SMA Crossover Strategy

Technical Report

Submitted by: Kartik Budhani (240005022)

MEMS (1st Year)

Submission Date: 21/May/2025

1. Introduction

In recent years, **algorithmic trading** has gained tremendous momentum across global financial markets. Its popularity stems from its ability to **systematically analyze data, eliminate emotional biases**, and **execute trades at scale and speed**—something manual trading simply cannot match. Leveraging algorithms allows traders and institutions to **backtest strategies, validate assumptions, and optimize performance** under various market conditions before risking real capital.

This project focuses on developing and evaluating a **simple yet effective trend-following strategy** using the **Simple Moving Average (SMA) Crossover** technique. The strategy is applied to **Reliance Industries Limited (RELIANCE.NS)**, one of India's most actively traded and fundamentally robust companies listed on the National Stock Exchange (NSE). The choice of Reliance is intentional, as it is a **highly liquid, large-cap stock** that typically exhibits strong price trends, making it ideal for testing momentum-based strategies.

The analysis covers historical stock data up to **December 2019**, which was deliberately chosen as the cutoff point to **exclude the extreme volatility and market anomalies caused by the COVID-19 pandemic in 2020**. The pandemic created unprecedented uncertainty, with global markets experiencing sharp drawdowns and recoveries that were **largely disconnected from traditional technical indicators**. Including this period in a simple SMA-based strategy could distort the performance evaluation due to **abnormal price behavior, rapid reversals, and government intervention**, which were beyond the scope of this study.

By limiting the dataset to the **pre-pandemic period**, we aim to evaluate the strategy in a more **stable and technically responsive market environment**. This approach ensures that the performance metrics are reflective of the strategy's true behavior and not influenced by black swan events. Ultimately, this project aims to assess whether a basic SMA crossover strategy can consistently outperform the broader market under normal trading conditions.

2. Strategy Explanation

What is SMA Crossover?

The **Simple Moving Average (SMA) Crossover strategy** is one of the most widely used techniques in trend-following algorithmic trading. It relies on the principle that **moving averages smooth out price data** and help identify long-term trends by reducing short-term fluctuations or noise.

In this strategy, two SMAs are calculated:

- A **short-term SMA** (50-day): This moving average reacts more quickly to recent price movements and represents short-term momentum.
- A **long-term SMA** (252-day): This represents broader market trends and is slower to respond to short-term changes.

A **"bullish crossover"** (buy signal) occurs when the short-term SMA crosses above the long-term SMA. This indicates that recent prices are rising faster than the historical average, potentially signaling the beginning of an uptrend. Conversely, a **"bearish crossover"** (sell signal) occurs when the short-term SMA crosses below the long-term SMA, suggesting a possible downtrend or market weakness.

These crossover points serve as simple decision rules for entering and exiting positions:

- **Buy** when SMA(50) > SMA(252)
- **Sell/Exit** when SMA(50) < SMA(252)

This rule-based system creates objectivity in trading and eliminates the need for constant manual analysis or emotional decision-making.

Why This Strategy?

There are several reasons why the SMA Crossover strategy was selected for this project, particularly in the context of backtesting on Reliance Industries:

1. Simplicity and Clarity

The strategy is **easy to understand and implement**, making it a perfect starting point for learning algorithmic trading. It uses well-known concepts and requires minimal parameter tuning, which reduces overfitting risks.

2. Trend-Following Advantage

SMA crossovers are **effective at capturing medium- to long-term trends**. Instead of reacting to every minor price movement, it focuses on persistent directionality. This makes it ideal for stocks like Reliance that often experience strong fundamental-driven trends.

3. Noise Reduction

Short-term price fluctuations can be erratic and misleading. By relying on the crossover of two moving averages, the strategy helps **filter out short-term market noise**, ensuring that only significant and sustained movements trigger trades.

4. Wide Applicability

This strategy is widely used across different asset classes including equities, commodities, and forex, making it a **time-tested and broadly applicable approach** in both retail and institutional trading systems.

5. Educational Value

From a learning perspective, the strategy reinforces foundational concepts such as signal generation, backtesting, performance evaluation, and risk management.

