Uniswap v3 LP Rebalancing - Analysis & Performance Evaluation

Harsh Khandelwal

Introduction

This document presents an analysis of Uniswap v3 liquidity provision (LP) strategies by simulating rebalancing events and comparing their performance to a simple buy-and-hold (HODL) strategy. The study involves historical data processing, liquidity range calculations, rebalancing logic, and risk-adjusted return evaluations.

Approach

A) Data Processing & Retrieval:

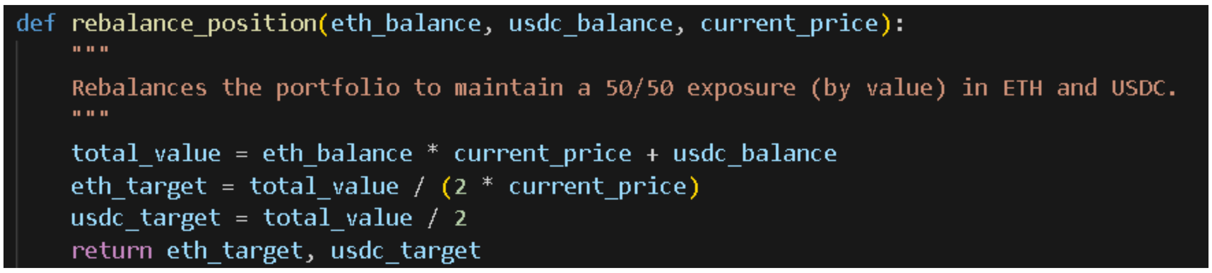
I fetched historical ETH/USD hourly prices from CoinGecko for the past year. Due to API constraints, I had to retrieve data in 90-day segments and then compile it into a continuous dataset. After collection, I filtered the data to ensure that it covered the most recent year up to the current date, aligning it with the simulation requirements. Also, checked that there is no missing values in the data.

Since the dataset is extracted in real time, the results stated in this document reflect those generated at runtime before submission. If re-executed in the future, the results may vary based on updated market data.

B) Helper Functions for Uniswap v3 LP Simulation (No Fee Accrual):

These are the Helper functions so for our strategies. Here, we have taken No fee accrual or fee rate as it will complex our maths calculation for all the data and makes our strategy complex so we have taken a more ideal condition.

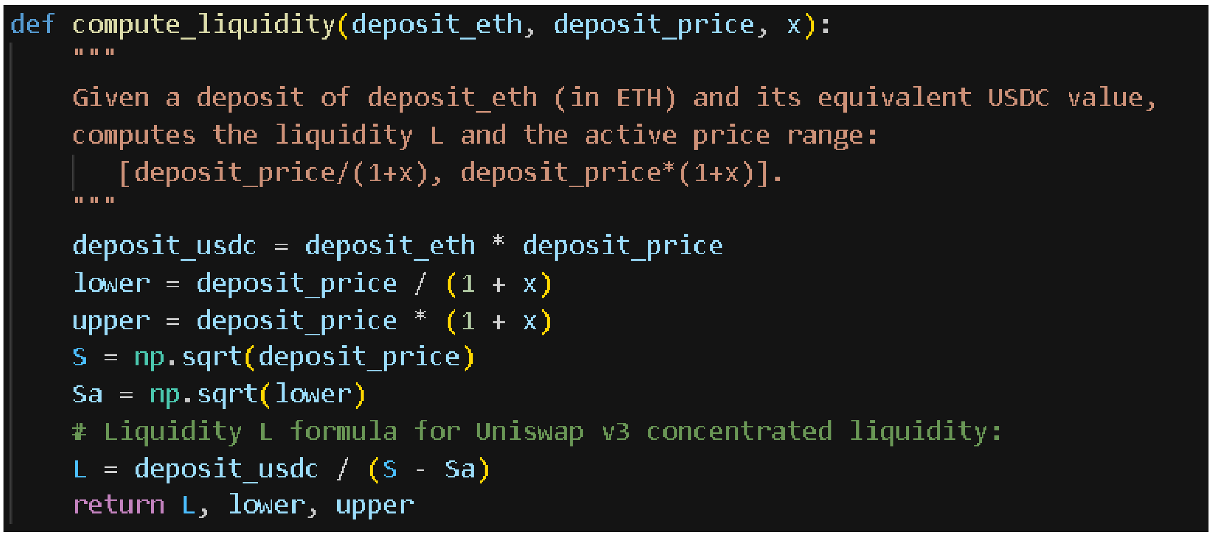
- **Function:** `rebalance_position(eth_balance, usdc_balance, current_price)`



```python
def rebalance_position(eth_balance, usdc_balance, current_price):
    """
    Rebalances the portfolio to maintain a 50/50 exposure (by value) in ETH and USDC.
    """
    total_value = eth_balance * current_price + usdc_balance
    eth_target = total_value / (2 * current_price)
    usdc_target = total_value / 2
    return eth_target, usdc_target
```

The rebalance_position function keeps the portfolio 50/50 in ETH and USDC by calculating the total value and adjusting each asset to half of it. It converts USDC to ETH based on the current price, ensuring balanced exposure and reducing risk.

