Liquidity Rebalancing Derivations

Lixir Automated Liquidity Manager

May 4, 2021

1 Introduction

The current generation of Decentralised Exchanges (DEXs) experienced massive growth in 2020, especially when compared against the growth of Centralised Exchanges (CEXs). This is largely due to the innovation and implementation of Automated Market Makers (AMMs), which helped to solve the biggest obstacle of DEXs – liquidity. AMMs are entities that pool liquidity and make it available to traders according to an algorithm. The vast majority of AMMs (Uniswap, Sushi, Cakeswap) utilize Constant Function Market Makers (CFMMs).

CFMMs are based on a function that establishes a predefined set of prices based on the available quantities of two or more assets. As they are implemented today, CFMMs are often capital inefficient. In the constant product market maker formula used, only a fraction of the assets in the pool are available at a given price. This is inefficient, particularly when assets are expected to trade close to a particular price at all times, making the majority of pooled liquidity wasted. Since the advent of CFMMs, there have been minorly successful attempts to enhance liquidity, but it is arguable that the downsides introduced were greater than the gains. This is set to change with the release of the first significant innovation in liquidity for DEXs since AMMs first launched – liquidity concentration.

Liquidity concentration gives liquidity providers (LPs) the ability to concentrate their capital within custom price ranges, providing greater amounts of liquidity at desired prices. By concentrating liquidity, LPs can provide the same liquidity depth as previous liquidity pools within specified price ranges while putting far less capital at risk. Liquidity concentration enables higher capital efficiency by allowing LPs to specify ranges instead of the entire pricing curve. Through liquidity concentration, capital can effectively be multiplied thousands of times compared to previous liquidity pools.

The challenge with concentrated liquidity is keeping it "in-range" so that it earns consistent fees. Lixir is an automated liquidity manager designed to maximize "in-range" time for capital by optimizing the rebalancing of liquidity pools.

2 Repricing and Rebalancing

Inevitably, Uni V3 pools move away from the center of the price range chosen. For example, if an USDC/ETH pool has a $2350-2750 range, and the price is $2550 initially, eventually the price could easily be $2420–near the edge of the range. There are two ways that pools can "reset" at a new "center" of their range. The first is re-pricing, and the second is adding capital.

2.1 Repricing

When a pool is created, the DAO sets a re-pricing limit for it. This is a percentage that represents how close to the edge of the range the pool can be before rewards are given for re-pricing it. In our example above, let's assume that limit is 10%. The range is $400 wide, so when the price goes below $2390 or above $2710, the pool is outside the limit. To re-price the pool, USDC will have to be sold into ETH.

Note that this repricing must be done by a permissioned keeper in order to avoid flash loan attacks.

2.2 Adding Capital

The second way to reset price is by adding capital. In our example, if USDC/ETH has moved to $2700, the pool now has more USDC than ETH. To make $2700 the new "center" of the range, the pool needs ETH to be added. The contract calculates the amount of one side needed in order to reset the price to the new range or to move it closer, and rewards users in $LIX for contributing capital that moves the pool closer to this range. At any given time, some pools may be further out of range than others, and more $LIX is given for adding capital to a pool that is further out of range.

3 Liquidity Rebalancing Formulas

Uniswap V3 has invariants for adding and removing liquidity, described in terms of a change in liquidity and the change in the reserves of one or the other token. When one adds or removes liquidity, it is with regards to our position which is referenced by a lower and an upper tick bound, indicating the prices at which our liquidity is kicked in or out. If adding liquidity where the current tick is below the lower tick of our position only token0 is added, conversely if the current tick is above the upper tick only token1 is added. These cases can be handled straightforwardly for single sided add, but if our lower tick is below or equal to the current tick and below our upper tick, both tokens are required and we need a formula to do this single sided provision optimally. This is a very likely scenario, because in most cases we want to provide liquidity across a range in between which we expect the price to stay. We can find some relevant equations in the Uniswap V3 whitepaper [1].

