Kaiko
smart data

Liquidity Distribution on Uniswap V3

Methodology

Dec. 2022 Version no.1 www.kaiko.com

Liquidity Distribution on Uniswap V3

Romain DURAND SARADJIAN*, Anastasia MELACHRINOS†

December 8, 2022

Abstract

On May 4th 2021, the Uniswap Foundation released the third version of Uniswap's source code: Uniswap V3, designed to progressively replace the second version of the protocol. Since its release, it has become the top decentralized exchange (DEX) by volume. Version 3's primary innovation is concentrated liquidity: liquidity bounded within a price range. despite the capital efficiency it brings for liquidity providers, Uniswap V3 adds an additional layer of complexity when it comes to modeling liquidity and tokens reserves available across all the possible price ranges of Uniswap v3 liquidity pools. This methodology article explains step by step how to reconstruct liquidity and tokens reserves for any pool of Uniswap v3, based on Uniswap V3 underlying math, smart contract methods, and on-chain events data.

*Kaiko - 2 rue de Choiseul 75002 Paris, France. Email: ROMAIN.DURAND@KAIKO.COM
†Kaiko - 2 rue de Choiseul 75002 Paris, France. Email: ANASTASIA@KAIKO.COM

Contents

1 Introduction

Uniswap V3 liquidity pools are fed by liquidity providers. They provide liquidity on specific price ranges according to the concentrated liquidity principle, a key feature of the Uniswap V3 algorithm. When aggregated, liquidity provider positions enable us to know how much liquidity is available in each of all initiated price ranges. Those data are extremely valuable. For example, they are necessary if one wants to evaluate market depth on Uniswap V3 markets as well as slippage on all initiated price levels of the Uniswap V3 liquidity pools. However, there is a major complexity layer since this data is not directly available through Uniswap V3's smart contracts at each price level, and thus has to be deducted.

This article first discusses the concept of concentrated liquidity in Uniswap V3, which is derived from the Uniswap V3 Core white paper [1]. By formalizing liquidity pools, we can estimate the amount of liquidity available and the amount of each token locked in each block and price range for any Uniswap V3 liquidity pool.

2 Uniswap V3 Fundamentals

2.1 Maximizing capital efficiency for liquidity providers

The third version of Uniswap introduces the concept of concentrated liquidity, which solves the main problem of Uniswap's previous version: the lack of incentives for liquidity providers due to low capital efficiency [1]. On Uniswap V2, liquidity providers provide an equivalent amount of both tokens on the overall price curve of a liquidity pool. In return, they receive a share of the transaction fees (0.3% of each swap amount on Uniswap V2) generated by traders using the liquidity to make swaps. This share is proportional to the liquidity provider's contribution relative to the other liquidity providers in the pool. Uniswap V3 allows liquidity providers to optimize their capital efficiency and fee-based rewards by letting them choose which price levels they want to provide liquidity at. The smaller the price range a liquidity provider operates in, the higher the likelihood that they will capture a significant share of liquidity and fees over the covered price ranges.

2.2 Constant Product Market Maker

Liquidity reserves in liquidity pools are crucial for understanding decentralized cryptocurrency markets and price formation, as prices on Uniswap rely solely on the relative amount of tokens in liquidity pools.

$$p = \frac{y}{x}$$

with p = the price of token token X in terms of token Y

x = the amount of token X in a liquidity pool

y = the amount of token Y in a liquidity pool

Uniswap V2 uses an automated market maker (AMM) model, which has been essential to the growth and adoption of decentralized finance. AMM-based decentralized exchanges (DEXs) like Uniswap represent the vast majority of volume on DEXs. They enable the automation of cryptocurrency exchanges and the algorithmic determination of asset prices, eliminating the need for intermediaries and reducing execution costs compared to order-book based DEXs [3]. However, there are different flavors of AMMs [4] such as Uniswap's constant product market makers (CPMM), Curve's Stableswap Invariant, which is a combination of constant product and constant sum invariants, or Balancer's constant mean market maker. This paper focuses on the CPMM AMM type, which is the foundation of all versions of Uniswap (V1, V2, and V3). In practice, prices on all Uniswap pools are determined using the constant product invariant rule, where the product of the reserves of each token in the pool is constant, $xy = k$. The main difference between Uniswap V2 and Uniswap V3 is that the reserves used in the invariant function in Uniswap V2 are actual reserves, while in Uniswap V3 they are virtual reserves.

