho_{ij}}$, then the indicator for reserve of pool ij exceeding its limit, $\mathbb{I}_{ij}^{reserve}$ equals zero, and it equals one otherwise.

$$\mathbb{I}_{ij}^{reserve} = 1 \text{ if } \rho_{ij} < \overline{\rho_{ij}}, \tag{2}$$

$$\mathbb{I}_{ij}^{reserve} = 0 \text{ if } \rho_{ij} \geq \overline{\rho_{ij}}. \tag{3}$$

3.2.3 Availability of reserve pools based on balance between real and virtual pools

At any point in time, the indicator of balance between exchange rates derived from real and virtual pools, η_{ij} for currency pair ij is a function of the deviation of η_{ij} from one. It equals one if the composition of virtual pool V_{ij} is proportional to the composition of the real pool R_{ij}, or deviates from one otherwise. Large deviations may lead to diverging arbitrage opportunities, severely increasing arbitrageurs' profits and liquidity providers' resulting impermanent losses. Thus, we define an indicator for pool balance that equals one if the deviation of η_{ij} from one is sufficiently small (i.e., the pool is "balanced") and that equals zero otherwise (i.e. the pool is "unbalanced"):

$$\mathbb{I}_{ij}^{balance} = 1 \text{ if } 1 - \overline{\eta_{ij}} < \eta_{ij} < \frac{1}{1 - \overline{\eta_{ij}}}, \tag{4}$$

$$\mathbb{I}_{ij}^{balance} = 0 \text{ if } \eta_{ij} \leq 1 - \overline{\eta_{ij}} \text{ or } \eta_{ij} \geq \frac{1}{1 - \overline{\eta_{ij}}}. \tag{5}$$

3.2.4 Using virtual pools

Any currency pair ij may be using virtual pools, provided by real pools R_{ik}, $k \notin \{i, j\}(R_{ij})$, in which case an indicator $\mathbb{I}_{ij}^{use}$ equals one (and it equals zero if pair ij is not using virtual pools).

3.3 Computation/updating of virtual and total pools

At any time when one or more real pools are updated for any reason (due to initial or subsequent deposit into a real pool, withdrawal from the pool, or trading in the pool), virtual pools are computed/updated. Virtual pool of tokens i and j, V_{ij} is computed as follows:

$$i(V_{ij}) = \mathbb{I}_{ij}^{use} \times \sum_{k \notin \{i,j\}} \left(\mathbb{I}_{ik}^{provide} \times \mathbb{I}_{ik}^{reserve} \times \mathbb{I}_{ik}^{balance} \times i(R_{ik}) \frac{\min(k(R_{ik}), k(R_{jk}))}{\max(k(R_{ik}), \epsilon)} \right), \tag{6}$$

$$j(V_{ij}) = \mathbb{I}_{ij}^{use} \times \sum_{k \notin \{i,j\}} \left(\mathbb{I}_{ik}^{provide} \times \mathbb{I}_{ik}^{reserve} \times \mathbb{I}_{ik}^{balance} \times j(R_{jk}) \frac{\min(k(R_{ik}), k(R_{jk}))}{\max(k(R_{jk}), \epsilon)} \right), \tag{7}$$

$$i(V_{ji}) = \mathbb{I}_{ji}^{use} \times \sum_{k \notin \{i,j\}} \left(\mathbb{I}_{jk}^{provide} \times \mathbb{I}_{jk}^{reserve} \times \mathbb{I}_{jk}^{balance} \times i(R_{ik}) \frac{\min(k(R_{ik}), k(R_{jk}))}{\max(k(R_{ik}), \epsilon)} \right), \tag{8}$$

$$j(V_{ji}) = \mathbb{I}_{ji}^{use} \times \sum_{k \notin \{i,j\}} \left(\mathbb{I}_{jk}^{provide} \times \mathbb{I}_{jk}^{reserve} \times \mathbb{I}_{jk}^{balance} \times j(R_{jk}) \frac{\min(k(R_{ik}), k(R_{jk}))}{\max(k(R_{jk}), \epsilon)} \right). \tag{9}$$

Whether pool R_{ik} can contribute i to any virtual pool depends on whether: i) pool R_{ik} is defined as providing liquidity to other pools, i.e. $\mathbb{I}_{ik}^{provide} = 1$; ii) the reserve ratio of pool R_{ik} is below its upper limit, i.e. $\mathbb{I}_{ik}^{reserve} = 1$; and iii) the real and virtual pools R_{ik} and V_{ik} are sufficiently balanced, i.e. $\mathbb{I}_{ik}^{balance} = 1$. Importantly, virtual pool V_{ij} is a function of real pools R_{ik} ($k \notin \{i, j\}$) and indicators $\mathbb{I}_{ik}$, whereas virtual pool V_{ji} is a function of real pools R_{jk} and indicators $\mathbb{I}_{jk}$. As a result, it is possible that virtual pools are asymmetric: ($i(V_{ij} \neq i(V_{ji})$ and $j(V_{ij} \neq j(V_{ji})$).

