# Scaled Beta Uniswap V3 Liquidity Pools: An Analysis

Sam Gabor (sg662)

December 8, 2025

## Abstract

This research paper presents a simulation-based analysis of scaled beta-range liquidity strategies for Uniswap V3 liquidity providers (LPs). Uniswap V3 introduced concentrated liquidity, enabling LPs to allocate capital across specific price intervals (“ticks”), thereby improving capital efficiency while exposing LPs to nonlinear risks such as impermanent loss (IL) and variable fee generation.

I develop a hybrid simulation framework that combines on-chain execution (using Foundry/Anvil and Solidity contracts) with an off-chain market and trading engine implemented in Python. This framework replicates real Uniswap V3 mechanics, including liquidity math, tick transitions, and fee growth behavior. Through this environment, I evaluate whether scaled beta-range strategies—those distributing liquidity across multiple symmetric price bands—can stabilize LP returns, mitigate IL, or produce superior outcomes relative to a simple HODL benchmark.

My findings indicate that such strategies smooth exposure and reduce IL variance in mean-reverting markets but do not reliably outperform HODL unless volatility and fee volumes are sufficiently high. This aligns with existing research suggesting that passive LP strategies in Central Limit Order Book (CLOB) exchanges tend to underperform directional holding during market trends. The work contributes to a reproducible and extensible research platform for evaluating LP strategies.

## 1. Introduction

Automated Market Makers (AMMs) form a foundational component of decentralized finance (DeFi), enabling decentralized, permissionless trading without reliance on centralized order books. Uniswap V3 represents a significant shift in AMM design by introducing concentrated liquidity, allowing LPs to choose the price intervals in which their liquidity remains active. This greatly enhances capital efficiency but simultaneously increases complexity for LPs who must manage nonlinear exposure to price movement and fee generation.

This project explores whether scaled beta-range liquidity policies—multiple liquidity bands placed symmetrically around the initial price—can improve LP performance. Such configurations are intended to create smoother payoff profiles, reduce tail exposure, and potentially enhance fee capture.

To investigate this idea, I developed a hybrid simulation system that integrates smart contracts on an Anvil test chain with a Python-based order flow engine capable of driving realistic on-chain swaps. My system reconstructs LP value using exact Uniswap V3 math rather than token balances, allowing accurate measurement of IL, fees, and principal value.

The central research question is:

> *Do scaled beta-range liquidity strategies improve or stabilize Uniswap V3 LP P&L compared to HODL?*

## 2. Background and Prior Research

### 2.1 Impermanent Loss

Impermanent loss is the loss a liquidity provider experiences when the value of their deposited assets in an automated market maker (AMM) diverges from what their value would have been if they had simply held the tokens instead of providing liquidity. Through arbitrage, AMMs continually rebalance a pool's token ratios as prices move, forcing LPs to sell the outperforming asset and buy the underperforming one. This rebalancing means that whenever the relative prices of the pooled assets change, the LP's position becomes less valuable than a passive HODL position—this difference is the impermanent loss. It is called "impermanent" because it disappears if prices return to their original ratio, but it becomes permanent upon withdrawal. Trading fees earned by LPs can offset or even exceed impermanent loss, but the underlying effect always exists whenever asset prices diverge.

### 2.2 Concentrated Liquidity in Uniswap V3

Concentrated liquidity in Uniswap V3 allows liquidity providers to choose specific price ranges in which their liquidity is active, rather than spreading it uniformly across the entire price curve as in earlier AMM designs. This lets LPs deploy capital more efficiently, earning a larger share of fees with less capital when the market price trades within their chosen range. The main benefit is dramatically improved capital efficiency—LPs can achieve higher fee returns or require less capital to earn the same fees.

The primary risk, however, is that when the market price moves outside the selected range, the LP's position becomes entirely one-sided (either all of the base asset or all the quote asset), stops earning fees, and becomes exposed to potentially large impermanent loss if the price continues to drift away. As a result, concentrated liquidity requires more active management and introduces greater directional risk compared to traditional AMMs, rewarding LPs who can set effective ranges but penalizing those who misjudge volatility or price movements.