Normally, one must add liquidity on both sides in proportion to the reserve (in V2) conditions or the price and ticks of the position being added to (in V3). In V2, there is a constant product $k = xy$ maintained, and so by solving $k = (res^X + amtIn^X - z)(y + swapIn(z))$ for z one can derive the amount of $tokenX$ to swap to Y in order to use all a given amount of $tokenX$ to enter the pool. This is the method that Zapper.fi uses for UniswapV2, and the math behind that is explained by Nipun Pitimannaaree on Alpha Finance Lab blog[2]. We apply a similar method to UniswapV3.

Firstly, UniswapV3 idiomatically orders the tokens by the order of the token addresses. We refer to $token0$ by X, $token1$ by Y. The price is in terms of y/x

The formulas depend on the lower bound of the price at the ticks which concern us.

We denote the current price c, the price at the lower bound of our lower tick l and the price at the lower bound of our upper tick u. For gas and algebraic reasons, all formulas are posed in terms of the square root of the price.

These are the equations describing the invariants for adding or removing liquidity in the case that concerns us, where the current price is greater than or equal to the lower and upper bounds of our position. They say how much of each token is to be added (conv. removed) when we are adding (conv. removing) liquidity.

$$\Delta X = \Delta L \left(\frac{1}{\sqrt{c}} - \frac{1}{\sqrt{u}} \right) \tag{1}$$

$$\Delta Y = \Delta L \left(\sqrt{c} - \sqrt{l} \right) \tag{2}$$

We can start solving our problem by rewriting these two equations.

$$\Delta L = \frac{\Delta X}{\frac{1}{\sqrt{c}} - \frac{1}{\sqrt{u}}} \tag{3}$$

$$\Delta L = \frac{\Delta Y}{\sqrt{c} - \sqrt{l}} \tag{4}$$

We can set these to be equal.

$$\frac{\Delta X}{\frac{1}{\sqrt{c}} - \frac{1}{\sqrt{u}}} = \frac{\Delta Y}{\sqrt{c} - \sqrt{l}} \tag{5}$$

Following an equivalent methodology described in the blog post by Alpha Homorra linked above, we can now formulate the equations we must solve for.

Since we are swapping one or the other token before adding liquidity, we must replace $\sqrt{c}$ by $\sqrt{p(n)}$ where $\sqrt{p(n)}$ is the price after we swap the z part of the token amount we must swap for single sided entry.

In the case of doing single sided entry of X, we can derive the amount of X to solve by:

$$\frac{x - z}{\frac{1}{\sqrt{p(n)}} - \frac{1}{\sqrt{u}}} = \frac{swapIn^X(z)}{\sqrt{p(n)} - \sqrt{l}} \tag{6}$$

And in the case of doing single sided entry of Y:

$$\frac{swapIn^Y(z)}{\frac{1}{\sqrt{p(n)}} - \frac{1}{\sqrt{u}}} = \frac{y - z}{\sqrt{p(n)} - \sqrt{l}} \tag{7}$$

In both cases we solve for z.

The formula for $swapIn^X$ is given by:

$$swapIn^X(x) = getAmtDelta^Y(sqrtPriceForAmtIn^X((1-f)x)) \tag{8}$$

and $swapIn^Y$:

$$swapIn^Y(y) = getAmtDelta^X(sqrtPriceForAmtIn^Y((1-f)y)) \tag{9}$$

where f is the amount of fees taken by the liquidity providers and the protocol.

The functions used here are:

$$sqrtPriceForAmtIn^X(x) = \frac{L\sqrt{c}}{L + \sqrt{c}x} \tag{10}$$

$$amtDelta^Y\left(\sqrt{p(i)}\right) = L(\sqrt{c} - \sqrt{p(i)}) \tag{11}$$

$$sqrtPriceForAmtIn^Y(y) = \sqrt{c} + \frac{y}{L} \tag{12}$$

$$amtDelta^X\left(\sqrt{p(i)}\right) = \frac{L}{\sqrt{c}} - \frac{L}{\sqrt{p(i)}} \tag{13}$$

Note the similarity of equation 2 and equation 11, and the similarity of equation 13 and equation 1. They use the same solidity implementation (in the core contracts - they have a simpler implementation in the periphery that are only used for testing)

Here's a list of the formulas with a corresponding solidity implementation:

- (10) getNextSqrtPriceFromAmount0RoundingUp [1] - (12) getNextSqrtPriceFromAmount1RoundingDown [2] - (2), (11) getAmount1Delta [3] - (1), (13) getAmount0Delta [4] - (3) getLiquidityForAmount0 [5] - (4) getLiquidityForAmount1 [6]

We note here that in our swap equations, the argument to $getAmtDelta$ is our $\sqrt{p(n)}$ so $\sqrt{p(n)} = sqrtPriceForAmtIn(z)$.