At any point in time, total pool T_{ij} is computed as follows:

$$i(T_{ij}) = i(R_{ij}) + i(V_{ij}), \tag{10}$$

$$j(T_{ij}) = j(R_{ij}) + j(V_{ij}). \tag{11}$$

Note that since V_{ij} does not have to equal V_{ji}, T_{ij} does not have to equal T_{ji}.

Proportional trading fees in total pool T_{ij} are computed as:

$$f_{T_{ij}} = \frac{i(R_{ij})f_{R_{ij}} + i(V_{ij})f_{V_{ij}}}{i(T_{ij})}. \tag{12}$$

3.4 Initial liquidity provision and addition of liquidity to a pool

If LP X is the first to provide liquidity into pool R_{ij} by supplying $i(R_{ij})$ and $j(R_{ij})$, she receives one token of the pool R_{ij}, i.e. $t_{X,ij} = 1$.

Assume LP X wishes to add liquidity into an existing pool R_{ij}. To do so, she needs to add tokens i and j, i_X and j_X, respectively, proportional to current pool composition, $i(R_{ij})$ and $j(R_{ij})$, i.e.

$$j_X = i_X \frac{j(R_{ij})}{i(R_{ij})}. \tag{13}$$

The updated pool, $\tilde{i}(R_{ij})$ and $\tilde{j}(R_{ij})$, and the updated number of tokens of pool R_{ij} that LP X owns, $\tilde{t}_{X,ij}$, equal:

$$\tilde{i}(R_{ij}) = i(R_{ij}) + i_X, \tag{14}$$

$$\tilde{j}(R_{ij}) = j(R_{ij}) + j_X, \tag{15}$$

$$\tilde{t}_{X,ij} = t_{X,ij} + i_X \frac{\sum_I t_{I,ij}}{i(R_{ij})(1 + \frac{1}{2}\rho_{ij})}, \tag{16}$$

where the values of both native and non-native tokens in pool R_{ij} prior to the liquidity provision are taken into account when computing $\tilde{t}_{X,ij}$.

Note that after addition of liquidity to the pools, pools' reserve ratios, virtual and total pools, as well as proportional fees need to be updated, as in Sections 2 and 3.

3.5 Trading

Assume a trader wants to buy β units of token i in exchange for token j. If real pool, R_{ij}, and/or virtual pool, V_{ij}, are available and the size of the total pool, T_{ij} exceeds the size of the trade, i.e. $\beta < i(T_{ij})$, then the trade can happen via the total pool.

However, it may also be possible to perform the trade using the traditional way. First, if $\beta < i(R_{ij})$, a direct trade between i and j may be performed along the trading curve of real pool R_{ij}. Second, it is possible to route the trade through any currency $k \notin \{i, j\}$.

Depending on the composition of real and virtual pools and on real and virtual trading fee components, each of the three options – i) trading on the curve defined by the total pool; ii) trading on the curve defined by the real pool; and iii) routing the trade through a different currency, may lead to lowest trading costs. Thus, we need to compare the costs of these options. Notably, however, option ii) above will not be entertained by KirtuSwap algorithm.
The reason is that if the liquidity trade is performed on the real pool curve,
whereas resulting arbitrage is performed on the combination of real and virtual pools,
then arbitrage profits are likely to be larger than liquidity traders' costs of trading. As a result, liquidity providers' returns may turn negative. Thus, in what follows, we compare option i) (trading on total pool) and iii) (indirect trading) as in a traditional AMM.

If trading is performed using the curve of the total pool, T_{ij}, then the cost (in units of currency j for buying β units of currency i) is:

$$\Delta j_{T_{ij}} = \left(\frac{i(T_{ij})j(T_{ij})}{i(T_{ij}) - \beta} - j(T_{ij}) \right) (1 + f_{T_{ij}}). \tag{17}$$

If trading is routed through real pools involving any currency $k \notin \{i, j\}$, then the cost of buying β units of currency i is:

$$\Delta j_{R_{ik},R_{kj}} = \left(\frac{j(R_{kj})k(R_{kj})}{k(R_{kj}) - \left(\frac{i(R_{ik})k(R_{ik})}{i(R_{ik})-\beta} - k(R_{ik} \right) (1 + f_{R_{ik}})} - j(R_{ij}) \right) (1 + f_{R_{kj}}). \tag{18}$$

If $\Delta j_{T_{ij}} < \max_k \Delta j_{R_{ik},R_{kj}}$ then the trade is performed using the curve of the total pool, T_{ij}, whereas in the opposite case it is routed through currency k, i.e. executed on real pools R_{ik} and R_{kj}.

