Project Phase II Report

On

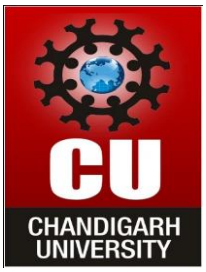
Advanced algorithmic trading system

Submitted for the requirement of

Project course

BACHELOR OF ENGINEERING

COMPUTER SCIENCE & ENGINEERING



Submitted to:
Project Teacher (Supervisor)
Shuvendu Das (12496)

Shuvendu Das

Submitted By:
Student Group

NAME	UID
Ayush Raj	**20BCS7576**
Lekhraj Sharma	**20BCS7553**
Shashwat Verma	**20BCS7532**
Shubham Kumar	**20BCS7572**

Co Supervisor Signature
Shikha Atwal

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CHANDIGARH UNIVERSITY, GHARUAN
June 2022

ABSTRACT

Algorithm trading is all about the use of computers programs to automate one or more stages of the trading process: pre trade analysis (data analysis), trading signal generation (buy and sell recommendations), and trade execution. Today, Algorithmic trading is amongst the most talked about technologies in the recent years. It has given trading Firms more power in the rapidly evolving markets by eliminating human errors and changing the way financial markets are interlinked today. Its usage is credited to most markets and even to commodity trading.

For this project we will use python and machine learning. Machine learning models are becoming increasingly prevalent in algorithmic trading and investment management. The spread of machine learning in finance challenges existing practices of modelling and model use and creates a demand for practical solutions for how to manage the complexity pertaining to these techniques. Python has become a preferred choice for trading recently as Python is open-source and all the packages are free for commercial use. Python has gained traction in the quant finance community. Python makes it easy to build intricate statistical models with ease due to the availability of sufficient scientific libraries.

The process to retrieve, format and use data is an essential part of trading using Python, as without data there is nothing we can go ahead with. Financial data is available on various online websites. This data is also called as time-series data as it is indexed by time (the timescale can be monthly, weekly, daily, 5 minutely, minutely, etc.). Apart from that, we can directly upload data from Excel sheets too which are in CSV format, which stores tabular values and can be imported to other files and codes.

After that at the end we will have to do Backtesting. Backtesting a trading strategy is the process of testing a trading hypothesis/strategy on the historical data. Let's say we formed a hypothesis. This hypothesis states that securities that have positive returns over the past one year are likely to give positive returns over the next one month. By using historical data, we can backtest and see whether our hypothesis is true or not. It helps assess the feasibility of a trading strategy by discovering how it performs on the historical data. If we backtest our strategy on the historical data and it gives good returns, we will be confident to trade using it. If the strategy is performing poorly on the historical data, we will discard or re-evaluate the hypothesis.

TABLE OF CONTENT

1. Introduction

Advances in telecommunications and computer technologies during the past decade have created increasingly global, dynamic, and complex financial markets, which in turn have stimulated trading by computer programs and the rise of systems for algorithmic trading—also known as AT, algo, or black-box- to automate one or more stages of the trading process.

These systems seek to capture fleeting anomalies in market prices, profit from statistical patterns within or across financial markets, optimally execute orders, disguise a trader's intentions, or detect and exploit rivals' strategies. Ultimately, profits drive any algorithmic trading system—whether in the form of cost savings, client commissions, or proprietary trading.

Algorithmic trading and artificial stock markets have generated huge interest not only among brokers and traders in the financial markets but also across various disciplines in the academia. The emergence of algorithmic trading has created a new environment where the classic way of trading requires new approaches. In order to understand the impact of such a trading process on the functioning of the market, new tools, theories and approaches need to be created. Thus, artificial stock markets have emerged as simulation environments to test, understand and model the impact of algorithmic trading, where humans and software agents may compete on the same market. The purpose of this project is to create a framework to test and analyze various trading strategies in a dedicated artificial environment.

Among the hottest programming languages for algo trading, are R and Python, alongside languages such as C++, C#, and Java. In this project, we'll use Python for Algorithmic trading system. The project will cover the following:

- The basics that we need to get started: We'll first know more about the stocks and trading strategies, what time series data is and what we need to do to set up our workspace.
- An introduction to time series data and some of the most common financial analyses, such as moving windows, volatility calculation, … with the Python package Pandas.
- The development of a simple momentum strategy: We'll first go through the development process step-by-step and start by formulating and coding up an algorithmic trading strategy.
- Next, you'll backtest the formulated trading strategy with Pandas, zipline and Quantopian.
- Afterward, we'll see how we can do optimizations to our strategy to make it perform better, and we'll eventually evaluate our strategy's performance and robustness.

