Autonomous Bitcoin Trading

Documentation

Gymnasium Thun

26gs

Submitted by:

Juri Stoffers

Supervised by:

Dr. Geoffrey Ostrin

August 20, 2025

Abstract

This thesis examines the development of an autonomous Bitcoin trading algorithm, based on the observation that human emotion leads traders to act irrationally in the markets. The research combines technical analysis with algorithmic trading, employing a quantitative methodology that includes data collection, strategy development using Python, and testing strategies on historical price data. The implementation involves creating a custom backtesting framework, developing rule-based trading strategies, and deploying a live trading system on a Linux based server.

Foreword

My interest in Bitcoin began in early 2022 after reading *The Bitcoin Standard by Saifedean Ammous*. The book explores the history of money and argues that Bitcoin is a revolutionary form of hard money [1] superior to traditional currencies. The book sparked an interest in me, and I was drawn into the world of crypto. As a result, I spent more and more time watching charts and losing my mind over them. The idea of making profit from these seemingly random price movements fascinated me. Trading profitably is straightforward: sell high and buy low. The practical implementation, however, is filled with uncertainty and fight against your own instincts and emotions. In addition to getting into crypto I also tried to learn to code since I wanted to make money on the side without having to leave my house. The first time I learned about automated trading algorithms I thought it was one of the hardest things to do. The combination of knowledge made me admire people who code in the finance space. It felt impossible but the more I tried the closer I got to something into the direction of a trading algorithm. With my matura thesis approaching I was pretty confident that I would be able to build an automated Bitcoin trading algorithm.

I had no idea what I was getting myself into, and if I had known before, I would probably have never started. I spent more time than I intended and completely lost myself during the summer vacation. I totally forgot about the writing part and spent a lot of time just coding. This project grew much bigger than I initially thought, but I'm proud to present both the work outcome and the learnings I gained from it.

---

[1]A form of money that maintains its value over time, offering stability and reliability

Contents

1 Introduction

This thesis explores whether I can develop an autonomous trading algorithm that can generate me a consistent profit. To that end my thesis will cover the entire development cycle of an algorithmic trading strategy. The process will include research, finding a trading strategy, backtesting, and live trading algorithm results.

The work is divided in three parts: Finding a strategy, Backtesting, and Live trading.

Finding a strategy can be done in multiple ways just by watching markets over longer time periods to get ideas, reading books about how other people were able to develope strategies and steal logic from them. In my case I will try to find a strategy by analysing the orderbook delta and identifying trends on the not stationary bitcoin price. [2]

Backtesting is the process of testing a trading strategy on historical data to evaluate its performance. It is used to see if the strategy is worth automating.

Live trading is the process of executing a trading strategy on a live market. Here I will see if my strategy works in the real market and how it will behave. The strategy will be automated in python. A major difference is that I will use two different python programs for the backtesting process and the live trading algorithm.

The project requires know how in three key areas: Statistics, Programming and understanding how the cryptocurrencies market operates. I need to apply stasticial rules to check if my idea is worthless or has potential to make me a profit if I automate the execution process. Programming is just needed to make the process of testing or applyinng my ideas easy on large datasets since I have to test on timeseries up including up to 200'000 values. The algorithm will be written in Python and will run 24 hours a day seven days a week.

By the end of the project, I aim to deliver the following components:

- Backtest results
- Trading strategy
- Trading algorithm programm
- Live execution results

---

[2]Not stationary means not being still or fixed in one place; moving or changing rather than remaining constant.

2 Theoretical Background

What is trading?

Trading means buying and selling financial assets like stocks, cryptocurrencies, or commodities, usually with the goal of making a profit. Traders try to buy at a low price and sell at a higher one, or profit from falling prices by selling high and buying back lower. Prices in financial markets change constantly, which creates opportunities — but also risks. There are many different types of trading, depending on the chosen timeframe, strategy, and how trades are managed. Losing is part of every strategy the goal is to lose less money than you win. As trading is often seen as an easy way to make money it is important to understand that it is not easy. It is a skill that requires a lot of time and effort to master. Type of trading strategy does not matter there is no easy or fast way to success in trading.