So our equation for X becomes:

$$\frac{x - z}{\frac{1}{sqrtPriceForAmtIn^X((1-f)z)} - \frac{1}{\sqrt{u}}} = \frac{getAmtDelta^Y(sqrtPriceForAmtIn^X((1-f)z))}{sqrtPriceForAmtIn^X((1-f)z) - \sqrt{l}}$$

[1] https://github.com/Uniswap/uniswap-v3-core/blob/v1.0.0-rc.2/contracts/libraries/SqrtPriceMath.sol#L28

[2] https://github.com/Uniswap/uniswap-v3-core/blob/v1.0.0-rc.2/contracts/libraries/SqrtPriceMath.sol#L68

[3] https://github.com/Uniswap/uniswap-v3-core/blob/v1.0.0-rc.2/contracts/libraries/SqrtPriceMath.sol#L153

[4] https://github.com/Uniswap/uniswap-v3-core/blob/v1.0.0-rc.2/contracts/libraries/SqrtPriceMath.sol#L182

[5] https://github.com/Uniswap/uniswap-v3-periphery/blob/v1.0.0-beta.23/contracts/libraries/LiquidityAmounts.sol#L23

[6] https://github.com/Uniswap/uniswap-v3-periphery/blob/v1.0.0-beta.23/contracts/libraries/LiquidityAmounts.sol#L39

And for Y:

$$\frac{getAmtDelta^X(sqrtPriceForAmtIn^Y((1-f)z))}{\frac{1}{sqrtPriceForAmtIn^Y((1-f)z}-\frac{1}{\sqrt{u}}} = \frac{y-z}{sqrtPriceForAmtIn^Y((1-f)z)-\sqrt{l}}$$

Expanding for X

$$\frac{x-z}{\frac{1}{\frac{L\sqrt{c}}{L+\sqrt{c}(1-f)z}}-\frac{1}{\sqrt{u}}} = \frac{L\left(\sqrt{c}-\frac{L\sqrt{c}}{L+\sqrt{c}(1-f)z}\right)}{\frac{L\sqrt{c}}{L+\sqrt{c}(1-f)z}-\sqrt{l}}$$

And for Y:

$$\frac{\frac{L}{\sqrt{c}}-\frac{L}{\sqrt{c}+\frac{(1-f)z}{L}}}{\frac{1}{\sqrt{c}+\frac{(1-f)z}{L}}-\frac{1}{\sqrt{u}}} = \frac{y-z}{\sqrt{c}+\frac{(1-f)z}{L}-\sqrt{l}}$$

Solving for X

$$\frac{x-z}{\frac{1}{\frac{L\sqrt{c}}{L+\sqrt{c}(1-f)z}}-\frac{1}{\sqrt{u}}} = \frac{L\left(\sqrt{c}-\frac{L\sqrt{c}}{L+\sqrt{c}(1-f)z}\right)}{\frac{L\sqrt{c}}{L+\sqrt{c}(1-f)z}-\sqrt{l}}$$

Simplify the left denominator, multiply the numerator and denominator of the right by $\frac{L+\sqrt{c}(1-f)z}{L\sqrt{c}}$

$$\frac{x-z}{\frac{L+\sqrt{c}(1-f)z}{L\sqrt{c}}-\frac{1}{\sqrt{u}}} = \frac{L\left(\frac{\left(L+\sqrt{c}(1-f)z\right)}{L}-1\right)}{1-\frac{\sqrt{l}\left(L+\sqrt{c}(1-f)z\right)}{L\sqrt{c}}}$$

Divide both sides by $L\sqrt{c}$, distribute the $\sqrt{l}$

$$\frac{x-z}{L+\sqrt{c}(1-f)z-\frac{L\sqrt{c}}{\sqrt{u}}} = \frac{\frac{L+\sqrt{c}(1-f)z}{L}-1}{\sqrt{c}-\frac{L\sqrt{l}+\sqrt{c}\sqrt{l}(1-f)z}{L}}$$