2. Literature Review

Algorithmic or Quantitative trading can be defined as the process of designing and developing statistical and mathematical trading strategies. It is an extremely sophisticated area of finance.

Following are the steps to make an algorithmic trading system:

1. Learn Python Programming

In order to start developing an algorithmic trading system, we need solid fundamentals. In Python, we have to thoroughly understand certain topics in the language.

Here's what it takes to master in the Python ecosystem for data science:

- **Environment Setup—**this includes creating a virtual environment, installing required packages, and working with Jupyter notebooks.
- **Data Structures**—some of the most important python data structures are lists, dictionaries, NumPy arrays, tuples, and sets.
- **Object-Oriented Programming—**As a quant analyst, we should make sure we are good at writing well-structured code with proper classes defined. We must learn to use objects and their methods while using external packages like Pandas, NumPy, SciPy, and so on.

2. Learn How to Crunch Financial Data

Data analysis is a crucial part of finance. Besides learning to handle dataframes using Pandas, there are a few specific topics that we have to pay attention to while dealing with trading data.

How to exploring data using Pandas
One of the most important packages in the Python data science stack is undoubtedly Pandas. We can accomplish almost all major tasks using the functions defined in the package.
We have to focus on creating dataframes, filtering (loc, iloc, query), descriptive statistics (summary), join/merge, grouping, and subsetting.

How to deal with time-series data
Trading data is all about time-series analysis. We have to learn to resample or reindex the data to change the frequency of the data, from minutes to hours or from the end of day OHLC data to end of week data.

3. Figuring out how to Write Fundamental Trading Algorithms

A project in quantitative finance requires a solid understanding of statistical hypothesis testing and mathematics. A good grip over concepts like multivariate calculus, linear algebra, probability theory will help us lay a good foundation for designing and writing algorithms.
We can start by calculating moving averages on stock pricing data, writing simple algorithmic strategies like moving average crossover or mean reversion strategy and learning about relative strength trading.
After taking this small yet significant leap of practicing and understanding how basic statistical algorithms work, we can look into the more sophisticated areas of machine learning techniques. These require a deeper understanding of statistics and mathematics.

4. Learn About Backtesting

Once we are done coding our trading strategy, we can't simply put it to the test in the live market with actual capital.
The next step is to expose this strategy to a stream of historical trading data, which would generate trading signals. The carried-out trades would then accrue an associated profit or loss (P&L) and the accumulation of all the trades would give us the total P&L. This is called backtesting.
Backtesting requires you to be well-versed in many areas, like mathematics, statistics, software engineering, and market microstructure. Some concepts we will learn to get a decent understanding of backtesting:

- We can start by understanding technical indicators. Explore the Python package called TA_Lib to use these indicators.
- Employ momentum indicators like parabolic SAR, and try to calculate the transaction cost and slippage.
- Learn to plot cumulative strategy returns and study the overall performance of the strategy.
- A very important concept that affects the performance of the backtest is bias. We should learn about optimization bias, look-ahead bias, psychological tolerance, and survivorship bias.

5. Performance Metrics—How to Evaluate Trading Strategies

It's important for us to be able to explain our strategy concisely. If we don't understand our strategy, chances are on any external modification of regulation or regime shift, our strategy will start behaving abnormally.
Once we understand the strategy confidently, the following performance metrics can help us learn how good or bad the strategy actually is:

- **Sharpe Ratio**—heuristically characterizes the risk/reward ratio of the strategy. It quantifies the return we can accrue for the level of volatility undergone by the equity curve.
- **Volatility**—quantifies the "risk" related to the strategy. The Sharpe ratio also embodies this characteristic. Higher volatility of an underlying asset often leads to higher risk in the equity curve and that results in smaller Sharpe ratios.
- **Maximum Drawdown**—the largest overall peak-to-trough percentage drop on the equity curve of the strategy. Maximum drawdowns are often studied in conjunction with momentum strategies as they suffer from them. We will learn to calculate it using the NumPy library.
- **Capacity/Liquidity**—determines the scalability of the strategy to further capital. Many funds and investment management firms suffer from these capacity issues when strategies increase in capital allocation.
- **CAGR—**measures the average rate of a strategy's growth over a period of time. It is calculated by the formula: (cumulative strategy returns) ^ (252/number of trading days)-1

3. Problem Definition

Algorithmic trading, also referred to as algo-trading, is a variant of automated trading that involves the usage of computerized platforms and advanced math and computer programming tools to drive trading transactions in the financial markets. The system utilizes a mathematical model or algorithm or standardized instruction set that facilitates buying or sell signals in the financial markets and facilitates trade without humans' involvement.