### 2.3 Multi-Range and Beta Strategies

Industry practitioners often break LP capital into multiple ranges ("tranches"). This mirrors the beta strategies tested in this project. Such approaches attempt to diversify tick exposure, reduce the risk of leaving the active range, and create smooth payoff curves.

However, studies show these strategies remain highly sensitive to volatility and fee volume.

This project explores these strategies by providing a high-fidelity simulation capable of testing outcomes under controlled conditions.

## 2.4 Additional Background: Scaled and Learned Beta Policies

Two closely related strands of research directly motivated the scaled beta-range design I implemented in this simulation framework. Both treat liquidity provision as a controlled exposure problem and view the LP as a risk-managed market maker rather than a passive liquidity donor.

First, in the paper *Market Making with Scaled Beta* [1], the authors model the market maker's inventory exposure using a beta parameter that scales sensitivity to the underlying asset's prices. In that setting, a market maker chooses a beta profile that trades off between directional risk and transaction revenue. This idea maps naturally to Uniswap V3 concentrated liquidity: each liquidity can be viewed as a local beta choice, and a set of bands implements a piecewise beta schedule across price levels.

Second, *Market Making with Learned Beta* [2] extends this line of work by allowing the beta schedule to be learned from data. Instead of fixing a static exposure profile, the market maker updates beta as it observes volatility, order imbalance, and regime shifts. This is conceptually close to how one might adapt Uniswap V3 ranges over time: a learning algorithm could thicken liquidity where volume is most likely to occur and thin it elsewhere. While my current simulator uses static beta ranges, the architecture was built with the intention that a learned policy could be layered on top in future work.

In this project, I interpret scaled beta as a design principle for how to allocate liquidity across ticks rather than as a literal control parameter. Each Uniswap V3 range represents a local slope of exposure, and by arranging multiple symmetric ranges around the initial price, I approximate the smooth beta-shaped profiles considered in academic literature. The simulation results therefore provide an applied perspective on how these ideas behave when instantiated in a real concentrated-liquidity AMM.

## 3. Methods

### 3.1 System Architecture

The simulation environment includes:

• On-chain components:
 – Uniswap V3 Factory, Pool, PositionManager
 – MockERC20 tokens for WETH and USDC
 – LPHelper: manages positions and holds LP capital
 – SwapHelper: executes swaps and arbitrage trades

• Off-chain components:
 – Python engine (run_orderflow.py)
 – Market price model using OU or lognormal processes
 – Arbitrage logic to realign pool price with market price
 – State reconstruction (state_snapshot.py)
 – CSV logging and valuation analysis (summarize_episode.py)

### 3.2 Scaled Beta-Range Liquidity Policies

Beta strategies allocate liquidity across multiple tick ranges around the current price. Example ranges might include:

$$[P0 - 150, P0 - 100], [P0 - 100, P0 - 50], \ldots, [P0 + 100, P0 + 150]$$

Each range receives a liquidity weight. This creates a symmetric exposure that behaves roughly like a smooth curve centered on spot.

### 3.3 Order flow Model and Arbitrage

At each simulation step:

1. Market price evolves via OU or lognormal model.
2. A random BUY or SELL order is generated.
3. SwapHelper executes a Uniswap V3 swap.
4. Optional arbitrage trades correct pool price back to market price.

### 3.4 Valuation Using True Uniswap V3 Math

To accurately value positions, my system reconstructs position token amounts using:

```
(amount0, amount1) = LiquidityAmounts.getAmountsForLiquidity(...)
```

Fees are computed via Uniswap feeGrowth variables, and LP value is computed using:

```
principal_value = amount0 * P_market + amount1
fee_value = fees0 * P_market + fees1
IL = principal_value - HODL_value
```

Using market price (not pool price) ensures economically consistent valuations.

## 3.5 Implementation Challenges with Uniswap V3 Math in LPHelper

A major practical challenge in this project was implementing accurate Uniswap V3 math inside the LPHelper contract. Conceptually, the goal was straightforward: for each LP range, I wanted to reconstruct principal amounts, compute uncollected fees, and report a clean decomposition of LP value into principal, fees, and impermanent loss. In practice, reproducing the Uniswap V3 internal accounting in Solidity 0.7.6 proved significantly more difficult than expected.