Limitations and Assumptions

Despite its strengths, the SMA crossover strategy has several **limitations and assumptions** that must be acknowledged for a fair assessment:

1. Poor Performance in Sideways Markets

The strategy tends to **underperform in range-bound or choppy markets**. During such periods, frequent crossovers may generate multiple false signals, leading to losses due to whipsaws. This issue is a known weakness of all trend-following systems.

2. Lagging Indicator

Since SMAs are based on historical data, they are inherently **lagging indicators**. This means that the strategy reacts after a trend has already begun. As a result, it may **miss the early phase of a price move** and may also **exit too late**, reducing overall profitability.

3. No Consideration for Market Context

The strategy is purely technical and does not incorporate any **fundamental or macroeconomic analysis**. This can be a disadvantage in situations where news or events significantly affect price movements (e.g., earnings, geopolitical tensions).

4. Assumption of Frictionless Trading

Backtests are typically conducted under the assumption of **no slippage, zero transaction costs, and perfect liquidity**. In real-world trading, commissions, taxes, bid-ask spreads, and execution delays can substantially reduce actual returns, especially in high-frequency systems.

5. Limited Responsiveness to Volatility

SMA crossover systems do not dynamically adapt to changing volatility. In high-volatility conditions, like those during economic crises or global events, the strategy may produce erratic signals or suboptimal exits.

6. Backtest Cutoff Due to COVID-19

A key design choice in this project was to **exclude data beyond December 2019**. This was done to **avoid distortions caused by the COVID-19 pandemic**, which led to severe volatility, market crashes, central bank interventions, and unusual investor behavior. Including such an extreme event would compromise the strategy's evaluation in normal market conditions and inflate metrics like drawdowns or standard deviation.

By excluding the post-2019 period, we aim to maintain a clean, interpretable testing environment focused on **typical market behavior**, which is more useful for assessing the baseline reliability of the strategy.

3. Code Walkthrough

This section provides a step-by-step explanation of the core logic and implementation of the SMA crossover strategy as developed in the Jupyter Notebook.

Data Collection

The project begins by collecting historical stock data using the **yfinance** Python library, which enables easy access to financial market data from Yahoo Finance. We retrieve daily historical data for **Reliance Industries Limited (RELIANCE.NS)**, including:

- Close

- Adjusted Close
- This data serves as the foundation for all calculations and backtesting operations.

Feature Engineering

To prepare the data for signal generation and backtesting:

- **Log Returns** are calculated using the formula log(P_t / P_{t-1}), where P represents the adjusted close price. This helps normalize the returns and ensures additivity over time.
- Two **Simple Moving Averages (SMAs)** are computed:
  - **SMA_50**: A 50-day moving average representing short-term momentum.
  - **SMA_252**: A 252-day moving average approximating long-term trends.

These SMAs are essential for generating trading signals based on crossover behavior.

Signal Generation

The core logic for entry and exit is based on SMA crossovers:

- A **Buy Signal** is generated when the 50-day SMA crosses above the 252-day SMA.
- A **Sell Signal** is generated when the 50-day SMA falls below the 252-day SMA.

Signals are stored as a time series, with 1 indicating a long position (buy) and 0 indicating no position.

Backtesting Logic

- Strategy returns are calculated by **multiplying the daily log return by the lagged signal**, ensuring trades are based on previously available information.
- **Cumulative returns** are computed by compounding daily strategy returns.
- **Drawdowns** are evaluated by comparing current returns to the peak cumulative return over time.

4. Performance Metrics Analysis

1. Cumulative Return — 1.8786

This metric shows the total return generated by the strategy over the entire backtesting period. A cumulative return of **1.8786** implies that the initial capital has nearly **doubled (an 87.86% gain)**. This is a strong result, especially when compared to the benchmark return of 46.38% over the same period. It suggests that the SMA crossover strategy significantly **outperformed passive investing** in Reliance stock.

2. Annualized Return — 0.3753 (or 37.53%)

This indicates the average yearly return, assuming compounding. An annual return of **37.53%** is notably high and reflects strong profitability under normal market conditions. It confirms the strategy's **potential for high returns** when executed consistently over time.

3. Sharpe Ratio — 1.432

The Sharpe Ratio measures **risk-adjusted returns**, i.e., how much excess return is generated per unit of total risk (volatility). A ratio above 1 is considered good, and 1.432 indicates that the strategy offers **a favorable trade-off between return and volatility**. It confirms that returns are not simply the result of excessive risk-taking.