- **Function:** `compute_liquidity(deposit_eth, deposit_price, x)`



```python
def compute_liquidity(deposit_eth, deposit_price, x):
    """
    Given a deposit of deposit_eth (in ETH) and its equivalent USDC value,
    computes the liquidity L and the active price range:
       [deposit_price/(1+x), deposit_price*(1+x)].
    """
    deposit_usdc = deposit_eth * deposit_price
    lower = deposit_price / (1 + x)
    upper = deposit_price * (1 + x)
    S = np.sqrt(deposit_price)
    Sa = np.sqrt(lower)
    # Liquidity L formula for Uniswap v3 concentrated liquidity:
    L = deposit_usdc / (S - Sa)
    return L, lower, upper
```

- **Function:** `compute_lp_value(L, current_price, lower, upper)`

The compute_lp_value function calculates the total LP position value and token holdings based on the current price within a defined range.

- Below Lower Bound (current_price ≤ lower) → LP holds only ETH:

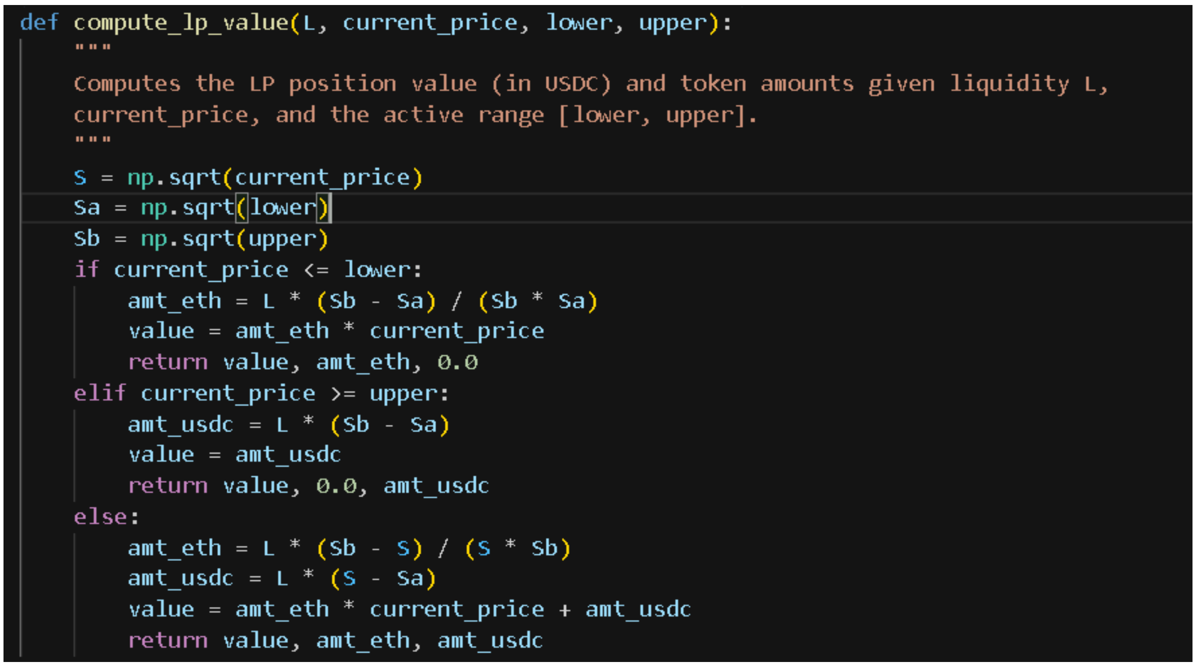
$$\text{amt\_eth} = \frac{L \cdot (S_b - S_a)}{S_b \cdot S_a}$$

Value: amt_eth * current_price

- Above Upper Bound (current_price ≥ upper) → LP holds only USDC:
  amt_usdc=L · (Sb−Sa) & Value: amt_usdc
- Within Range (lower < current_price < upper) → LP holds both ETH & USDC:

$$\text{amt\_eth} = \frac{L \cdot (S_b - S_a)}{S_b \cdot S_a}$$

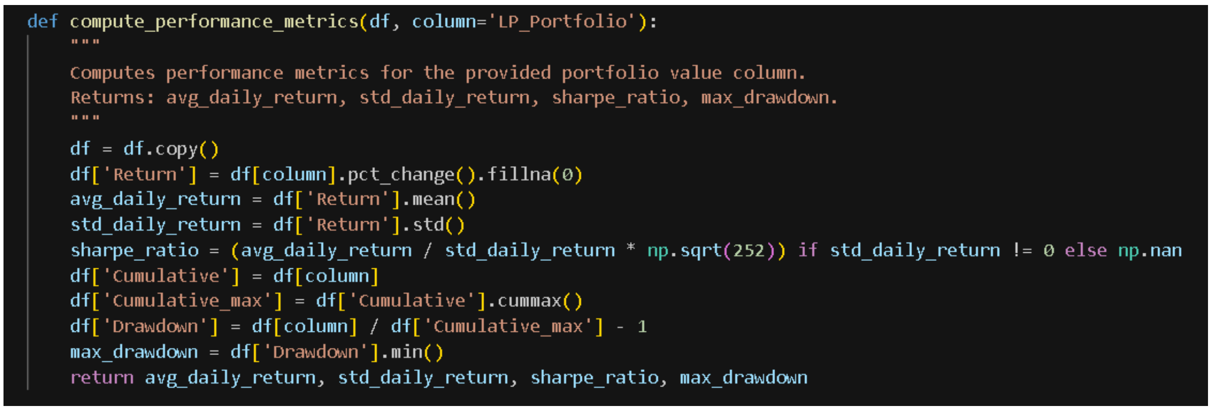
amt_usdc=L · (S−Sa)



```python
def compute_lp_value(L, current_price, lower, upper):
    """
    Computes the LP position value (in USDC) and token amounts given liquidity L,
    current_price, and the active range [lower, upper].
    """
    S = np.sqrt(current_price)
    Sa = np.sqrt(lower)
    Sb = np.sqrt(upper)
    if current_price <= lower:
        amt_eth = L * (Sb - Sa) / (Sb * Sa)
        value = amt_eth * current_price
        return value, amt_eth, 0.0
    elif current_price >= upper:
        amt_usdc = L * (Sb - Sa)
        value = amt_usdc
        return value, 0.0, amt_usdc
    else:
        amt_eth = L * (Sb - S) / (S * Sb)
        amt_usdc = L * (S - Sa)
        value = amt_eth * current_price + amt_usdc
        return value, amt_eth, amt_usdc
```

- **Function:** `compute_performance_metrices(df,column='LP_Portfolio')`

This function calculates key performance metrics for an LP portfolio, helping to evaluate risk and return.



```python
def compute_performance_metrics(df, column='LP_Portfolio'):
    """
    Computes performance metrics for the provided portfolio value column.
    Returns: avg_daily_return, std_daily_return, sharpe_ratio, max_drawdown.
    """
    df = df.copy()
    df['Return'] = df[column].pct_change().fillna(0)
    avg_daily_return = df['Return'].mean()
    std_daily_return = df['Return'].std()
    sharpe_ratio = (avg_daily_return / std_daily_return * np.sqrt(252)) if std_daily_return != 0 else np.nan
    df['Cumulative'] = df[column]
    df['Cumulative_max'] = df['Cumulative'].cummax()
    df['Drawdown'] = df[column] / df['Cumulative_max'] - 1
    max_drawdown = df['Drawdown'].min()
    return avg_daily_return, std_daily_return, sharpe_ratio, max_drawdown
```

C) Simulation Functions:

1) LP Strategy Simulation:

- Starts with **100 ETH**, splits into **50% ETH & 50% USDC**and computes **liquidity (L) and price range** using `compute_liquidity()`
- If price is within range: Compute LP portfolio value using `compute_lp_value()`.
- If price exits range: Record **impermanent loss (IL): LP_value- HODL_value** This is the basic approach of Uniswap v3 maths as there is fee accrual, gas_rate. Then Rebalance using `rebalance_position()`and Recalculate liquidity for new LP position.

- Also recorded all the iL_events date and loss in a dataframe as rebalancing_dates
- Hence returning simulation_df, iL_events and rebalancing_dates

2) **HODL Strategy Simulation:**

It is a simple buy-and-hold strategy for Ethereum (ETH) over time. Maintains an initial 50/50 ETH and USDC balance without any rebalancing.

D) Optimal x Value Search

Grid search conducted over x values ranging from **0.01 to 0.20**.

```
x_grid = np.linspace(0.01, 0.20, 50)
```

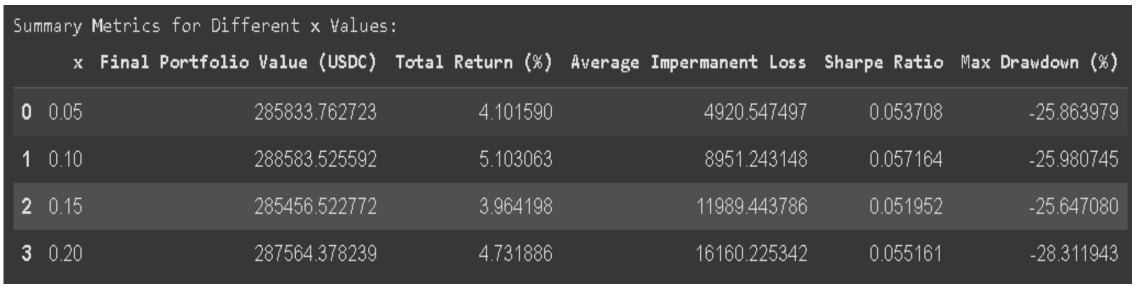
Optimal value of x is calculated based on a composite metric that averages the Sharpe ratio (a measure of risk-adjusted return) and a Calmar-like ratio (total return divided by the absolute value of maximum drawdown).

```
if abs(max_drawdown_pct) > 0:

    calmar_ratio = total_return / abs(max_drawdown_pct)

else:

    calmar_ratio = np.nan

composite = (sharpe + calmar_ratio) / 2
```

There are many more methods such as taking High sharpe ratio , taking high total returns or taking Sortino ratio. But i used this approach because it uses all the information and then helps us to find the best optimal_x keeping records on all the metrics.