If the trade takes place on the total pool, T_{ij}, the following steps need to be executed:

1. Trading along the curve defined by the total pool T_{ij}:

$$\tilde{i}(T_{ij}) = i(T_{ij}) - \beta(1 - f_{T_{ij}}), \tag{19}$$

$$\tilde{j}(T_{ij}) = \frac{i(T_{ij})j(T_{ij})}{i(T_{ij}) - \beta}. \tag{20}$$

2. Updating native and non-native real pools that result from trading on the total pool.

   (a) Updating the native pool, R_{ij}:

$$\tilde{i}(R_{ij}) = \tilde{i}(T_{ij}) \frac{i(R_{ij})}{i(T_{ij})}, \tag{21}$$

$$\tilde{j}(R_{ij}) = \tilde{j}(T_{ij}) \frac{j(R_{ij})}{j(T_{ij})}. \tag{22}$$

   (b) Updating other, non-native pools that contribute to $i(V_{ij})$. For every $k \notin \{i, j\}$ perform:

$$\tilde{i}(R_{ik}) = i(R_{ik}) + ((\tilde{i}(T_{ij}) - i(T_{ij})) - (\tilde{i}(R_{ij}) - i(R_{ij}))) \frac{i(R_{ik})}{\sum_{l \notin \{i,j\}} i(R_{il})}, \tag{23}$$

   (c) Updating reserves of j in non-native pools from which i was taken to execute the part of trade on V_{ij}. For every $k \notin \{i, j\}$ perform:

$$\tilde{j}(R_{ik}) = j(R_{ik}) + ((\tilde{j}(T_{ij}) - j(T_{ij})) - (\tilde{j}(R_{ij}) - j(R_{ij}))) \frac{i(R_{ik})}{\sum_{l \notin \{i,j\}} i(R_{il})}. \tag{24}$$

If the trade is routed through currency k, real pools R_{ik} and R_{jk} have to be updated accordingly.

Finally, updating of reserve ratios, virtual and total pools, as well as proportional fees, as in subsections 3.2 and 3.3, needs to be performed.

3.6 Return of reserves to native pools

3.6.1 Continuous return of reserves

When a situation arises in which $k(R_{ij}) > 0$ and $j(R_{ik}) > 0$, (some of) j and k can be moved from reserves on non-native pools to their native pools:

$$\tilde{j}(R_{ij}) = j(R_{ij}) + \min\left(j(R_{ik}), k(R_{ij})\frac{j(R_{ij})}{i(R_{ij})}\frac{i(R_{ik})}{j(R_{ik})}\right), \tag{25}$$

$$\tilde{j}(R_{ik}) = j(R_{ik}) - \min\left(j(R_{ik}), k(R_{ij})\frac{j(R_{ij})}{i(R_{ij})}\frac{i(R_{ik})}{j(R_{ik})}\right), \tag{26}$$

$$\tilde{k}(R_{ik}) = k(R_{ik}) + \min\left(k(R_{ij}), j(R_{ik})\frac{k(R_{ik})}{i(R_{ik})}\frac{i(R_{ij})}{k(R_{ij})}\right), \tag{27}$$

$$\tilde{k}(R_{ij}) = k(R_{ij}) - \min\left(k(R_{ij}), j(R_{ik})\frac{k(R_{ik})}{i(R_{ik})}\frac{i(R_{ij})}{k(R_{ij})}\right). \tag{28}$$

As with any updating of real pools, reserve ratios, virtual and total pools, as well as proportional fees need to be updated, as in Sections 2 and 3.

If more than one return of reserves is possible (i.e. there are multiple pairs $k(R_{ij}) > 0$ and $j(R_{ik}) > 0$), we repeat this step until there are no pairs such that $k(R_{ij}) > 0$ and $j(R_{ik}) > 0$.