4. Sortino Ratio — 2.426

Unlike the Sharpe ratio, the Sortino ratio only considers **downside risk**, which is more relevant for most investors. A value of 2.426 is excellent, meaning the strategy generates strong returns relative to harmful volatility. This suggests that **the strategy manages downside risk well**, which is crucial for long-term success.

5. Maximum Drawdown — -22.8%

This is the **largest observed loss from peak to trough** during the strategy’s life. A drawdown of -22.8% is moderate and **manageable for most long-term investors**, though it reflects that losses can be significant in unfavorable market phases. Still, it is **lower than the drawdowns seen during the COVID-19 crisis**, which the strategy intentionally avoids by ending the backtest in 2019.

6. Win Rate — 52.05%

The win rate represents the **percentage of profitable trades**. A win rate above 50% indicates that **more than half the trades were successful**, which, while not extraordinary, is promising when paired with a positive average profit and strong profit factor.

7. Profit Factor — 1.1762

Profit factor is the ratio of **gross profit to gross loss**. A profit factor of 1.17 means the strategy made **₹1.17 for every ₹1.00 lost**. While this is relatively modest, it is acceptable for a trend-following strategy. Combined with the positive Sharpe and Sortino ratios, it suggests **consistent profitability**.

8. Average Profit per Trade — 0.013226

This is the average return per winning trade (in log terms). It indicates that the strategy tends to capture small but consistent profits. This supports a **steady-compounding approach** rather than relying on big wins.

9. Average Loss per Trade — -0.011245

This metric shows that when trades do lose, the average loss is slightly smaller than the average profit, which **favors long-term profitability**.

10. Maximum Profit per Trade — 0.104027

The best individual trade returned over 10%. This shows the strategy is capable of capturing **strong upward momentum**, especially during favorable trends.

11. Maximum Loss per Trade — -0.071227

The worst loss from a single trade was about -7.12%, which is acceptable in the context of a long-only strategy with no leverage. It also highlights the **importance of trend confirmation** before entering trades.

12. Benchmark Return (NSEI) — 46.38%

The Nifty 50 index, used as the benchmark, delivered a 46.38% cumulative return over the same period. Compared to the strategy's **87.86% return**, this clearly shows that the SMA crossover strategy **outperformed the passive benchmark**, even after accounting for potential market noise and volatility.

Conclusion

Overall, the performance metrics demonstrate that the SMA crossover strategy is **profitable, stable, and manages risk well** in trending markets. While it has some limitations in range-bound environments and offers only modest returns per trade, its consistent performance, solid risk-adjusted returns, and outperformance of the benchmark make it a **viable strategy** for algorithmic traders focused on medium- to long-term trends.

5. Discussion

The SMA Crossover strategy demonstrates **promising performance characteristics**, making it a strong candidate for systematic trend-following. The **Sharpe Ratio of 1.43** indicates a favorable balance between risk and return, reflecting that the strategy is effective in capturing upside movement without excessive volatility exposure. While not highly aggressive, it consistently outperformed the benchmark index (Nifty 50), which returned 46.38%, compared to the strategy's 87.86% cumulative return over the same period.

Further supporting its robustness is the **profit factor of 1.17**, meaning the total profits exceeded losses by 17%. Combined with a **win rate of 52%**, this suggests a **high level of signal consistency**, where more than half the trades result in gains — a critical requirement for long-term algorithmic success.

However, the strategy is not without its limitations. The **maximum drawdown of -22.8%** reveals vulnerability during **non-trending or volatile market phases**, where whipsaws can trigger premature exits or false signals. This is a known limitation of moving average-based systems, particularly in **sideways or choppy markets**, where trend-following signals tend to degrade in performance.