Outcomes

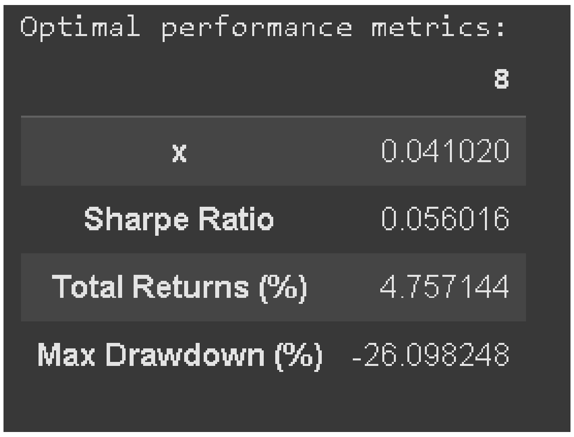
A) Performance Metrics



Summary Metrics for Different x Values:

	x	Final Portfolio Value (USDC)	Total Return (%)	Average Impermanent Loss	Sharpe Ratio	Max Drawdown (%)
0	0.05	285833.762723	4.101590	4920.547497	0.053708	-25.863979
1	0.10	288583.525592	5.103063	8951.243148	0.057164	-25.980745
2	0.15	285456.522772	3.964198	11989.443786	0.051952	-25.647080
3	0.20	287564.378239	4.731886	16160.225342	0.055161	-28.311943

B) Optimal x Value Search



Optimal performance metrics:

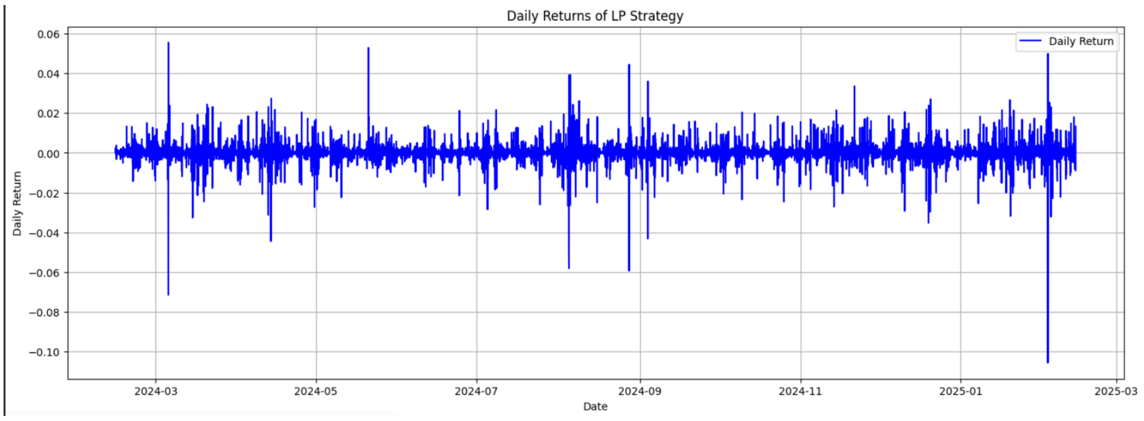
	8
x	0.041020
Sharpe Ratio	0.056016
Total Returns (%)	4.757144
Max Drawdown (%)	-26.098248

C) From HODL Strategy :

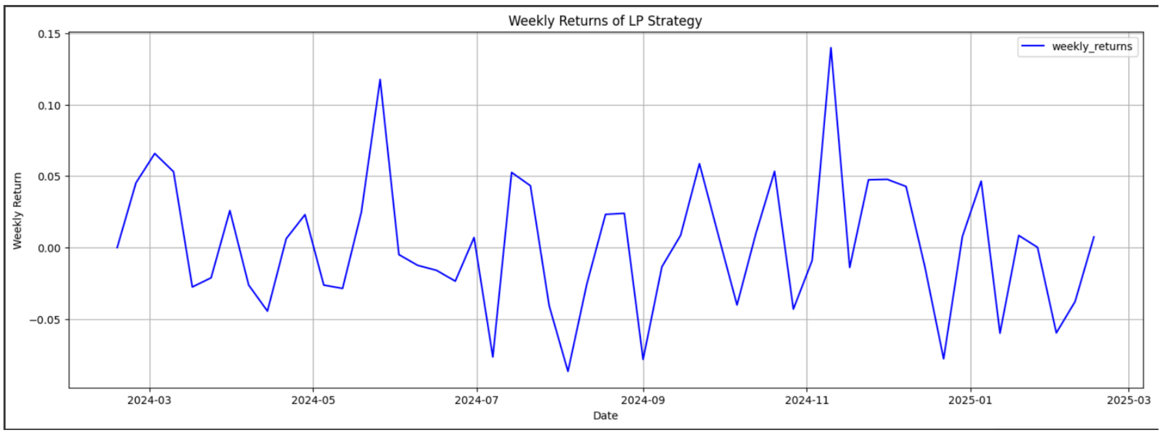
Final portfolio value: 269715.35 USDC, Total Return: -1.77%, Sharpe Ratio: 0.02, Max Drawdown: -27.55%

D) Time Based Returns : There is no such pattern in Time based returns it only depends on our strategy and the market.