Simplify the right by multiplying by L/L

$$\frac{x-z}{L+\sqrt{c}(1-f)z-\frac{L\sqrt{c}}{\sqrt{u}}} = \frac{\sqrt{c}(1-f)z}{L\sqrt{c}-L\sqrt{l}-\sqrt{c}\sqrt{l}(1-f)z}$$

Divide both sides by $\sqrt{u}$

$$\frac{x-z}{L\sqrt{u}+\sqrt{c}\sqrt{u}(1-f)z-L\sqrt{c}}=\frac{\sqrt{c}(1-f)z}{L\sqrt{c}\sqrt{u}-L\sqrt{l}\sqrt{u}-\sqrt{c}\sqrt{u}\sqrt{l}(1-f)z}$$

Factor for simplicity.

$$\frac{x-z}{L\left(\sqrt{u}-\sqrt{c}\right)+\sqrt{c}\sqrt{u}(1-f)z}=\frac{\sqrt{c}(1-f)z}{L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)-\sqrt{c}\sqrt{u}\sqrt{l}(1-f)z}$$

Cross multiply

$$\begin{aligned}
&L\left(\sqrt{u}-\sqrt{c}\right)\sqrt{c}(1-f)z+\sqrt{c}\sqrt{c}\sqrt{u}(1-f)^2z^2\\
&=L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)x-x\sqrt{c}\sqrt{u}\sqrt{l}(1-f)z\\
&-L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)z+\sqrt{c}\sqrt{u}\sqrt{l}(1-f)z^2
\end{aligned}$$

Rerange for quadratic setup

$$\begin{aligned}
&0=\\
&\sqrt{c}\sqrt{c}\sqrt{u}\left(1-f\right)^2z^2-\sqrt{c}\sqrt{u}\sqrt{l}\left(1-f\right)z^2\\
&+L\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right)\left(1-f\right)z+x\sqrt{c}\sqrt{u}\sqrt{l}\left(1-f\right)z+L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)z\\
&-L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)x
\end{aligned}$$

Factor

$$\begin{aligned}
&0=\\
&\sqrt{c}\sqrt{u}(1-f)\left(\sqrt{c}(1-f)-\sqrt{l}\right)z^2\\
&+\left(\sqrt{c}(1-f)\left(L\left(\sqrt{u}-\sqrt{c}\right)+x\sqrt{u}\sqrt{l}\right)+L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)\right)z\\
&-L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)x
\end{aligned}$$

$$\begin{aligned}
&0=\\
&\sqrt{c}\sqrt{u}\left(1-f\right)\left(\sqrt{c}\left(1-f\right)-\sqrt{l}\right)z^2\\
&+\left(\sqrt{c}\left(1-f\right)\left(L\left(\sqrt{u}-\sqrt{c}\right)+x\sqrt{u}\sqrt{l}\right)+L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)\right)z\\
&-L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)x
\end{aligned}$$

Apply quadratic formula, solving for positive solutions only.

$$z = \frac{\left(\begin{array}{l} -\left(\sqrt{c}(1-f)\left(L\left(\sqrt{u}-\sqrt{c}\right)+x\sqrt{u}\sqrt{l}\right)+L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)\right) \\ +\sqrt{\begin{array}{l}\left(\sqrt{c}(1-f)\left(L\left(\sqrt{u}-\sqrt{c}\right)+x\sqrt{u}\sqrt{l}\right)+L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)\right)^2 \\ 4\sqrt{c}\sqrt{u}\left(1-f\right)\left(\sqrt{c}\left(1-f\right)-\sqrt{l}\right)L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)x\end{array}} \end{array} \right)}{2\sqrt{c}\sqrt{u}\left(1-f\right)\left(\sqrt{c}\left(1-f\right)-\sqrt{l}\right)}$$

Solving for Y

$$\frac{\frac{L}{\sqrt{c}} - \frac{L}{\sqrt{c}+\frac{(1-f)z}{L}}}{\frac{1}{\sqrt{c}+\frac{(1-f)z}{L}} - \frac{1}{\sqrt{u}}} = \frac{y-z}{\sqrt{c}+\frac{(1-f)z}{L} - \sqrt{l}}$$