3.6.2 Periodic return of reserves

Every given interval, τ_{ij} for every pool R_{ij} (where τ_{ij} can be either pool-specific or equal for all pools), the remaining reserves in pool R_{ij} will be emptied, i.e. substituted for the pool's native tokens. For every $k \notin \{i, j\}$:

$$\tilde{k}(R_{ij}) = 0, \tag{29}$$

$$\tilde{k}(R_{ik}) = k(R_{ik}) + k(R_{ij})\frac{k(R_{ik})}{\max(k(R_{ik}) + k(R_{jk}), \epsilon)}, \tag{30}$$

$$\tilde{k}(R_{jk}) = k(R_{jk}) + k(R_{ij})\frac{k(R_{jk})}{\max(k(R_{ik}) + k(R_{jk}), \epsilon)}, \tag{31}$$

$$\tilde{i}(R_{ik}) = \frac{i(R_{ik})k(R_{ik})}{\max(\tilde{k}(R_{ik}), \epsilon)}, \tag{32}$$

$$\tilde{j}(R_{jk}) = \frac{j(R_{jk})k(R_{jk})}{\max(\tilde{k}(R_{jk}), \epsilon)}, \tag{33}$$

$$\tilde{i}(R_{ij}) = i(R_{ij}) + (i(R_{ik}) - \tilde{i}(R_{ik})), \tag{34}$$

$$\tilde{j}(R_{ij}) = j(R_{ij}) + (j(R_{ik}) - \tilde{i}(R_{ik})). \tag{35}$$

3.7 Withdrawal of liquidity

Assume LP X wishes to remove proportion ω of the liquidity she has provided to pool R_{ij}. The following steps need to be executed:

1. Sale of proportional share of tokens i and j from all non-native pools on real pool R_{ij}:

(a) Removal of proportional share of i from every real pool R_{kl} where $k \neq i$ and $l \neq i$ and of j from every real pool R_{kl} where $k \neq j$ and $l \neq j$:

$$\tilde{i}(R_{kl}) = i(R_{kl}) \times \left(1 - \frac{\omega t_{X,ij}}{\sum_I t_{I,ij}}\right), \tag{36}$$

$$TotalRem_i = \sum_{m \neq i, n \neq i, n > m} (i(R_{mn}) - \tilde{i}(R_{mn})), \tag{37}$$

$$RemShare_i(R_{kl}) = \frac{i(R_{kl}) - \tilde{i}(R_{kl})}{TotalRem_i}, \tag{38}$$

$$\tilde{j}(R_{kl}) = j(R_{kl}) \times \left(1 - \frac{\omega t_{X,ij}}{\sum_I t_{I,ij}}\right), \tag{39}$$

$$TotalRem_j = \sum_{m \neq j, n \neq j, n > m} (j(R_{mn}) - \tilde{j}(R_{mn})), \tag{40}$$

$$RemShare_j(R_{kl}) = \frac{j(R_{kl}) - \tilde{j}(R_{kl})}{TotalRem_j}. \tag{41}$$

(b) Substitution of removed i from all non-native pools on real pool R_{ij} and putting resulting j into a "holding tank" H_j:

$$\tilde{i}(R_{ij}) = i(R_{ij}) + TotalRem_i, \tag{42}$$

$$\tilde{j}(R_{ij}) = \frac{i(R_{ij}) \times j(R_{ij})}{\tilde{i}(R_{ij})}, \tag{43}$$

$$H_j = \tilde{j}(R_{ij}) - j(R_{ij}). \tag{44}$$

(c) Substitution of removed j from all non-native pools on pool R_{ij} and putting resulting i into a "holding tank" H_i:

$$\tilde{\tilde{j}}(R_{ij}) = \tilde{j}(R_{ij}) + TotalRem_j, \tag{45}$$

$$\tilde{\tilde{i}}(R_{ij}) = \frac{\tilde{i}(R_{ij}) \times \tilde{j}(R_{ij})}{\tilde{\tilde{j}}(R_{ij})}, \tag{46}$$

$$H_i = \tilde{\tilde{i}}(R_{ij}) - \tilde{i}(R_{ij}). \tag{47}$$

(d) Return of proportional share of i from holding tank H_i to every R_{kl} where $k \neq i$ and $l \neq i$, and proportional share of j from holding tank H_j to every R_{kl} where $k \neq j$ and $l \neq j$

$$\tilde{\tilde{i}}(R_{kl}) = i(R_{kl}) + H_i \times RemShare_j(R_{kl}), \tag{48}$$

$$\tilde{\tilde{j}}(R_{kl}) = j(R_{kl}) + H_j \times RemShare_i(R_{kl}). \tag{49}$$

This step is performed to prevent a situation in which a malicious LP attempts to provide liquidity into a pool that contains a mispriced token (e.g. a worthless token) and drain liquidity from other pools by trading via virtual pools. The requirement to substitute the pool's (potentially mispriced) token from the reserves of other pools prior to withdrawing liquidity from the pool eliminates the malicious LP's gains and removes incentives for setting up liquidity pools with mispriced assets.