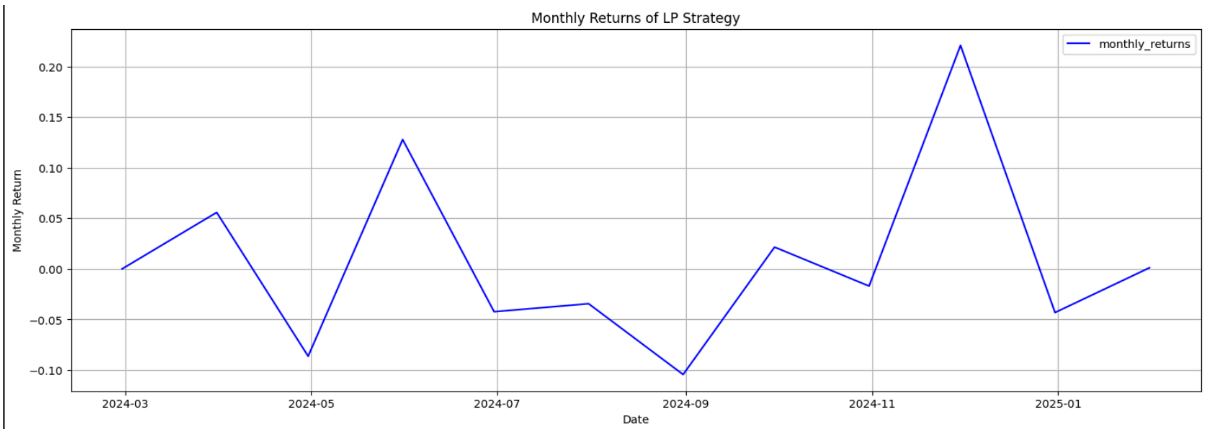
- **Daily Returns:**



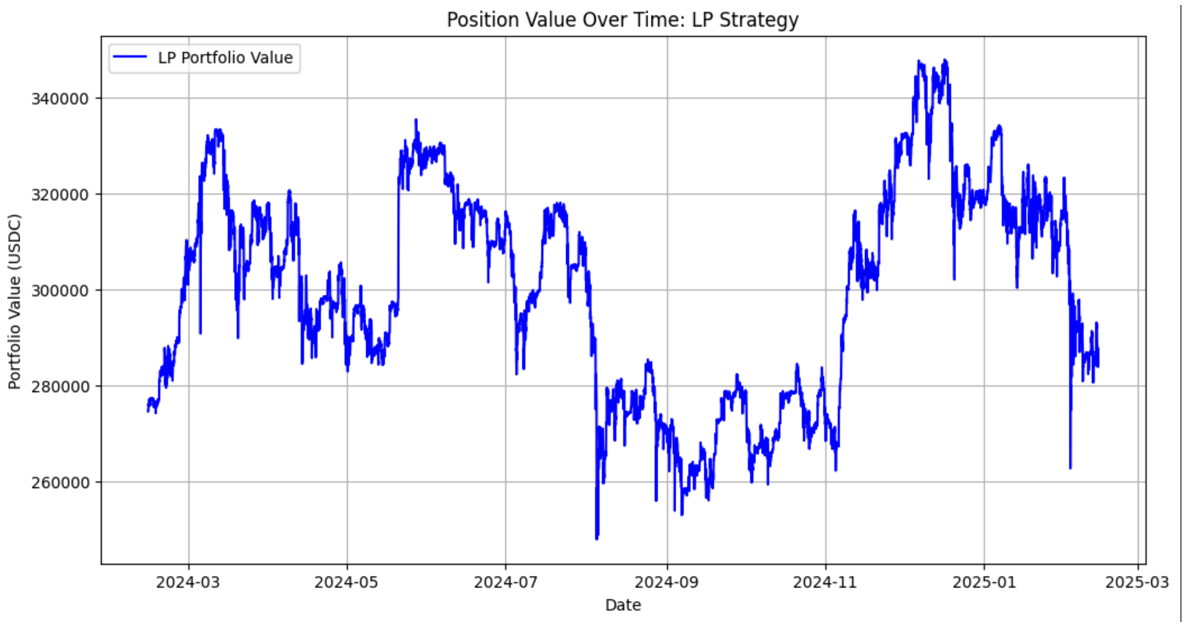
- **Weekly Returns:**



- **Monthly Returns:**



E) LP Strategy Position Value over Time:

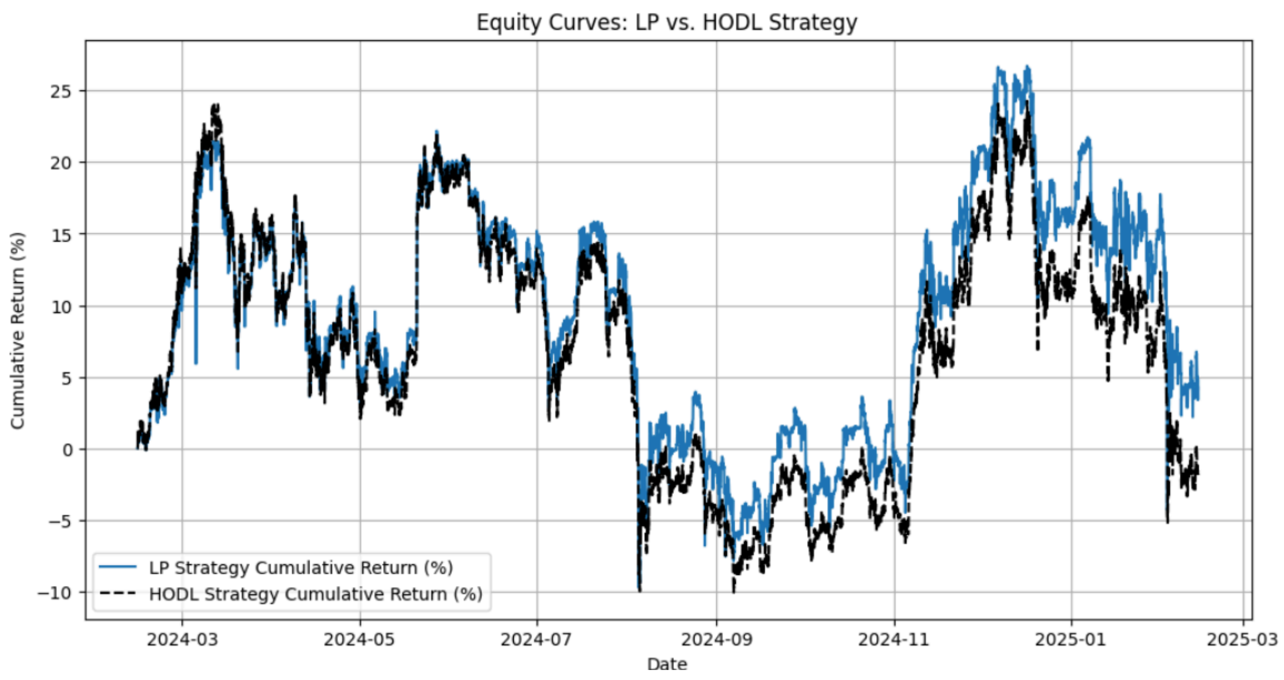


After reviewing the LP strategy Position value over time we can see that at the end of the year the position value approximately reaches out the the same position where it starts indicating that it gives us not that satisfying results of what we wanted.

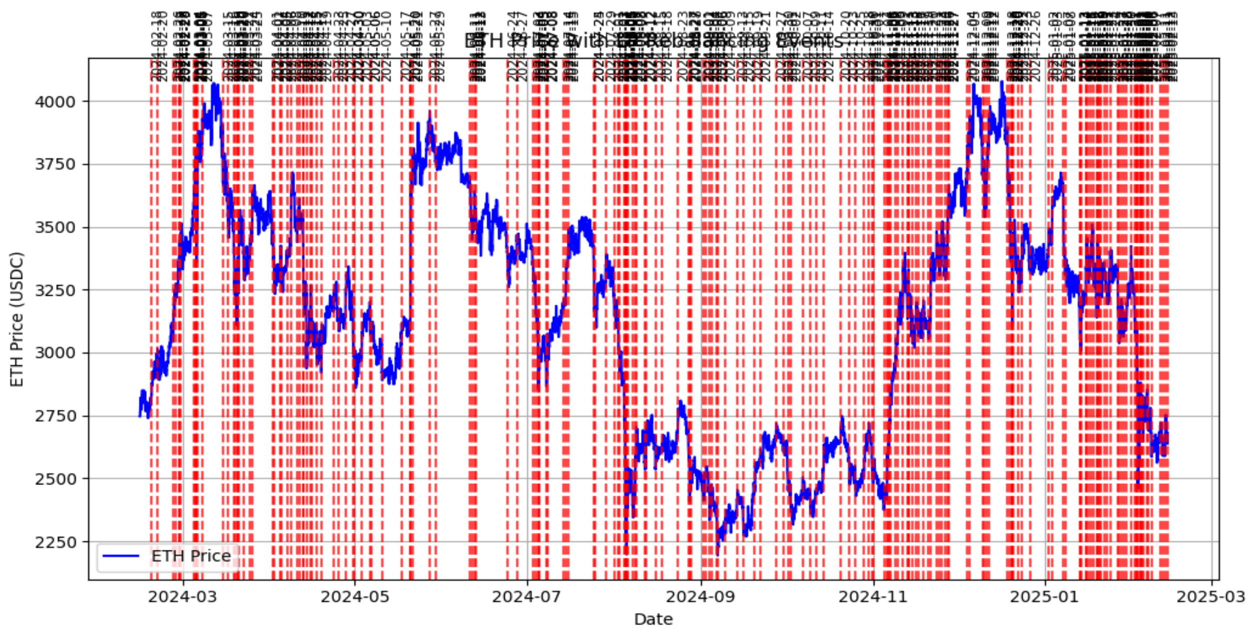
By seeing the curve we can tell that it gives us 3 - 4 maximas within a year so if we can sell our buying at that points we can make great returns.