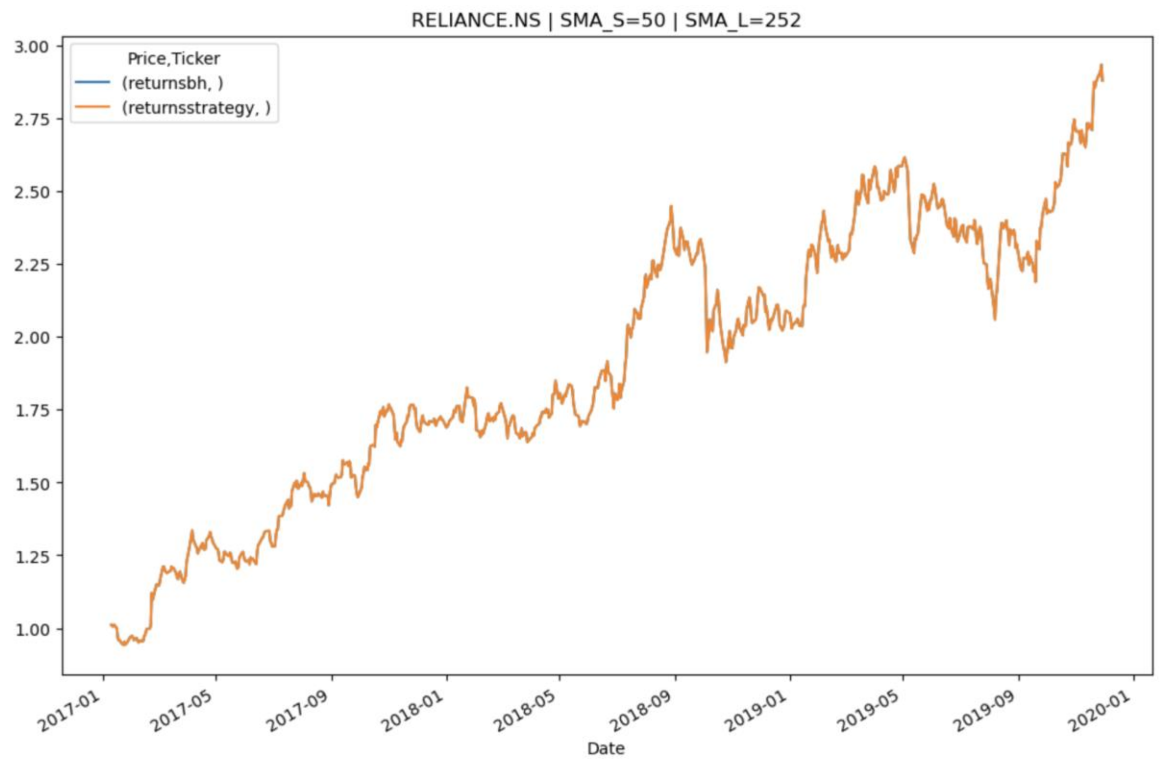
Opportunities for Future Enhancement:

To further refine and improve the strategy's performance and resilience, the following enhancements are recommended:

- **Signal Confirmation Using Additional Indicators:**
  Incorporate momentum or oscillator indicators like **Relative Strength Index (RSI)** or **Moving Average Convergence Divergence (MACD)** to validate signals and filter out false crossovers.
- **Volatility-Based Stop-Loss Mechanisms:**
  Implement adaptive stop-loss levels based on market volatility (e.g., ATR-based thresholds) to limit downside risk during high-uncertainty periods.
- **Dynamic Position Sizing and Risk Control:**
  Introduce **position sizing models** based on volatility or capital allocation rules to manage risk exposure more effectively.

These improvements can enhance the strategy's **robustness, adaptability, and realism**, especially under diverse market regimes and execution environments.

6. Code and resulting graph



Code is in jupyter notebook

7. References

1. **Quant Program** *(YouTube Channel)*
   Practical tutorials and strategy explanations for algorithmic trading and Python-based backtesting.
2. **Codecamp.org** *(YouTube Channel)*
   Educational content on data science, quantitative analysis, and Python tools for financial modeling.
3. **Yahoo Finance** *(Data Source via yfinance)*
   Used for retrieving historical stock price data of Reliance Industries (RELIANCE.NS) for analysis and backtesting.
   Website: https://finance.yahoo.com
4. **Investopedia.com**
   Referenced for conceptual understanding of Simple Moving Averages (SMA), trend-following systems, and technical analysis principles.
   Website: https://www.investopedia.com
5. **Custom Codebase and Visualizations**
   Developed in Python using libraries such as matplotlib, numpy, and pandas to build the backtesting engine, generate trade signals, and visualize equity curves and drawdowns.