Algorithmic Trading

In this thesis I will try to find success in the algorithmic trading niche, also know as Algo-trading or automated trading. This is a type of trading which uses computer programs to execute trades in finacial markets based on predefined rules or mathematical models. Inside of algorithmic trading there are again many different types. Some examples are: statistical arbitrage, trend following, and High-Frequency Trading. In terms of logic all these are not related at all and strategies are often very complex in these areas.

My strategy is best descirbed as a statistical signal-based strategy. As I try to find a metric that shows statistical evidence on which I can capitalized on. In financial markets we can profit by betting on a volatilty change [3] or directional bias. Capitalizing from directional bias is as simple as buying if your directional bias positive and selling if we have a negative directional bias. (I am reffering to positive and negative returns)

---

[3] a measure of how much and how quickly the price of a trading instrument fluctuates over time

Efficient markets

Making consistent profits in financial markets is difficult. Prices move in ways that often seem unpredictable, and most traders both discretionary and systematic struggle to gain a lasting edge. [1]One explination for this is the Efficient market hypothesis (EMH) which suggest that asset prices already reflect all available information, leaving little room for profitable oppertunities.

If the EMH were fully accurate in practice, then even the most advanced trading algorithms and best discretionary traders would not be able to consistently beat the market. [2] Yet markets are not perfectly efficient, being heavily influenced by emotion, fear, and herd behaviour. Traders often fail to act on clear signals simply because it is psychologically hard to buy on a red day or sell while everybody is full of euphoria.

To make things even better depending on where you trade market can change in efficient. But still some of the most valuable assets like stocks, gold and Bitcoin prices are a reflection of human emotion pricing in information. Especially now with Trump in office a lot of volatility came in with big opportunities

Depending on the timeframe there are still a lot inefficiencies from which individuals can make a profit from. I will try to act on short term inefficiencies with my trading algorithm because I only have about one month for the live test and it would not make sense to run an algorithm which holds a trade over weeks and takes a trade every two months. In addition my initial idea when I started of with my matura thesis was already based on short term signals.§

It is possible to make a profit in the markets but it is not easy, not for everyone, and needs dedication. I am pretty sure I am able to make a profit. The question rather is if it is possible to do so in the limited time I have and at what scale.

---

[1]An edge is a consistent advantage over the market. It can be achieved through superior information, better analysis, or a unique trading approach.

[2]Beating the market means achieving a higher ROI ("Return on Investment") than a certain benchmark. A common benchmark is the S&P 500, which is an index including the 500 largest companies listed on U.S. stock exchanges. The S&P 500 has had an average yearly return of about 10.33% since 1957

Development of a Systematic Strategy

A systematic strategy is a trading strategy which uses fixed entry and exit conditions. Entry conditions in systematic trading strategies can vary from entries based on news, technical indicators, or other external factors. In contrast to discretionary trading where the trader decides whether to enter or exit a trade based on human emotion or intuition.

Idea Generation: The systematic strategy has to be based on a hypothesis that once signal X occurs the market is wrong and we can profit from the market correcting back to a fair value. Finding an idea to test on if there is an edge to it is the first step. This can be done by copying other strategies and applying them in a different way or in a different market. The approach I took was trying to make a discretionary strategy into a systematic one.

Backtesting: After coming up with an idea we have to make a stric entry and exit conditions. We test the rules on historical data. This allows the developer to evaluate how the strategy would have performed in the past. This allows us to get a hint on how the strategy would perform in the future and if it is worth automating.

Optimization and Robustness Testing: After initial tests, parameters can be adjusted. A simple example would be adjusting the holding time of a trade. If a strategy ends up to not be profitable with a trade holding time of 15 minutes this doesn't mean the strategy is not profitable. It just means that the strategy is not profitable with a trade holding time of 15 minutes. Some inefficiencies can be corrected in minutes others might take a day or two. Holding time is a simple example which doesn't bring a lot of risk with it if we test on different holding times. As long as we use a limited amount and have clear differences between the different holding times we can be sure that the strategy is robust.