2. Return of reserves to native pools as in subsection 3.6.

3. Proportional withdrawal from R_{ij}:

$$\tilde{i}(R_{ij}) = i(R_{ij}) \left(1 - \frac{\omega t_{X,ij}}{\sum_I t_{I,ij}} \times (1 + \frac{1}{2}\rho_{ij})\right), \tag{50}$$

$$\tilde{j}(R_{ij}) = j(R_{ij}) \left(1 - \frac{\omega t_{X,ij}}{\sum_I t_{I,ij}} \times (1 + \frac{1}{2}\rho_{ij})\right). \tag{51}$$

Such proportional withdrawal compensates the withdrawing LP for the reserves in R_{ij} that remain in the pool and are not withdrawn.

4. Updating the number of tokens of LP X in pool R_{ij}:

$$\tilde{t}_{X,ij} = t_{X,ij}\,(1 - \omega)\,. \tag{52}$$

5. Updating of reserve ratios and virtual and total pools, as well as proportional fees, as in subsections 3.2 and 3.3.

4 Gains from trading on KirtuSwap: A simulation

In this section we perform a simulation of performance of KirtuSwap relative to a traditional AMM. To aid intuition, we examine an economy with 4 currencies.
We chose currencies that have sizeable real-world TVL and volume of trading on Uniswap: ETH, USTD, WBTC, and dYdX. The analysis is based on estimating evolution of prices, liquidity trades, and arbitrage trades over a period of one month, and
averaging across a large number of possible sample paths ("trials").

4.1 Setup and procedure

The simulation proceeds via the following steps:

1. Collecting real-world data on TVL and monthly estimated trading volume on Uniswap, which we use as a benchmark for the analysis.

   (a) TVL for each currency pair is estimated as of October 16, 2021.

   (b) Monthly trading volume in the ETH-USDT, ETH-WBTC, and USDT-WBTC are true trading volumes over the preceding month.

   (c) Given that USDT-dYdX and WBTC-dYdX pools are small, it is likely that a sizeable portion of ETH-dYdX trades are parts of indirectly routed USDT-dYdX and WBTC-dYdX trades. To estimate the proportions of ETH-dYdX, USDT-dYdX, and WBTC-dYdX trading, we obtain statistics on overall trading volume of ETH pairs, USDT pairs, and WBTC pairs. Trading in ETH pairs correspond to 63%, in USDT pairs to 30%, and in WBTC pairs to 7%, respectively, during the month preceding October 16, 2021. Thus, out of the overall combined monthly trading volume of $850M on ETH-dYdX, USDT-dYdX, and WBTC-dYdX pairs, we assume that 63% ($530M) is on ETH-dYdX pair, 30% ($250M) is on USDT-dYdX pair, and remaining $70M is on WBTC-dYdX pair.

   Table 1 describes the values of TVL and monthly trading volume for the six currency pairs.

2. Defining the evolution of prices of the four currencies.

   (a) We assume a one-factor structure, in which each currency is correlated with the BTC factor and, in addition, has an idiosyncratic component.

(b) We estimate beta for each of the four currencies with respect to the BTC factor using daily data for 180 days prior to October 16, 2021.

(c) We estimate total return volatility of each currency using the same time period and compute idiosyncratic volatility by subtracting the systematic volatility component from total volatility estimate.
Table 2 describes the estimates of betas and total monthly return volatilities.

(d) We assume that the drifts of the currency prices (i.e. mean returns) equal zero; this assumption is immaterial.

(e) We divide the month into $30 \times 24 \times 60$ one-minute intervals and translate total and idiosyncratic volatilities into one-minute frequency.

(f) We draw random values for BTC returns and for the idiosyncratic components of other currencies' returns, and compute prices of every currency every minute.

3. Defining the distribution of liquidity trades in each currency pair.

   (a) We compute mean and standard deviation of trade sizes on Uniswap in the 6 months prior to October 16, 2021. These values are roughly \$20K and \$50K respectively.

   (b) We compute mean and volatility parameters of a log-normal distribution that correspond to the mean and standard deviation above.

   (c) We estimate the probability of a liquidity trade during a one-minute interval in a given currency pair by dividing expected one-minute-level trading volume by mean trade size.