F) Visualization Curves for optimal_x:

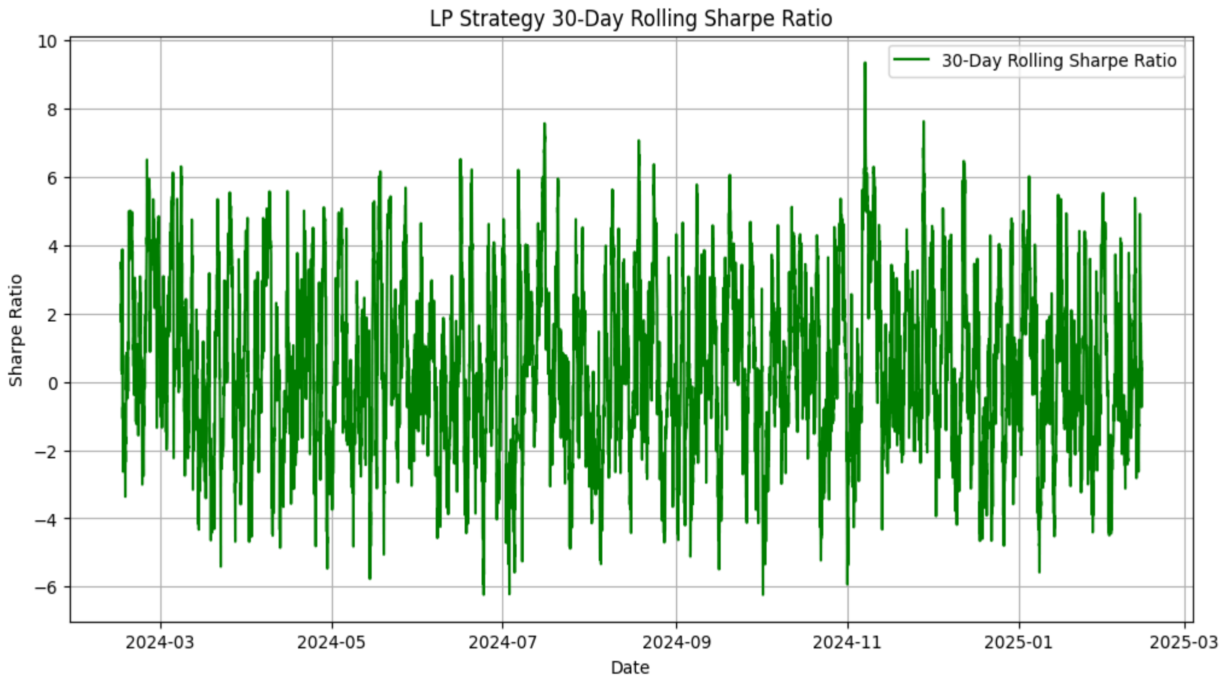
- **Equity curves:** Tells us that our Lp strategy gives better returns than HODL returns

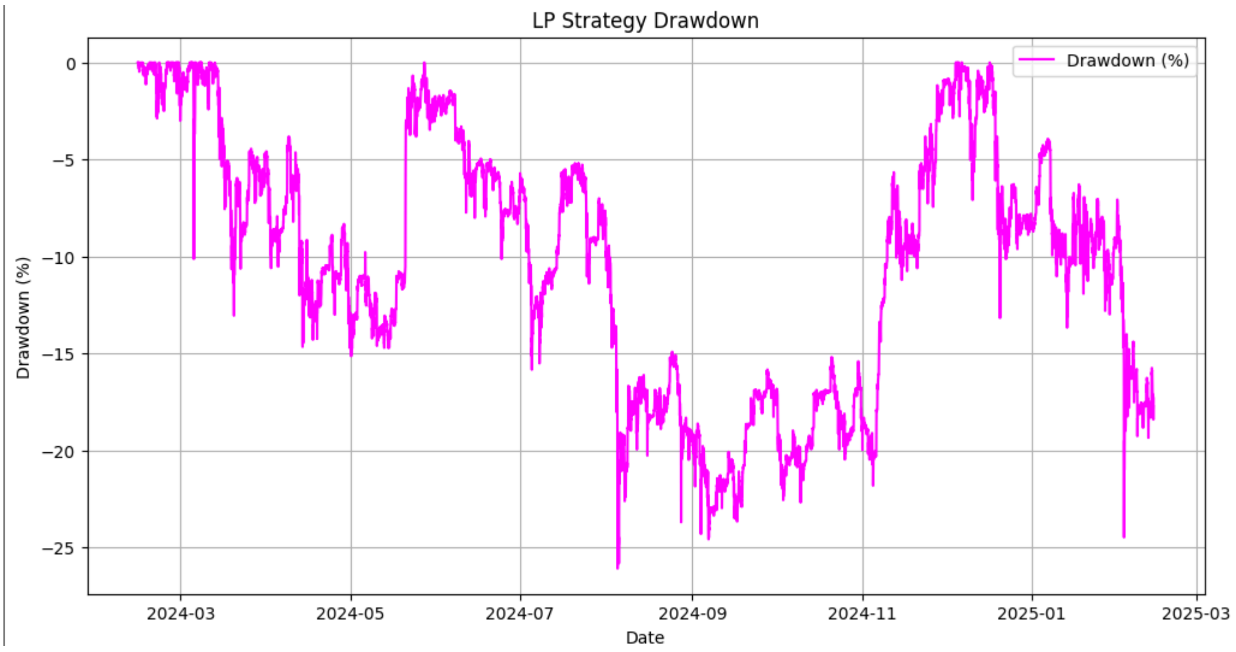


- **Price Data with Date:** Here red lines shows us the rebalancing points. For our optimal_x total 198 rebalance points exists.