x represents the amount of token X and y represents the amount of token Y locked in a liquidity pool at a given time. k is the invariant, a constant that determines the relationship between the two tokens in the liquidity pool. Only liquidity providers can change k. Adding liquidity (x and y) to a liquidity pool (essentially creating/minting liquidity pool tokens) will increase k, and the opposite is true for a liquidity withdrawal (essentially burning liquidity pool tokens). However, it's worth noting that k is not constant in practice. In fact, only for Uniswap V2 pools, fees paid by traders to liquidity providers are added to the pool's reserves. This means that each swap increases x or y and thus k.

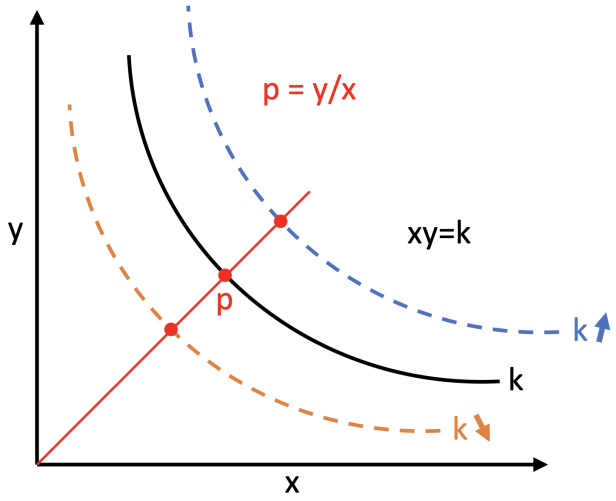

Figure 1: Representation of how prices and token quantities evolve on Uniswap V2

The constant product rule is shown in Figure 1. The black hyperbola represents the possible quantities of tokens X and Y that can be in the liquidity pool. Traders who swap these tokens are constrained to make the quantities move along this hyperbola. The price for some given quantities (x, y) is the slope of the line connecting $(0, 0)$ to (x, y).

2.3 Concentrated Liquidity Formalized

By introducing concentrated liquidity, Uniswap V3 liquidity (previously defined as the total amount of reserves locked and available within a liquidity pool) cannot be defined in the same way as on Uniswap V2. Liquidity on Uniswap V3 is no longer distributed uniformly along the reserves curve represented by the constant product ($x * y = k$ that we can rewrite $x * y = L^2$). Instead, it is represented by the combined positions of all liquidity providers for a given liquidity pool. As a result, liquidity can only be measured within price ranges. To clarify, an LP providing liquidity to a Uniswap V3 pool can specify a lower bound price p_a and an upper bound price p_b, so that their liquidity can only be used on swaps within the price range $[p_a, p_b]$. Therefore, the constant product formula can be formalized as follows:

$$L^2 = (x + \frac{L}{\sqrt{p_b}})(y + L\sqrt{p_a})$$

The relationship between this key equation of Uniswap V3 and the constant product equation of Uniswap V2 becomes even clearer when we consider the situation where $p_a, p_b = 0, \infty$. In this case, $\frac{L}{\sqrt{p_b}}$ tends to a very small number and so when p_b tends to ∞, we can consider that $\frac{L}{\sqrt{p_b}} = 0$. On the other hand, with $p_a = 0$, $L\sqrt{p_a} = 0$. This results in the same curve as the trading curve on Uniswap V2, represented by $L^2 = xy$. So, when traders want to swap token X for token Y or vice versa, the amount of tokens X and Y in the pool varies according to the Constant Market Maker Formula.