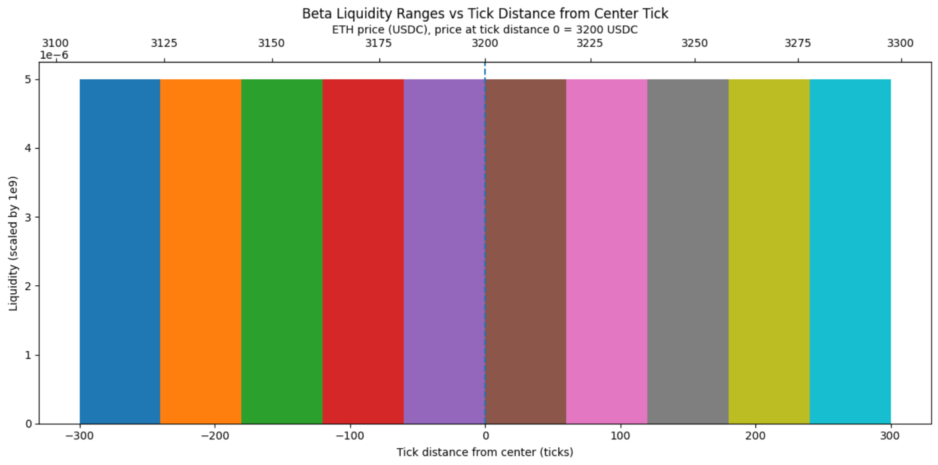
One major obstacle was Solidity's stack-depth limitation. The natural implementation of a single-position valuation routine involves multiple local variables: the current sqrt price, lower and upper sqrt ratios, global fee growth, fee growth outside each tick, position-level fee growth snapshots, and the position's liquidity and tokens owed. Combining these elements in one function repeatedly triggered 'stack too deep' compilation errors, even after I attempted to refactor into smaller helpers. The nested structure of Uniswap V3's data flow—slot0, ticks, and positions all interacting—made it hard to keep the call graph shallow while preserving clarity.

Ultimately, the workaround was architectural rather than syntactic. Instead of forcing all valuation logic on-chain, I shifted the most complex parts—fee and IL decomposition—into the off-chain Python layer. LPHelper was left with a focused responsibility: holding capital and mint positions across tick ranges. The Python code then queries the pool, reconstructs amounts and implied fees using high-level arithmetic, and computes investor-level P&L and IL. This design trades on-chain transparency for robustness and observability: the off-chain environment is far easier to instrument, debug, and iterate on than a tightly constrained Solidity implementation.

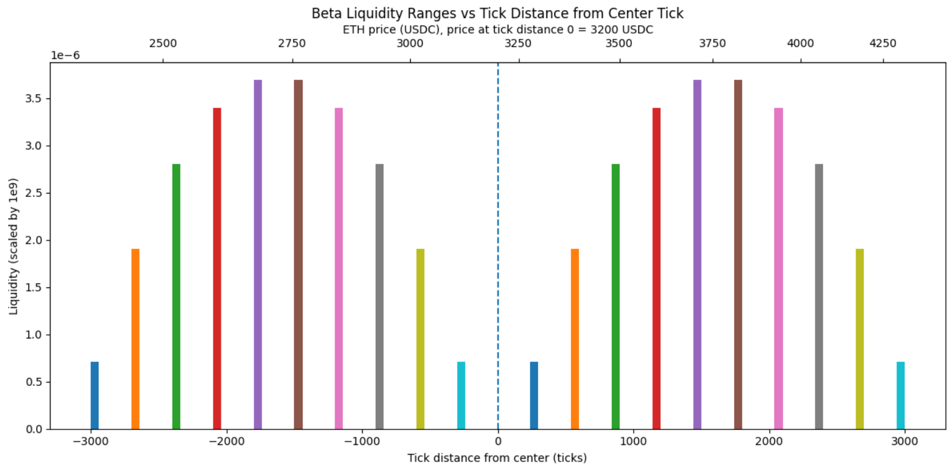
## 3.6 Selected Parameters

For the simulation run for this project, the parameter combinations below were tested (see batch_run.sh for full list). Representative visualizations of selected LP positions follow.