Live Execution: If the backtesting results are promising and the strategy appears robust, it can be deployed live. Execution is handled by a bot or script running on a server, which listens to market data and acts as soon as the conditions are met. New things have to be taken in consideration here whicha are not a problem in backtests. One such thing is the time it takes to calculate the entry and exit conditions. This is a problem, as the price might change between when the actual entry conditions occur and when the entry signal is calculated.

Throughout this process, the focus remains on consistency and measurable results. Unlike human decision-making, a systematic strategy must behave identically entry and exit conditions. This makes it possible to evaluate, improve and automate trading in a transparent way.

3 Methodology

3.1 Project Overview

The aim of this project was to design, test, and deploy a fully automated trading strategy for Bitcoin. The development process consisted of four main stages. First, historical price data was collected through a public exchange API and fetched in one minute intervals stored inside a database on a PaaS [1]. In the second stage, a simple rule-based strategy was defined based on technical indicators, with clearly specified entry and exit conditions. Next, the strategy was backtested on several months of historical data using a custom Python-based framework to evaluate its performance across different market conditions. Finally, if the results met predefined criteria (e.g. acceptable loses, stable returns), the strategy was deployed live on a cloud server that connects to the exchange and executes trades automatically in real time. Each stage of the process is documented in detail in the following sections.

3.2 Data collection

I chose to create my own dataset because it offers a key advantage: it allows me to experiment with ideas and patterns that are less likely to have been explored before. This increases the chance of discovering something new that might be missed when using more commonly available datasets.

In my dataset, I calculate the 1%, 2.5%, and 5% delta of the order book. Additionally, I include the current Bitcoin price and the current timestamp. The program I wrote collects data by sending API requests [2] to the Coinbase Exchange API at one-minute intervals. It retrieves both the current Bitcoin price and a full snapshot of the order book at that moment. A complete snapshot is necessary to calculate order book deltas at different percentage depths accurately.

The script is hosted on my PaaS where my database (Postgres) is also hosted where all the data is stored. It has been running 24/7 since March 11, 2025. Live datamining progress

The orderbook delta is calculated by summing the total value of bid orders $n\%$ below the current price (P_t) and subtracting the total value of ask orders that are $n\%$ above the current price (P_t).

$$\Delta_{OB} = \sum_{i=1}^{n} V_{bid_i} - \sum_{i=1}^{n} V_{ask_i}$$

---

[1] Stands for Platform as a service and it's a cloud computing model which gives developers a platform to host their applications so that they are globaly accesable on the internet and run 24/7

[2] API stands for Application Programming Interface. It is a tool that allows programs to request and receive specific data from external services in this case, used to fetch real-time Bitcoin price and orderbook data from the coinbase crypto exchange.

3.3 Designing and Developing Systems

The most difficult part of developing a trading algorithm is finding a strategy that is actually worth integrating. As explained in the Efficient Markets section, it is in principle possible to find a strategy that generates profit. The problem is that finding a strategy which consistently makes money and outperforms the market is still extremely difficult.

Before I start searching, I need to define what I am searching for. To do this, I use the S.M.A.R.T. goal framework (Specific, Measurable, Attainable, Relevant, and Time-bound), as described in the book *Building Winning Algorithmic Trading Systems* by Kevin J. Davey.

Goals

- Two months allocated for finding a strategy

- Strategy operates on the Bitcoin/USD pair using Hyperliquid

- Target Sharpe ratio between 1 and 2 [4]

- Approximately 20 trades per month

- No strict requirement to outperform the market; focus is on stability and consistency

The area where I saw the most potential was the order book delta. My hypothesis was that, since it directly reflects the relationship between buyers and sellers at specific price levels, it could serve as the basis for a statistical signal. By running volatility and directional bias tests under various order book delta conditions, I aimed to identify patterns that might be predictive of short-term price movements.

---

[4]The Sharpe ratio is a measure of risk-adjusted return. It compares the average return of a strategy to its volatility. A higher Sharpe ratio indicates a better balance between risk and return. A Sharpe ratio between 1 and 2 is considered decent in most financial contexts.