Components of Algorithmic Trading

1. An Algorithm

An algorithm can be defined as instructions that perform certain repetitive functions. It can also be developed to cater to certain problem-solving situations. For example, it helps in the easy facilitation of data processing and identifying trends.

2. Computer Program & Automated Trading Platforms

An automated trading platform provides a means to execute the algorithm developed by the programmers. It, as a platform, manages the computer programs designed by the programmers and algo-traders, thereby facilitating buy and sell orders in the financial markets. These platforms also help in the Backtesting of algorithms developed by the algo-traders or programmers before they can be deployed.

3. Technical Analysis

The technical analysis involves studying and analyzing the price movements of the listed securities in the financial markets. There are several methods, such as 150-day moving average, 200-days moving average, a double exponential moving average, random oscillators, which helps in the identification of price trends for a particular security.

The methods of technical analysis can be developed as an algorithm. They can, in turn, be transformed into a computer program that one can then deploy into the financial markets to automate the trading function.

4. Backtesting

Backtesting is the process of testing the algorithm and verifying whether the strategy would deliver the results anticipated by the trader. It involves testing the programmer's approach on the historical market data. In addition, the Backtesting lets the trader identify the pitfalls that could have emerged if one used the strategy with the live market trades.

4. Objectives

Algo-trading is the fastest evolving technology with incredible and intelligent functionalities that improve trading performance and speed for market participants. Today, financial institutions such as banks, broking houses and investment funds employ sophisticated algorithms to establish and liquidate positions at breakneck speed. This setup enables such institutions to exploit minor discrepancies in live market conditions, eclipsing what's possible manually.

Algo-trading constantly monitors markets and places orders when conditions match a set of parameters such as volume, price, resistance, support, or any other factor that the trader or market participant is comfortable with. One of the advantages of algo-trading is that by using available data, it can easily and quickly identify an ongoing trend. It is otherwise a challenge for market participants to act fast by analyzing a large pool of data in quick time. Besides, market participants can use multiple strategies at once and decide the net outcome of the strategy.

The main objective of algo-trading is not just to profit by trading but to save costs, minimize market impact, and the execution risk of a trading order. Traders don't need to watch stocks or send slices manually.

Algorithmic trading is mainly used by institutional investors and big brokerage houses to cut down on costs associated with trading. According to research, algorithmic trading is especially beneficial for large order sizes that may comprise as much as 10% of overall trading volume. Typically, market makers use algorithmic trades to create liquidity.

Algorithmic trading also allows for faster and easier execution of orders, making it attractive for exchanges. In turn, this means that traders and investors can quickly book profits off small changes in price. The scalping trading strategy commonly employs algorithms because it involves rapid buying and selling of securities at small price increments.

Algorithmic trading can also be used for high frequency trading (HFT) or quant-based trading. In HFT, the objective is to enter and exit frequently and take advantage of daily and intra-day changes. Typically, HFT does not lead to any delivery positions and all transactions are reversed in the same day.

References

1. https://ieeexplore.ieee.org/abstract/document/5696713/authors#authors

2. http://faculty.haas.berkeley.edu/hender/ATInformation.pdf

3. https://www.activestate.com/blog/how-to-build-an-algorithmic-trading-bot/

4. https://www.datacamp.com/community/tutorials/finance-python-trading

5. https://blog.quantinsti.com/python-trading/

6. https://www.freecodecamp.org/news/how-to-get-started-with-algorithmic-trading-in-python/

7. https://www.wallstreetmojo.com/algorithmic-trading/

8. https://economictimes.indiatimes.com/markets/stocks/news/how-algo-trading-makes-it-easier-to-grow-protect-wealth-in-stock-market/articleshow/86701379.cms?from=mdr

9. https://journals.sagepub.com/doi/full/10.1177/2053951720926558

10. https://ideas.repec.org/p/nig/wpaper/0139.html

11. https://vixra.org/pdf/1912.0492v1.pdf