   (d) Every minute for every currency pair we draw a value of a liquidity trade (including its direction). The value is positive (and drawn from the log-normal distribution above) with the liquidity trade probability estimated above and equals zero with the remaining probability.

4. Defining parameter values used in the KirtuSwap algorithm. These parameter values are summarized in Table 3.

5. Every minute consists of the following steps:

   (a) Liquidity trades on all possible currency pairs (12 in total, i.e. 6 pairs in two possible directions) and all the steps following each liquidity trade,
   as described in the KirtuSwap algorithm
   in Section 3 (comparing trading costs using total pools versus costs of indirect trading on real pools; updating real pools; updating non-native pools that contribute to the virtual pools; updating reserves of real pools; updating reserve ratios and below-reserve-threshold indicators; updating real-to-virtual pool balances and associated indicators; updating virtual pools; updating total pools; and updating weighted average total fees.

   (b) Arbitrage trades that close all imbalances on all pools relative to external prices, and all the updates following each arbitrage trade, as in the previous item.

   (c) Exchange of reserves among pools and exchange through sale of reserves on their native pools, and all the updates following each exchange of reserves.

   (d) Price evolution from the previous minute.

6. Steps 1-5 (trials) are repeated 1,000 times.

7. Steps 1-5 are repeated 1,000 times for the same paths of currency prices and liquidity trades while substituting step 5 (a) (liquidity trades) with liquidity trades that would have been performed on a traditional AMM (using the cheapest of a direct and all possible indirect routes).

Table 1: **TVL and trading volumes**

Pool	**TVL (in $US)**	**Monthly volume (in $US)**
ETH-USDT	144M	1,600M
ETH-WBTC	285M	700M
ETH-dYdX	33M	500M
USDT-WBTC	25M	270M
USDT-dYdX	8M	280M
WBTC-dYdX	7M	60M

Table 2: **Betas and idiosyncratic volatilities**

Currency	**Beta (in $US)**	**Monthly idio. vol.**
ETH	0.5	25%
USDT	0	0%
WBTC	1	0%
dYdX	0.7	63%

Table 3: **Parameters**

Parameter	**Value**
Reserve threshold, ρ	2%
Pool balance threshold, η	1%
Real fee component, f_R	0.3%
Virtual fee component, f_V	0.6%
Traditional AMM fee	0.3%

4.2 Results

After performing 1,000 trial runs of the simulation, we obtain the following quantities:

1. Liquidity traders' cost of trading:

   - Mean proportional cost of a trade of a given currency pair when trading is done using curves of total pools according to KirtuSwap algorithm. Proportional cost of a trade is defi ned as the ratio of a combination of dollar trading fee and dollar slippage cost on one hand and dollar value of purchased tokens on the other hand;
   - Mean proportional cost of a trade of a given currency pair when trading is done using curves of real pools as in a traditional AMM;
   - Difference between mean proportional cost of a trade on KirtuSwap versus a traditional AMM;
   - Sharpe ratio of the diff erence between mean proportional cost of a trade on KirtuSwap versus a traditional AMM;

- Proportion of trial runs in which the mean proportional cost of a trade is lower on KIRTUSWAP versus a traditional AMM;
- All these quantities aggregated over all currency pairs.

2. Liquidity providers' (LPs') returns:

   - Mean return to each of the six pools' LPs when trading is done using curves of total pools according to KirtuSwap algorithm;
   - Mean return to each of the six pools' LPs when trading is done using curves of real pools as in a traditional AMM;
   - Difference between mean returns to LPs on KirtuSwap versus a traditional AMM;
   - Sharpe ratio of the difference between returns to LPs on KirtuSwap versus a traditional AMM;
   - Proportion of trials in which the return to liquidity provision on a given pool is larger when trading is done using curves of total pools according to KirtuSwap algorithm than using curves of real pools as in a traditional AMM;
   - All these quantities aggregated over all pools.