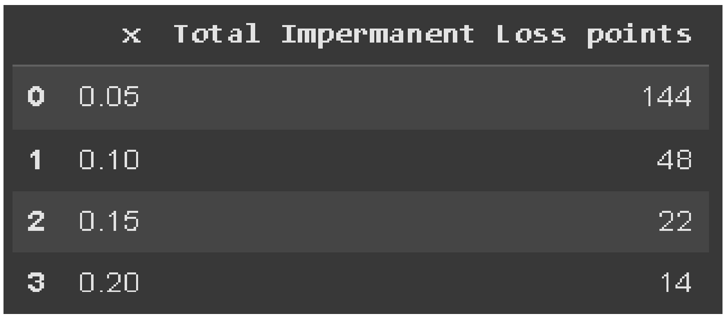
- **Risk Metrices Visualizations:**





Insights:

- As value of x increases IL points decreases as larger range leads to lesser chances of price going out of range.

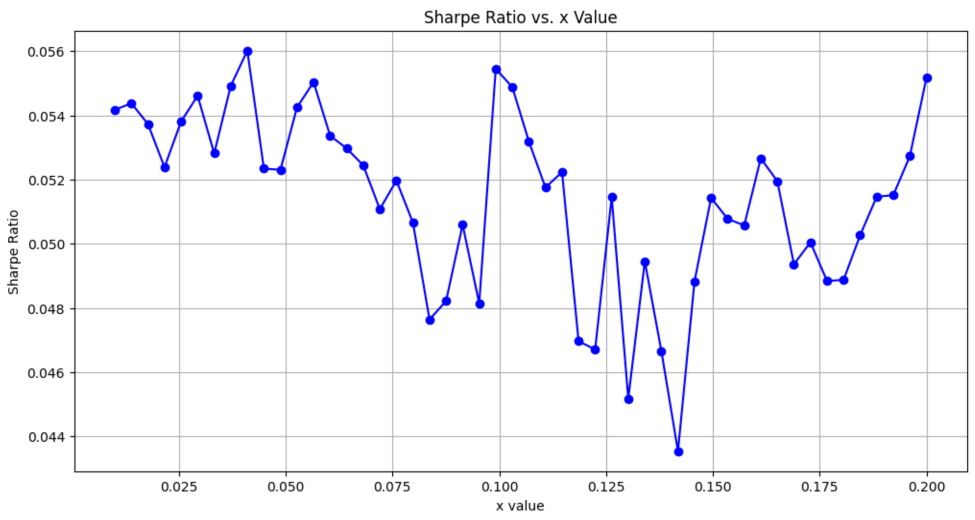


	x	Total Impermanent Loss points
0	0.05	144
1	0.10	48
2	0.15	22
3	0.20	14

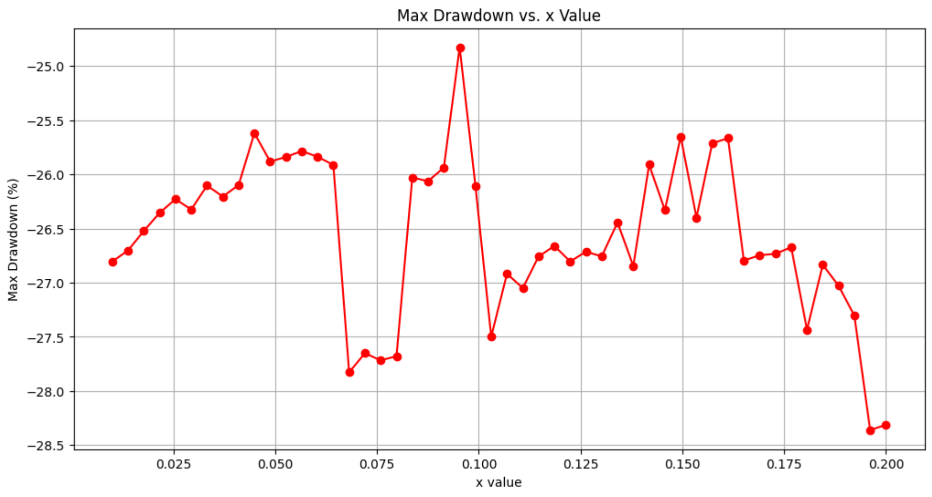
- The current simulation assumes idealized conditions (e.g., no fee accrual, no gas fees, and perfect rebalancing) which may differ from live market conditions.
- We have use Simplified V3 maths calculations at some places like for Impermanent loss because we are dealing with ideal conditions.

- <u>Sharpe ratio v/s different x value:</u>

  The trend can be seen from the graph that the sharpe ratio is approximately in range of 0.45 - 0.55 for all values of x.



- <u>Max Drawdown v/s x value</u> : Approx all values lie in the range of -25% to -28.5%


Max Drawdown vs. x Value
Max Drawdown (%)
x value