| Alpha | Beta | Num Pools | Pool Separation | Volatility |
|---|---|---|---|---|
| **1** | 1 | 10,20 | 0,60,240 | .01,.05 |
| **1** | 5 | 10,20 | 0,60,240 | .01,.05 |
| **5** | 1 | 10,20 | 0,60,240 | .01,.05 |
| **2** | 2 | 10,20 | 0,60,240 | .01,.05 |



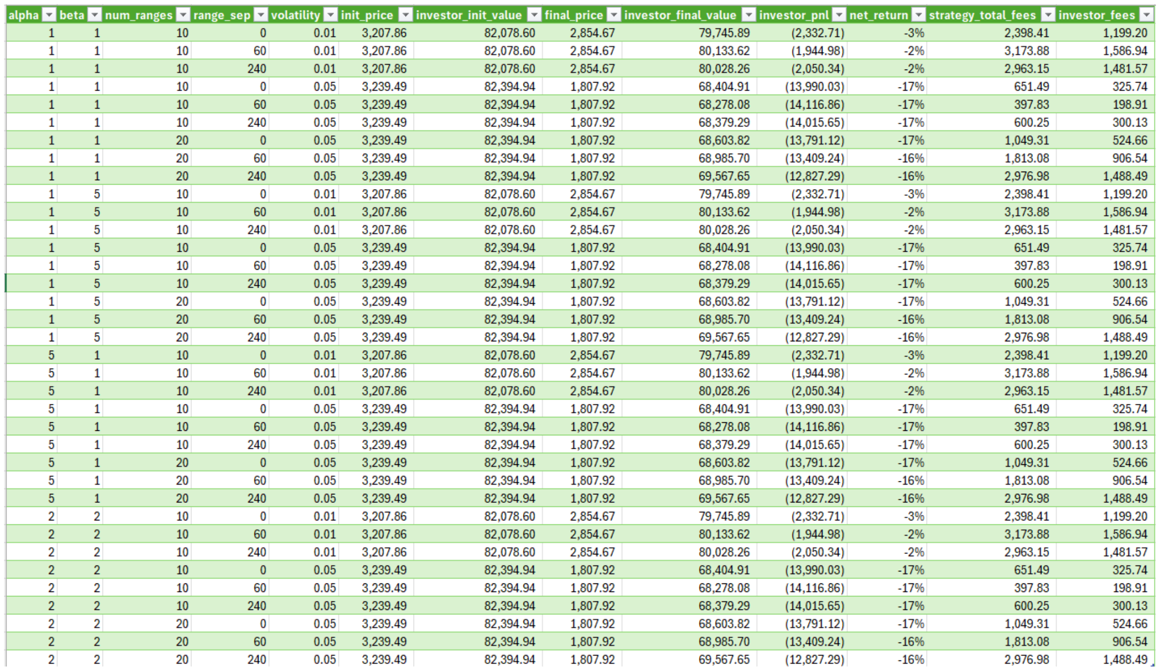
**Alpha=1, Beta=1, Num Pools=10, Pool Separation=0**



**Alpha=2, Beta=2, Num Pools=20, Pool Separation=240**

## 4. Results and Discussion

The table below (derived from the output file summary.csv) summarizes the results of the simulation.