3.4 Searching/Backtesting

As stated in my hypothesis: I think that the orderbook delta directly reflects the relationship between buyers and sellers at specific price levels. A strategy based on a negative order positive orderbook delta would not include enough context.
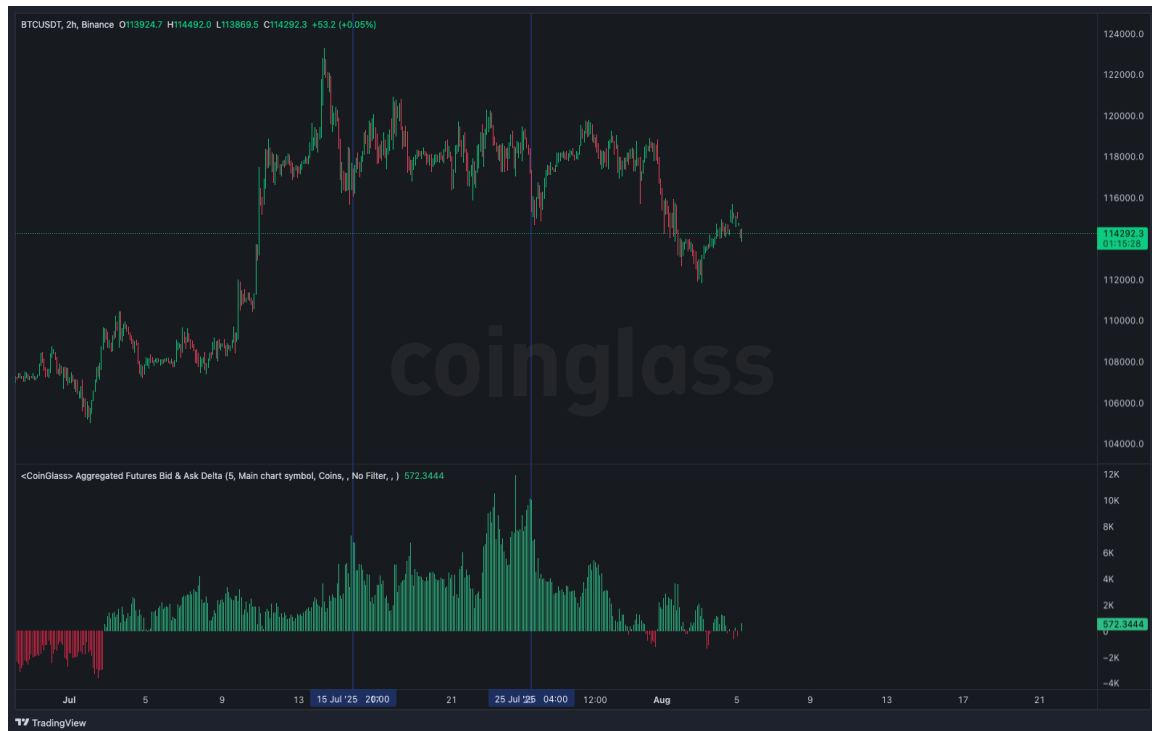

Figure 1: Orderbook delta visualization showing price action (top) and corresponding delta values (bottom)

As shown in Figure 1, the orderbook delta provides valuable insights at extreme values. The chart displays two key components:

- The price action (top chart) shows the actual Bitcoin price movements

- The orderbook delta (bottom chart) represents the imbalance between buy and sell orders

A high orderbook delta value indicates substantial buy orders below the current price. This creates a strong support level, as sellers would need significant pressure to break through these orders. Even if sellers manage to push through, these accumulated buy orders often create buying pressure that can lead to price support or potential rebounds.

This relationship between orderbook delta and price action is particularly useful when:

- The delta reaches extreme values, indicating significant order imbalance

- There's a clear divergence between price action and orderbook delta

- Multiple timeframe analysis shows consistent patterns

Example:

```
#Example for entry signals in the code
outlier_mask = (df['trend'] == 'uptrend') & (df['is_outlier'] == 1)
```

Then I test the strategy on one month of data to evaluate whether it shows any potential. I consider a strategy to have potential if it is around break-even [5] or slightly profitable and achieves a Sharpe ratio between 1 and 2.