Table 4: **Comparison of proportional trading costs**

Pool	KirtuSwap	**Trad. AMM**	**Difference**	**Sharpe (diff.)**	**% Pos. diff.**
ETH-USDT	0.488%	0.476%	0.012%	1.41	9%
ETH-WBTC	0.400%	0.386%	0.014%	1.60	7%
ETH-dYdX	0.923%	1.053%	-0.130%	-5.32	100%
USDT-WBTC	0.676%	0.745%	-0.069%	-2.34	100%
USDT-dYdX	0.991%	1.317%	-0.326%	-4.83	100%
WBTC-sYdX	0.961%	1.243%	-0.282%	-5.47	100%
Overall	0.545%	0.574%	-0.029%	-2.74	100%

Table 5: **Comparison of LPs' returns**

Pool	KirtuSwap	**Trad. AMM**	**Difference**	**Sharpe (diff.)**	**% Pos. diff.**
ETH-USDT	2.395%	2.402%	-0.007%	-0.13	46%
ETH-WBTC	1.396%	1.408%	-0.012%	-0.10	48%
ETH-dYdX	6.716%	6.841%	-0.125%	-0.32	43%
USDT-WBTC	1.969%	1.687%	0.282%	1.35	92%
USDT-dYdX	7.543%	-1.112%	8.655%	2.71	100%
WBTC-sYdX	7.075%	0.117%	6.958%	2.51	100%
Overall	2.238%	2.028%	0.210%	2.77	100%

Table 4, focusing on comparisons of trading costs on KirtuSwap versus a traditional AMM, demonstrates that the mean proportional cost of trading in the two currency pairs with pools associated with the largest TVL is slightly higher on KirtuSwap than on a traditional AMM. However, the differences are

small (1 basis point per month) and the Sharpe ratios of these differences are not large as well. On the other hand, there are significant improvements in the proportional trading costs of the other four pairs associated with pools with smaller TVL: the mean cost of trading (including fees and price slippage) is reduced by 7–33 basis points. These reductions correspond to 9%–25% reduction in the costs of trading versus a traditional AMM. The Sharpe ratios of these reductions in mean proportional trading costs are also substantial, ranging from 2.3 to 5.5. Another important result is that the mean trading cost in the four currency pairs with smaller TVL is reduced in VirsuSwap relative to a traditional AMM in 100% of trial runs.

A potential concern is that a small proportional increase in trading costs in currency pairs with large TVL (and large trading volume) may translate into large dollar losses to traders that could outweigh the dollar cost savings in smaller pairs. To alleviate this concern, we compute the differences between trading costs on KirtuSwap and a traditional AMM for the whole sample of all liquidity trades across all six pairs. The results are reported in the last row of Table 4. Overall, trading on KirtuSwap achieves a reduction of 3 basis points or 5% in the mean cost of liquidity trades relative to a traditional AMM. This may not look very impressive, however, it is important to remember that trading using KirtuSwap financial technology technology is designed to achieve efficiency gains in currency pairs with smaller trading volume (and TVL) and is not designed to deliver gains in trading costs in currency pairs associated with large pools. As discussed above, the trading cost reductions in currency pairs with lower trading volume are substantial; the point of comparing total costs of trading across all pairs is to show that trading efficiency gains in smaller pairs are not achieved through trading efficiency deterioration in larger pairs. The Sharpe ratio of the reduction in the overall trading costs across all pairs is 2.7, and the reduction in mean trading costs across all pairs is achieved in 100% of trial runs.
Table 5, highlighting the differences in returns to LPs on KirtuSwap versus a traditional AMM, shows that trading using KirtuSwapa lgorithm has differential effect on LPs indifferent pools: LPs' returns are insignificantly affected in pools with larger TVL – the reduction of 1-13 basic points in mean return to LPs corresponds to a 0.3%-1.8% reduction on LPs' mean returns and insignificant corresponding Sharpe ratios. On the other hand, there are tremendous increases in returns to LPs in pools with smaller TVL.In the USDT-WBTC pair, the monthly return increases by 28 basis points (or 17%). In the USDT-dYdX pair, the mean return to LPs increses from a negative 1.1% to a staggering 7.5% per month. Similarly, in the WBTC-dYdX pair, the mean monthly return to LPs increases from 0.1% to 7%. The Sharpe ratios of these increases in LPs' returns are substantial, especially in the cases of the latter two pairs. In addition, in the latter two pairs, the improvement in lLPs' returns occurs in 100% of trial runs.
The last row in Table 5 computes differences in mean LPs' returns across all pools. Overall, there is an increase of 21 basis points in LPs' average return on KirtuSwap relative to a traditional AMM,corresponding to 10% of the mean return to LPs. The Sharpe ratio of this increase is a substantial 2.8.Importantly, the increase in mean LPs' return across all trading pairs is achieved in 100% of trial runs.
The fact that KirtuSwap achieves both a reduction in trading costs and an increase in LPs' returns may seem puzzling. The key to understanding it is the fact that arbitrageurs' profi ts tend to be smaller on KirtuSwap than on a traditional AMM. This reduction in arbitrageurs' profits allows to keep a larger part of (smaller) costs paid by traders in LPs' hands.
The mechanism for achieving this reduction on arbitrageurs' profi ts is through decoupling of liquidity trades from arbitrage trades. The differential trading fees in real and virtual components of total pools on KirtuSwap create incentives for arbitrage urs to tilt their trades to pools with larger real components, whereas liquidity traders do not have a choice of which pool to trade on (within a particular trade), thus a significant fraction of liquidity trades occurs on pools with large virtual components.