This paper aims to outline the methodology for calculating the amount of x and y locked within each initiated price range of any Uniswap V3 liquidity pool. To do this, we will first review how liquidity is distributed across price ranges on liquidity pools, and then derive the equation that determines the price mechanism on Uniswap V3.

2.3.1 Ticks

Uniswap V3 liquidity pools are divided into price ranges, expressed in ticks. Each liquidity provider provides liquidity on custom price ranges, but the boundaries of these ranges are part of a specific list of possible price values for each liquidity pool. All prices can be expressed in ticks, which are practically integers, $\tau \in \mathbb{Z}$, where $p(\tau) = 1.0001^\tau$. The tick associated with a given square price p is given by $\tau = log_{\sqrt{1.0001}}\sqrt{p(\tau)}$. Liquidity is then available across tick ranges. The size of these ranges, also called the tick spacing, is fixed in advance when the pool is created. The tick spacing

depends on the transaction fees applied to each liquidity pool and can be interpreted as a fixed percentage change in price. There are three fee tiers and corresponding tick frames that can be applied to Uniswap V3 liquidity pools.

(1) Liquidity pools with a 0.05% fee have a tick spacing of 10, which corresponds to approximately a change of 0.1% in price between two initialized ticks.

(2) Liquidity pools with a 0.3% fee have a tick spacing of 60, which corresponds to approximately a change of 0.6% in price between two initialized ticks.

(3) Liquidity pools with a 1% fee have a tick spacing of 200, which corresponds to approximately a change of 2.02% in price between two initialized ticks.

An additional fee tier of 0.01% has been added for stablecoin pairs. Liquidity pools with this fee tier have a tick spacing of 1.

3 Data

This paper explains the methodology that allows us to accurately estimate the amount of liquidity and specific amount of tokens X and Y available in each tick range of any Uniswap V3 liquidity pool. The data necessary to obtain a reliable estimation of these reserves by tick is not available through direct collection from Uniswap's V3 smart contracts and events. To obtain this output, we have built a robust multi-node infrastructure (full archive nodes) and indexer. We thus have collected all the necessary variables at an event granularity and block granularity for the pool state variables (Table 1). We have collected the data historically and update it in real-time whenever a new event (mint, burn, or swap) occurs on a liquidity pool. The advantage of doing this is that the liquidity in the pool is reflected by both liquidity events and trades since both have an impact on liquidity.[1] This frequency of collection also enables us to meet the full archive nodes capabilities in terms of number of requests.

Table 1: Uniswap V3 Collected Data and Notations

	Description	Source
type	The type of the liquidity event (mint/burn)	Pool Events
ΔL	How much liqudity was added or removed (mint/burn)	Pool Events
τ_a	The lower tick of the position	Pool Events
τ_b	The upper tick of the position	Pool Events
txhash	The transaction hash of the liquidity event (mint/burn)	Pool Events
block	The Block height	On-chain
p	The current price observed by pool and by block	Pool State
τ_c	The current tick observed by pool and by block	Pool State

[1] Liquidity events affect the amount of liquidity available at a given tick range, while swaps impact the proportion of each token available at the current tick range and the current trading price.

4 Methodology

To determine the total amount of tokens X and Y locked in each tick range of a liquidity pool on the Uniswap V3 protocol, we followed a three-step methodology for all initiated tick ranges (those on which liquidity has been added at least once by a liquidity provider) at each block from the pool's genesis.

In this paper, we define the current price range as the smallest tick or price range that contains the current price p. In addition in section 4, we use the variables x and y to represent the real reserves. These reserves are denoted as x_r and y_r in the appendix.

4.1 Identifying mints and burns in the tick ranges frame

First, by examining the liquidity events (mint and burn events) and the interval of ticks $[\tau_a; \tau_b]$ each mint or burn is applied to, we can deduce the number of unique tick ranges impacted by this event. To accurately calculate the effect of a mint on liquidity, we need to consider the tick spacing of the liquidity pool. For instance, if a mint occurs between tick 20 and tick 200 in a pool with a tick spacing of 10, this means that the liquidity ΔL has been added uniformly across 19 tick ranges.[2]

4.2 Deducing liquidity across all initiated tick ranges

To calculate the total liquidity L available over time and tick ranges based on each liquidity provider's actions (mints/burns), we sum or subtract the variation in liquidity ΔL for each liquidity event that occurs. This depends on whether the event is a mint or a burn. For each tick range that is affected, we therefore obtain the cumulative value of L.