Results:
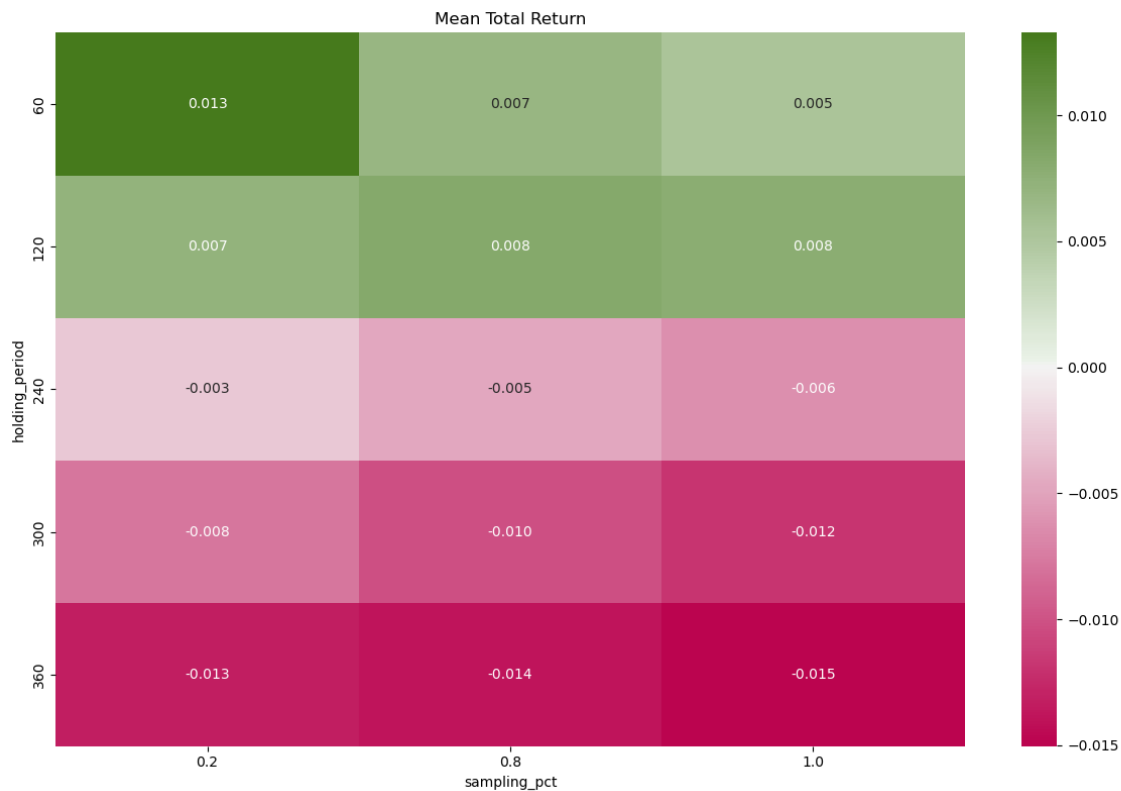
On the X axis you can see the sample size and on the Y axis you can see the average return over he different test runs.

---

[5]Break-even is the point where the total profit equals the total loss. In other words, it's the point where the strategy neither makes nor loses money.

4 Strategy backtest implementation

4.1 Sharp ratio

The sharp ratio will be the main strategy evaluation metric in my backtest

$$\text{Sharpe ratio} = \frac{\text{Average return } t}{\text{Standard deviation } t^6} \cdot \sqrt{98280}^7 \tag{1}$$

4.2 Backtesting results

- average sharpe ratio 1.55 [8]

- Over a time period of 137 days a return of 2.7% was achieved

4.3 Backtest parameters

- A stop lost of 2%

- Exit is 120 minutes after entry or if a regime change occurs

- fees of 0.0003% and slippage of 0.0001%

- accumulate $=$ True [9]

4.4 Code entry conditions

```
# Entry conditions
entries = (df_temp['outlier_context'] == 'b') & (df_temp['trend'] == 'Uptrend')


# Exit after 120 minutes
time_exit = exitSlots.shift(120).fillna(False)


# Exit on trend change
trendChange = (df_temp['trend'] != df_temp['trend'].shift(1))


#Or statment to exit
exits = trendChange | time_exit
```

---

[7]98280 is used to annualize the Sharpe ratio we do this in order to compare the Sharpe ratio with the Sharpe ratio of other strategies or funds