5 Data-driven choice of parameters

The parameter values (including real and virtual fee components) used in the simulation in the previous section are arbitrary. Arbitrary choices of parameters are prevalent in existing AMMs, which typically set a trading fee at a certain level or allow a choice of trading fees from a limited menu of options. In this section, we briefly illustrate potential advantages of data-driven choices of parameter values in an AMM. We perform the following exercise.

1. Instead of using fixed parameter values as in Table 3, we draw parameter values of from a uniform distribution with bounds of 0.5 to 1.5 multiples of original parameter values. We randomize the following parameters:
   (a) Reserve threshold, $\rho_{ij} = \rho$;
   (b) real-to-virtual imbalance threshold, $\eta_{ij} = \eta$;
   (c) Real fee component, $f_{R_{ij}} = f_R$;
   (d) Virtual fee component, $f_{V_{ij}} = f_V$.
2. For each realization of the set of parameters, we perform 20 trial runs.
3. We then re-draw the parameters and repeat this procedure 200 times and compute all the statistics reported in Tables 4 and 5.
4. We then choose the set of parameter values that achieves one of the following two objectives:
   (a) Highest increase in mean return to LPs, conditional on not increasing mean proportional trading costs relative to a traditional AMM;
   (b) Highest reduction in mean proportional trading costs, conditional on not reducing mean return to LPs relative to a traditional AMM.

The results are reported in Table 6. The first column of Table 6 shows the results of minimizing mean proportional trading costs while not hurting LPs' returns, whereas the second column of Table 6 illustrates the results of maximizing mean LPs' returns while not hurting the traders.

Table 6: **Optimization of KirtuSwap parameters**

Optimization	**Prop. trading cost**	**LP return**
KirtuSwap baseline LP return gain	0.21%	0.21%
KirtuSwap optimized LP return gain	0.02	
% 0.45%KirtuSwap baseline prop. trading cost gain		-0.03
% -0.03%KirtuSwap optimized prop.		
Optimal reserve threshold, ρ		
trading cost gain -0.06% -0.01%	1.12%	1.63%
Optimal balance threshold, η	0.75%	1.20%
Optimal real fee component, f_R	0.25%	0.26%
Optimal virtual fee component, f_V	0.66%	0.84%

Table 6 demonstrates the following findings.

1. It is possible to achieve a reduction of 6 basis points (or over 10%) in mean proportional trading costs (most of the reduction concentrating in currency pairs associated with pools with smaller TVL), relative to a reduction of 3 basis points in the baseline estimation in Table 4, without hurting returns of LPs.

2. It is possible to achieve a 45 basis point (or 22%) increase in LPs' returns (most of the increase, again, concentrating in pools with smaller TVL) without increasing mean proportional costs of trading.

3. In both optimization problems, the optimal virtual component of trading fees is substantially larger than the real fee component. As explained in Section 4, large disparity between virtual and real fee components leads to reduction of arbitrageurs' profits relative to liquidity traders' costs of trading, and to more efficient outcomes to both traders and LPs.

The results in this section suggest that the next step in maximizing the performance of an AMM is optimization of trading parameters. This is not an easy task, as the problem is highly multi-dimensional. (Note, for example, that in this section the trading parameter including fee components are identical for all pools, which is unlikely to be optimal.) However, the potential gains from solving this problem are large, and future versions of KirtuSwap will incorporate real-time data-driven trading parameter optimization, which will further reduce trading costs and enhance LPs' returns.

References

Aloosh, A. and Li, J. (2021), Direct evidence of crypto wash trading.

Amiram, D., Lyandres, E., and Rabetti, D. (2021), Competition and product quality: Fake trading on crypto exchanges, Tel Aviv University working paper.

Cong, W., Li, X., Tang, K., and Yang, Y. (2021), Crypto wash trading, Cornell University working paper.

Lehar, A. and Parlour, C. (2021), Decentralized exchanges, University of California Berkeley working paper.

Wang, Y., Chen, Y., Deng, S., and Wattenhofer, R. (2021), Cyclic arbitrage in Decentralized Exchange Markets, ETH Zurich working paper.