4.3 Deducing the quantity of tokens X and/or Y across all initiated tick ranges

Having obtained the value of L for all tick ranges, we can utilize Uniswap's V3 underlying mathematics to calculate the quantity of each token in the liquidity pool for each tick range. For all $p_a \leqslant p$, $p \leqslant p_b$ and $p_a < p_b$, the following equations enable us to get the real quantity of tokens x and y in a liquidity pool and at a specific price. These equations hold for any value of p.

$$\begin{cases} x = L\frac{\sqrt{p_b}-\sqrt{p}}{\sqrt{p}\sqrt{p_b}} \\ y = L(\sqrt{p} - \sqrt{p_a}) \end{cases}$$

From the equations above, we can derive two specific cases:

[2] In order to determine the total number of ranges, we first perform the calculation 200-20=180. We then divide this result by 10 to get 18. Since there are two numbers at the endpoints (20 and 200), we add 1 to the final result to obtain a total of 19 ranges.

(1) For all tick ranges where the value of P and so the current tick, is greater than or equal to the upper tick of the current tick range, or $p \geq p_b$:

$$\begin{cases} x = L\frac{\sqrt{p_b}-\sqrt{p_a}}{\sqrt{p_a}\sqrt{p_b}} \\ y = 0 \end{cases}$$

(2) For all tick ranges where the value of p is strictly inferior to the lower tick of the current tick range, or $p < p_a$:

$$\begin{cases} x = 0 \\ y = L(\sqrt{p_b} - \sqrt{p_a}) \end{cases}$$

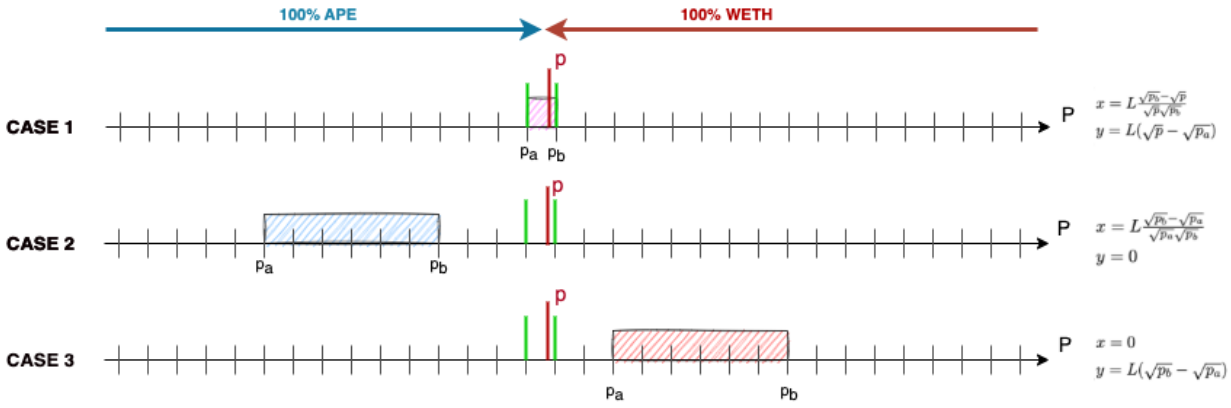

Figure 2: Deduction of Real Reserves in Pools: An Illustration

Figure 2 uses the APE-WETH pool as an example to demonstrate when each formula derived in the previous cases can be applied. The general case (1) shows that the general formulas can be used to calculate the token reserves at the current price range. Cases 2 and 3, which correspond to cases (1) and (2) above, show that specific derivations of the general case can be used to accurately measure the reserves of tokens X and Y. In this illustration, P represents any possible price within the overall price range for a given liquidity pool.