[8]The $\overline{SR}$ is the average Sharpe ratio over different walkforward periods

[9]This allows us to open multiple trades at the same timei

4.5 Entry conditions calculations

I have to give more context about the entry conditions. df_temp is just a variable to which I assign a timeseries dataframe [10] with different kind of metrics inside of it. The entry conditions are defined by two boolean conditions:

4.5.1 Outlier context

```
df_temp['outlier_context'] == 'b'
df_temp['trend'] == 'Uptrend'
```

As shown in Figure 2, the dataframe contains various metrics that we use for our trading decisions:

```
uτ1ͻ Len:  ϚϛϚϞϛϠ
                  delta1   delta2_5   delta5      price         ema directional_pressure is_outlier outlier_signal_strength outlier_context trend
timestamp
2025-03-11 23:37:01 -1.786011 -29.817527 -51.350122 82681.515        NaN                 NaN          0                    0.0                       NaN
2025-03-11 23:38:01 -9.159898 -28.886296 -59.164552 82658.855 82681.515000                NaN          0                    0.0                       NaN
2025-03-11 23:39:01  0.963591 -17.596764 -48.747833 82758.125 82681.483550                NaN          0                    0.0                       NaN
2025-03-11 23:40:02 -31.762113 -37.069179 -48.975283 82838.605 82681.589922                NaN          0                    0.0                       NaN
2025-03-11 23:41:01  1.521426 -15.890205 -46.926685 82742.585 82681.807847                NaN          0                    0.0                       NaN
```

Figure 2

The column $is\_outlier$ is a binary signal column if the current current orderbook delta value Δ_t is further than two standard deviations from the mean $\overline{\Delta}_{1440}$ of the orderbook delta. $outlier\_context$ is a string column that has a value of **b** if Δ_t has a postive Z score and **s** if it has a negative Z score.

4.6 Market Structure and Trend Identification

To identify trends in Bitcoin's price, we use a combination of price movement analysis and moving averages. The main challenge is to distinguish between real trends and temporary price movements. We start by converting our 1-minute price data into OHLC [11] 240 minute data to better see the overall market direction.

4.6.1 Price Movement Analysis

The core of our trend analysis looks at how price moves over time. We use two main indicators:

1. The derivative (rate of change) of a moving average:

$$\frac{d}{dt}\text{MA} = \text{Current MA value} - \text{Previous MA value}$$

---

[10]A dataframe is a two dimensional array with rows and columns

[11]OHLC stands for Open, High, Low, and Close. It is a type of data that is used to represent the price action of an asset over a period of time and is also known as a candlestick chart.

When this derivative is positive, it indicates an upward trend, and when negative, a downward trend.

2. Price levels comparison: We compare current prices with previous highs and lows using a window of time (w). This helps us confirm if we're really in a trend:

$$\text{Trend}_t = \begin{cases} \text{Uptrend} & \text{if price is making higher highs} \\ \text{Downtrend} & \text{if price is making lower lows} \\ \text{Ranging} & \text{otherwise} \end{cases}$$

4.6.2 Support and Resistance

We also look at support and resistance levels to strengthen our trend analysis:

- Support: Price levels where buying pressure tends to stop price from falling further

- Resistance: Price levels where selling pressure tends to stop price from rising further

The slopes of these levels help us determine trend strength. For example, rising support and resistance levels indicate a strong uptrend, while falling levels suggest a downtrend.

4.6.3 Final Trend Determination

To make the final decision about the trend, we combine all these factors:

$$\text{Final Trend} = \begin{cases} \text{Uptrend} & \text{if } \frac{d}{dt}\text{MA} > 0 \text{ and making higher highs} \\ \text{Downtrend} & \text{if } \frac{d}{dt}\text{MA} < 0 \text{ and making lower lows} \\ \text{Previous Trend} & \text{if moving average confirms} \\ \text{Ranging} & \text{if no clear direction} \end{cases}$$

This approach helps us filter out market noise and identify real trading opportunities. We only take trades when all these factors align, which increases our chances of success.

4.7 Related Work

Discussion of related work.

5 Methodology

Your methodology description.