Trading Strategy Optimisation with Nature-Inspired Algorithms

Your Name

April 20, 2025

1 Data Preparation

We use a high-frequency Bitcoin dataset obtained from Kaggle, containing historical trading data at 1-minute intervals from 2012 to 2025. The raw dataset includes timestamps, Open, High, Low, Close prices, and volume. To align with the strategy's daily signal generation, we preprocess the data as follows:

- **Datetime Parsing**: Convert Unix or string-formatted timestamps to `datetime` objects using pandas.
- **Resampling to Daily Frequency**: We extract the daily closing price using:

```
daily_df = df['Close'].resample('D').last().dropna()
```

- **Train-Test Split**: To evaluate generalisability, we split the data temporally:
  - `train_df`: All data before January 1, 2020
  - `test_df`: All data from 2020 onwards
- **Missing Values**: Any days without valid close prices are dropped.

This setup provides a clean and consistent price series for downstream strategy execution and optimisation.

2 Trading Strategy Design and Optimisation

2.1 Overview

We design a parameterised, rule-based trading strategy that fuses multiple technical indicators to form a composite decision signal. The goal is to maximise the final equity of a simulated trading account by optimising the strategy's parameters using nature-inspired metaheuristic algorithms.

2.2 Composite Strategy Formulation

The trading strategy integrates signals from three key technical indicators:

- **Exponential Moving Average (EMA) Crossover**: Detects trend changes by comparing fast and slow EMA windows.
- **Relative Strength Index (RSI)**: Captures momentum signals indicating overbought or oversold market conditions.
- **Bollinger Bands (BB)**: Identifies volatility extremes through standard deviation bounds.

Each indicator produces a discrete signal:

$$\text{Signal}_i(t) \in \{-1, 0, +1\}$$

The final decision signal is computed using a weighted average:

$$S(t) = \frac{w_1 \cdot f_{\text{EMA}}(t) + w_2 \cdot f_{\text{RSI}}(t) + w_3 \cdot f_{\text{BB}}(t)}{w_1 + w_2 + w_3}$$

A trade is executed when:

$$\text{FinalSignal}(t) = \begin{cases} +1 & \text{if } S(t) > \theta_{\text{decision}} \\ -1 & \text{if } S(t) < -\theta_{\text{decision}} \\ 0 & \text{otherwise} \end{cases}$$

2.3 Parameter Space

The strategy is controlled by a 10-dimensional continuous parameter vector:

- d_1, d_2: EMA fast and slow windows
- d_3: RSI window size
- θ_{RSI}: RSI threshold
- d_4, σ: BB window and standard deviation multiplier
- w_1, w_2, w_3: Weights for the three signals
- θ_{decision}: Decision threshold

2.4 Backtesting Procedure

We simulate a trading account that alternates between full cash and full BTC positions based on the trading signal. The account starts with \$1000 and applies a 3% transaction fee on every buy or sell. At the end of the evaluation period, any remaining BTC is liquidated.

The fitness of a given strategy is defined as the final equity (cash value) of the account:

$$\text{Fitness} = \text{Final Equity} = \text{Cash}_{\text{end}}$$

2.5 Metaheuristic Optimisation

To search for the optimal parameter configuration, we apply three nature-inspired algorithms:

- Whale Optimization Algorithm (WOA)
- Grey Wolf Optimizer (GWO)
- Particle Swarm Optimization (PSO)

Each optimiser iteratively evaluates candidate solutions on the training dataset, searching to maximise the fitness score.

2.6 Test Set Evaluation

The best parameter sets from each optimiser are evaluated on a temporally disjoint test dataset. We compare their generalisation performance by visualising equity curves, analysing return statistics, and inspecting buy/sell signal placements over time.