Multiply the numerator and denominator of the left by $\sqrt{c}+\frac{(1-f)z}{L}$

$$\frac{\frac{L\left(\sqrt{c}+\frac{(1-f)z}{L}\right)}{\sqrt{c}} - L}{1 - \frac{\sqrt{c}+\frac{(1-f)z}{L}}{\sqrt{u}}} = \frac{y-z}{\sqrt{c}+\frac{(1-f)z}{L} - \sqrt{l}}$$

Multiply the left by $\sqrt{u}$, multiply both sides by $\sqrt{c}$

$$\frac{L\sqrt{u}\left(\sqrt{c}+\frac{(1-f)z}{L}\right) - L\sqrt{c}\sqrt{u}}{\sqrt{u} - \left(\sqrt{c}+\frac{(1-f)z}{L}\right)} = \frac{\sqrt{c}\,(y-z)}{\sqrt{c}+\frac{(1-f)z}{L} - \sqrt{l}}$$

Divide both sides by L

$$\frac{\sqrt{u}\left(\sqrt{c}+\frac{(1-f)z}{L}\right) - \sqrt{c}\sqrt{u}}{\sqrt{u} - \left(\sqrt{c}+\frac{(1-f)z}{L}\right)} = \frac{\sqrt{c}\,(y-z)}{L\sqrt{c}+(1-f)z - L\sqrt{l}}$$

Distribute $\sqrt{u}$ on the left side ($\sqrt{c}\sqrt{u}$ cancel out, we bring the L down)

$$\frac{\sqrt{u}(1-f)z}{L\sqrt{u} - L\sqrt{c} - (1-f)z} = \frac{\sqrt{c}\,(y-z)}{L\sqrt{c}+(1-f)z - L\sqrt{l}}$$

Factor the denominators, then divide both sides by $\sqrt{c}$

$$\frac{\sqrt{u}(1-f)z}{L\sqrt{c}\,(\sqrt{u}-\sqrt{c}) - \sqrt{c}(1-f)z} = \frac{y-z}{L\left(\sqrt{c}-\sqrt{l}\right)+(1-f)z}$$

Cross multiply

$$\begin{aligned}
&L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)(1-f)z + \sqrt{u}(1-f)^2z^2 \\
&= Ly\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right) - y\sqrt{c}(1-f)z - L\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right)z + \sqrt{c}(1-f)z^2
\end{aligned}$$

Rearrange for quadratic setup

$$\begin{aligned}
&0 = \\
&\sqrt{u}(1-f)^2z^2 - \sqrt{c}(1-f)z^2 \\
&+ L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)(1-f)z + L\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right)z + y\sqrt{c}(1-f)z \\
&- Ly\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right)
\end{aligned}$$

Factor

$$
\begin{aligned}
&0 = \\
&(1-f)\left(\sqrt{u}(1-f)-\sqrt{c}\right)z^2 \\
&+\left((1-f)\left(L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)+y\sqrt{c}\right)+L\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right)\right)z \\
&-Ly\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right)
\end{aligned}
$$

Apply quadratic formula, solving for positive solutions only.

$$
\begin{aligned}
&y = \\
&\frac{\left(\begin{array}{l}-\left((1-f)\left(L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)+y\sqrt{c}\right)+L\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right)\right) \\ +\sqrt{\begin{array}{l}\left((1-f)\left(L\sqrt{u}\left(\sqrt{c}-\sqrt{l}\right)+y\sqrt{c}\right)+L\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right)\right)^2 \\ +4(1-f)\left(\sqrt{u}(1-f)-\sqrt{c}\right)Ly\sqrt{c}\left(\sqrt{u}-\sqrt{c}\right)\end{array}}\end{array}\right)}{2(1-f)\left(\sqrt{u}(1-f)-\sqrt{c}\right)}
\end{aligned}
$$

References

[1] Hayden Adams, Noah Zinsmeister, Moody Salem, River Keefer, Dan Robinson *Uniswap V3 Core*.
`https://uniswap.org/whitepaper-v3.pdf`, 2021.

[2] Nipun Pitimanaaree *Optimal One-sided Supply to Uniswap*.
`https://blog.alphafinance.io/onesideduniswap/`, Alpha Finance Lab 2020.