5 Data Structure

Our methodology, which includes the use of Kaiko's multi-node infrastructure and in-house blockchain indexer, allows us to create the following data structure (see Listing 1). The variables used in this structure are defined in Table 2 below and in Kaiko's official documentation..

Table 2: Uniswap V3 Liquidity Snapshots Output Fields

Field	Definition	Example
block_number	The height of the block.	16028979
timestamp	The timestamp of the block.	1669161611000
lower_tick	The lower tick of the range.	-59580
upper_tick	The upper tick of the range.	-59520
current_price	The current price at this block.	0.002857788744308444
current_tick	The current tick at this block.	-58580
amount	The amount of liquidity in the specified tick range.	1.7305248294559624e+23
amount0	The amount of token0 in the specified tick range.	0
amount1	The amount of token1 in the specified tick range.	26.438107860682372

These data can be represented as a graph showing the distribution of liquidity at different price levels, with liquidity on the Y-axis and prices on the X-axis.

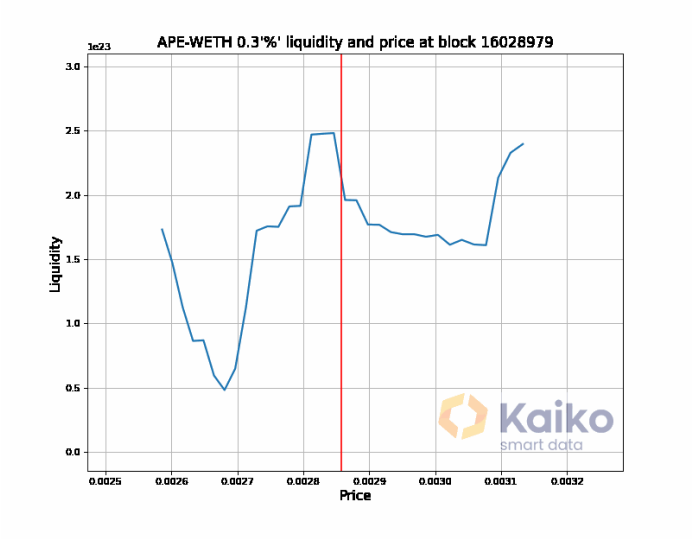

Figure 3: Distribution of liquidity over price levels for the APE-WETH 0.3% liquidity pool

A Appendix: Demonstrating Uniswap's V3 Main Equation

As mentioned in section 2.3.1, Uniswap V3 pools are divided into small price ranges (tick ranges) on which liquidity is made available by liquidity providers. The entire price curve from 0 to infinite is divided into shorter tick ranges, each of which has its own amount of liquidity. According to Uniswap's V3 whitepaper, there are two ways of expressing liquidity pools reserves: the real and the virtual reserves.

(1) Real reserves in Uniswap liquidity pools are defined as the amount of tokens x_r and y_r locked within any price range for a given liquidity pool.

(2) These real reserves are derived from the virtual reserves curve, as explained in detail by Heimbach et al.[2], *virtual reserves simulate trading on the price interval as if the liquidity distribution on the entire price range* $(0; \infty)$ *is constant and matches that of the interval* $[p_a; p_b]$. *The virtual reserves thus behave according to the constant product price curve. This way, the protocol ensures that the product of the virtual reserves x and y stays constant, i.e.,* $xy = L^2$. *Here, L is the liquidity reserved on the price interval* $[p_a; p_b]$ *and the pool's marginal price is given by* $p = \frac{y}{x}$ *[1].*

Since Uniswap V3 liquidity pools consist of different types of reserves, we can rewrite the main equation of Uniswap V3 as follows. As mentioned in section 2.2, On the price range $[p_a, p_b]$ the constant product rule $x * y = L^2$ applies, with the main difference that x and y can be decomposed as the sum of the real number of tokens in the pool and other quantities that we call offsets :

$$L^2 = (x_r + x_o)(y_r + y_o)$$

x_r are the real reserves of token X in the $[p_a; p_b]$ price range

y_r are the real reserves of token Y in the $[p_a; p_b]$ price range

x_o and $y_o$can be viewed as fake reserves of tokens X and Y in the trading curve, limited to $[p_a; p_b]$