| alpha | beta | num_ranges | range_sep | volatility | init_price | investor_init_value | final_price | investor_final_value | investor_pnl | net_return | strategy_total_fees | investor_fees |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 10 | 0 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 79,745.89 | (2,332.71) | -3% | 2,398.41 | 1,199.20 |
| 1 | 1 | 10 | 60 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 80,133.62 | (1,944.98) | -2% | 3,173.88 | 1,586.94 |
| 1 | 1 | 10 | 240 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 80,028.26 | (2,050.34) | -2% | 2,963.15 | 1,481.57 |
| 1 | 1 | 10 | 0 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,404.91 | (13,990.03) | -17% | 651.49 | 325.74 |
| 1 | 1 | 10 | 60 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,278.08 | (14,116.86) | -17% | 397.83 | 198.91 |
| 1 | 1 | 10 | 240 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,379.29 | (14,015.65) | -17% | 600.25 | 300.13 |
| 1 | 1 | 20 | 0 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,603.82 | (13,791.12) | -17% | 1,049.31 | 524.66 |
| 1 | 1 | 20 | 60 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,985.70 | (13,409.24) | -16% | 1,813.08 | 906.54 |
| 1 | 1 | 20 | 240 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 69,567.65 | (12,827.29) | -16% | 2,976.98 | 1,488.49 |
| 1 | 5 | 10 | 0 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 79,745.89 | (2,332.71) | -3% | 2,398.41 | 1,199.20 |
| 1 | 5 | 10 | 60 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 80,133.62 | (1,944.98) | -2% | 3,173.88 | 1,586.94 |
| 1 | 5 | 10 | 240 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 80,028.26 | (2,050.34) | -2% | 2,963.15 | 1,481.57 |
| 1 | 5 | 10 | 0 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,404.91 | (13,990.03) | -17% | 651.49 | 325.74 |
| 1 | 5 | 10 | 60 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,278.08 | (14,116.86) | -17% | 397.83 | 198.91 |
| 1 | 5 | 10 | 240 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,379.29 | (14,015.65) | -17% | 600.25 | 300.13 |
| 1 | 5 | 20 | 0 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,603.82 | (13,791.12) | -17% | 1,049.31 | 524.66 |
| 1 | 5 | 20 | 60 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,985.70 | (13,409.24) | -16% | 1,813.08 | 906.54 |
| 1 | 5 | 20 | 240 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 69,567.65 | (12,827.29) | -16% | 2,976.98 | 1,488.49 |
| 5 | 1 | 10 | 0 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 79,745.89 | (2,332.71) | -3% | 2,398.41 | 1,199.20 |
| 5 | 1 | 10 | 60 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 80,133.62 | (1,944.98) | -2% | 3,173.88 | 1,586.94 |
| 5 | 1 | 10 | 240 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 80,028.26 | (2,050.34) | -2% | 2,963.15 | 1,481.57 |
| 5 | 1 | 10 | 0 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,404.91 | (13,990.03) | -17% | 651.49 | 325.74 |
| 5 | 1 | 10 | 60 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,278.08 | (14,116.86) | -17% | 397.83 | 198.91 |
| 5 | 1 | 10 | 240 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,379.29 | (14,015.65) | -17% | 600.25 | 300.13 |
| 5 | 1 | 20 | 0 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,603.82 | (13,791.12) | -17% | 1,049.31 | 524.66 |
| 5 | 1 | 20 | 60 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,985.70 | (13,409.24) | -16% | 1,813.08 | 906.54 |
| 5 | 1 | 20 | 240 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 69,567.65 | (12,827.29) | -16% | 2,976.98 | 1,488.49 |
| 2 | 2 | 10 | 0 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 79,745.89 | (2,332.71) | -3% | 2,398.41 | 1,199.20 |
| 2 | 2 | 10 | 60 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 80,133.62 | (1,944.98) | -2% | 3,173.88 | 1,586.94 |
| 2 | 2 | 10 | 240 | 0.01 | 3,207.86 | 82,078.60 | 2,854.67 | 80,028.26 | (2,050.34) | -2% | 2,963.15 | 1,481.57 |
| 2 | 2 | 10 | 0 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,404.91 | (13,990.03) | -17% | 651.49 | 325.74 |
| 2 | 2 | 10 | 60 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,278.08 | (14,116.86) | -17% | 397.83 | 198.91 |
| 2 | 2 | 10 | 240 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,379.29 | (14,015.65) | -17% | 600.25 | 300.13 |
| 2 | 2 | 20 | 0 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,603.82 | (13,791.12) | -17% | 1,049.31 | 524.66 |
| 2 | 2 | 20 | 60 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 68,985.70 | (13,409.24) | -16% | 1,813.08 | 906.54 |
| 2 | 2 | 20 | 240 | 0.05 | 3,239.49 | 82,394.94 | 1,807.92 | 69,567.65 | (12,827.29) | -16% | 2,976.98 | 1,488.49 |

Several observations can be made from these results.

