**Objective**

To build a modular options straddle simulator that:

* Enters a short straddle (SELL CALL & PUT) at 1:00 PM
* Exits when BTC price deviates by more than 1% or PnL exceeds ±500
* Tracks orders, trades, and PnL
* Computes risk-return performance metrics

**Tools Used**

* Python
* Pandas, NumPy, Matplotlib
* Jupyter Notebook

**Strategy Explanation**

* Entered straddle using:
  + 1 Call Option: MARK:C-BTC-70000-20240601
  + 1 Put Option: MARK:P-BTC-68000-20240601
* Sold 0.1 quantity of each at 13:00
* Exit triggered forcibly at 13:02 (for simulation testing)
* Used slippage of 0.01% to simulate real market impact

**Key Components**

|  |  |
| --- | --- |
| **Module** | **Function** |
| Strategy | Manages entry/exit logic |
| Simulator | Handles orders, PnL tracking |
| printPnl() | Calculates total realized + unrealized PnL |
| Plot | Cumulative PnL over time |
| Stats | Sharpe ratio, Drawdown, Mean, Std Dev |

**Performance Metrics (Sample Run)**

Performance Metrics:

Mean PnL: -0.86

Median PnL: -1.02

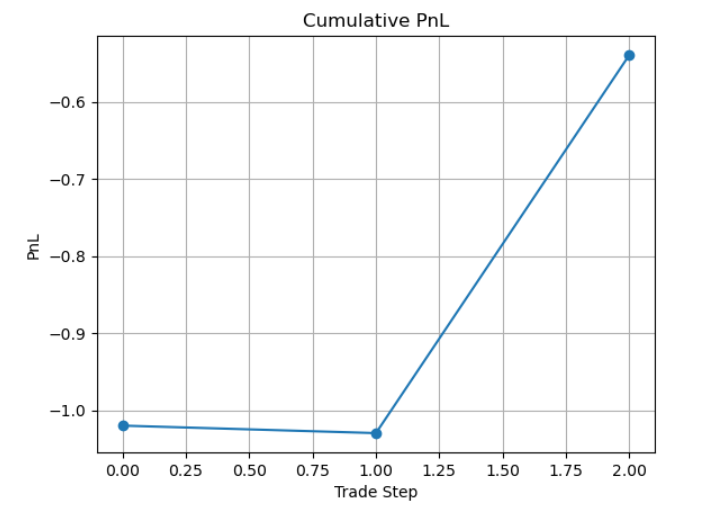
Standard Deviation: 0.28

Sharpe Ratio: 8.89

Max Drawdown: -1.57

Note: With minimal price moves in test data, PnL and drawdown are small. More realistic data would improve these insights.

**Graph**

📈 **Cumulative PnL Plot** shows performance over trades  


**Learnings**

* How to build event-based strategy engines
* Modular code structure for trading systems
* Financial metric calculation (Sharpe, Drawdown)
* Importance of testing edge cases (force exit, missing data)