Using the formalization of the trading curve given by Uniswap's V3 whitepaper, which states that $L = \sqrt{k} \iff L = \sqrt{xy} \iff L^2 = xy$, we can demonstrate that:

$$L^2 = (x_r + x_o)(y_r + y_o) = (x_r + \frac{L}{\sqrt{p_b}})(y_r + L\sqrt{p_a})$$

The above equation formalizes the relationship that always holds between real reserves, offset reserves, and liquidity, and does not depend on the value of the marginal price of the liquidity pool. This rule applied to a specific price range $[p_a, p_b]$ is directly derived from Uniswap's V2 constant product market maker rule $x * y = L^2$. It holds for any value on $[p_a, p_b]$, and for the following offset reserves values $x_o = \frac{L}{\sqrt{p_b}}$ and $y_0 = L\sqrt{p_a}$. For instance, when $p = p_a$, we can rewrite the rule as follows, $L^2 = (x_r^a + x_o)(y_r^a + y_o)$. In this case $y_r^a = 0$, and thus we can deduce that $p_a = \frac{y_o}{x_r^a + x_o}$. By

combining this with the constant product market maker rule, we can derive the value of y_o: Since $L^2 = \frac{y_o}{p_a} y_o$, we deduce that $y_o = L\sqrt{p_a}$. The same mechanism can be applied when $p = p_b$, with $x_o x_o p_b = L^2$, we deduce that $x_o = \frac{L}{\sqrt{p_b}}$.

References

[1] H. Adams, N. Zinsmeister, M. Salem, R. Keefer, and D. Robinson. Uniswap v3 core. *Uniswap, Tech. Rep.*, 2021. https://uniswap.org/whitepaper-v3.pdf.

[2] L. Heimbach, E. Schertenleib, and R. Wattenhofer. Exploring price accuracy on uniswap v3 in times of distress. In *Proceedings of the 2022 ACM CCS Workshop on Decentralized Finance and Security*, 2022.

[3] F. Schär. Decentralized finance: On blockchain-and smart contract-based financial markets. *FRB of St. Louis Review*, 2021.

[4] J. Xu, K. Paruch, S. Cousaert, and Y. Feng. Sok: Decentralized exchanges (dex) with automated market maker (amm) protocols. *arXiv preprint arXiv:2103.12732*, Mar. 2021.

```
{
  "data": {
    "block_number": 14737185,
    "timestamp": 1652028160,
    "pool_name": "WETH-USDC0.3",
    "pool_address": "0xbebc44",
    "exchange": "usp3",
    "current_price": 5004,
    "current_tick": 650063,
    "snapshots": [{
      "amount0": 678,
      "amount1": 0,
      "amount": 37894,
      "lower_tick": 650000,
      "upper_tick": 650060,
    }, {
      "amount0": 790,
      "amount1": 56890,
      "amount": 56890383,
      "lower_tick": 650060,
      "upper_tick": 650120,
    }, {
      "amount0": 0,
      "amount1": 3678,
      "amount": 765890,
      "lower_tick": 650120,
      "upper_tick": 650180,
    }]
  }
}
```


Listing 1: Uniswap V3 Liquidity Snapshots Output Example

CONTACT

Paris
2 rue de Choiseul
75002 Paris
France

Singapore
9 Battery Road
Singapore
049910

New York
750 Lexington Ave,
New York, NY 10022
USA

London
73 Watling Street
London
EC4M 9BJ

Kaiko
smart data

www.kaiko.com

This content is the property of Kaiko, its affiliates and licensors. Any use, reproduction or distribution is permitted only if ownership and source are expressly attributed to Kaiko. This content is for informational purposes only, does not constitute investment advice, and is not intended as an offer or solicitation for the purchase or sale of any financial instrument.

© 2022, Kaiko