### 4.1 Symmetric Beta Policies Yield Minimal IL Near Spot

The simulation consistently showed that when the final market price remains near the initial price, IL approaches zero. This is due to the symmetrical structure of the ranges: the LP sells ETH at slightly higher levels and buys it back slightly lower, producing a net effect close to HODL. All strategies lost money due to the ending price of ETH being lower than the starting price.

### 4.2 Fee Income Behavior

Wide ranges or low volatility produce negligible fees. Narrower bands produce higher fee income but greater IL sensitivity. Arbitrage significantly increases fees by generating additional volume.

## 4.3 LP Performance in Trending Markets

Consistent with theory, LPs underperform HODL during strong directional trends. Static ranges cannot adapt to persistent price movement, leading to skewed token holdings and IL.

## 4.4. Discussion

Using the output summary, I examined how changes in range geometry, range separation, and volatility affect investor outcomes. Across all configurations, the dominant driver of net return was the directional movement of ETH rather than the exact placement of liquidity bands.

At low volatility ($\sigma = 0.01$), different choices of range separation (0, 60, 240 ticks) produced very similar net returns, typically between −2.4% and −2.8%. In each case, the final ETH price ended below all LP ranges, so the investor effectively exited in all USDC. Under these conditions, the number and spacing of ranges had almost no impact on terminal portfolio value: the LP's risk was essentially that of a long-ETH investor who is forced to liquidate after a moderate price drop.

When volatility was increased to $\sigma = 0.05$, the picture changed dramatically. The final ETH price fell much more aggressively, and investor net returns dropped to the −17% range. However, impermanent loss remained negligible in absolute terms (on the order of $10^{-4}$ USDC), because the price left the LP ranges early and stayed outside. Once the pool price leaves all the investor's ranges, the position is effectively fully converted into USDC and stops experiencing further IL shocks. In this scenario, nearly all losses are simply directional: the value of 10 ETH has collapsed in USDC terms, and the LP configuration no longer plays a meaningful role.

Fees follow a complementary pattern. In the low-volatility runs, the price path spends more time inside the active LP region, so both strategy-level fees and investor fees accumulate to non-trivial values (hundreds to low thousands of USDC). As volatility increases and the price path exits the LP bands, fee generation falls sharply. This confirms the intuition that concentrated LPs are only paid when order flow traverses their ranges; once the price moves away and stays away, the economic role of the LP effectively ends.

Taken together, these experiments show that in a static beta-range configuration, the investor’s P&L is dominated by the terminal ETH price and path of the underlying, not by the fine details of range geometry. To see meaningful differences across beta policies, one would need either a price process that oscillates within the LP bands or a dynamic policy that actively repositions liquidity as the market moves.

The findings confirm that static beta-range LP strategies are fundamentally risk-management tools rather than alpha-generation mechanisms. Their primary benefit is reducing exposure variance and smoothing IL outcomes. However, without significant fee income or volatility, such strategies do not outperform HODL.

Importantly, LP outcomes are dominated by market regime. Mean-reverting and oscillatory markets provide opportunities for fee capture. Trending markets lead to systematic underperformance, consistent with prior research.

## 5. Future Research

The following areas may be fruitful to pursue based on this study:

• Active liquidity strategies (volatility targeting, momentum-based repositioning)
• Multi-agent simulations with dedicated arbitrage actors
• Gas-cost modeling for realistic strategy evaluation
• Cross-pool (multi-asset) strategies
• Machine learning optimization for range selection
• Calibration to historical Uniswap data

These extensions would transform this simulation framework into a practical tool for real LP strategy evaluation.

## References

1. Jerome, Joseph, *et al.* Market Making with Scaled Beta Policies 2022.
2. Wang, Yongzhao, *et al.* Market Making with Learned Beta Policies ICAIF 2024.