Financial Market Simulation

Table of content

Financial Market Simulation

Overview:

This project covers an overview of the financial market using Monte Carlo simulation, analyzing the system influenced by uncertainty.

Monte Carlo Simulation is a statistical method used to model and analyze systems influenced by uncertainty. It uses repeated random sampling to calculate the probability of different outcomes.

In finance, it's commonly applied in:

- Stock price prediction
- Portfolio risk assessment
- Option pricing

Key Features:

Monte Carlo Simulation:

How does it work in the Financial Market?
Monte Carlo simulations model the future price of assets by assuming that price changes follow a random process. The most popular method is based on Geometric Brownian Motion (GBM) which incorporates:

- **Drift**: The average rate of return of the stock.
- **Volatility**: The degree of variation in stock prices over time.
- **Randomness**: Unpredictable market factors modelled using random numbers.

Mathematical Formula:
The stock price at time t is modelled as:

$$S_t = S_0 e^{(r-\sigma^2/2)t + \sigma\epsilon\sqrt{t}}$$

Where:

- S_0: initial stock price
- r : Expected return
- σ: Volatility
- ϵ: Random number drawn from a standard normal distribution
- t: Times in years

Financial Market Simulation

Markov Chain Simulation:

Markov Chains are used to model systems where the future state depends only on the current state (memoryless property). In the financial market, Markov Chains can model:

- **Price State Transitions**: Modeling how stock prices or market conditions transition between states, such as bullish, bearish, or neutral.
- **Credit Risk Analysis**: Predicting the probability of default or transition between credit ratings over time.
- **Portfolio Optimization**: Understanding market state changes to optimize asset allocation.

Mathematical Representation:

A Markov Chain is represented by a transition matrix, where each element P(i,j) indicates the probability of transitioning from state i to state j. For example:

$$P = \begin{pmatrix} P(B \to B) & P(B \to S) & P(B \to N) \\ P(S \to B) & P(S \to S) & P(S \to N) \\ P(N \to B) & P(N \to S) & P(N \to N) \end{pmatrix}$$

Where:

- $P(B \to B)$ is the probability of staying in a bullish state
- $P(B \to S)$ is the probability of transitioning from bullish to bearish, and so on for other transitions.

The Markov property ensures that the probability of the market's next state only depends on its current state and not its history, which makes it useful for modelling complex financial dynamics with fewer parameters.

Genetic Algorithm Simulation:

Genetic algorithms (GA) are a type of optimization algorithm inspired by the process of natural selection. In financial market simulations, GAs are used to find the best possible strategy or portfolio allocation by mimicking evolutionary processes, such as selection, crossover, and mutation. These algorithms work by generating a population of potential solutions, evaluating them based on a fitness function, and then evolving better solutions over successive generations.

- **Population**: A set of candidate solutions (e.g., portfolio configurations or trading strategies).
- **Selection**: The process of choosing the best solutions from the current population.
- **Crossover**: The combination of two parent solutions to produce offspring solutions.
- **Mutation**: A random modification of a solution to introduce new variations.

Financial Market Simulation

Mathematical Representation:
Let the population at generation t be represented as P_t. Each individual solution $x_i \in P_t$ is evaluated using a fitness function $f(x_i)$. The best solutions are selected to reproduce, and their offspring are evaluated for the next generation.

Folder Structure:

```
project-root/
├── public/
│   ├── css/
│   │   └── styles.css
│   ├── js/
│   │   └── main.js
│   ├── images/
│   │   └── logo.png
│   └── index.html
├── src/
│   ├── routes/
│   │   └── api.js
│   ├── controllers/
│   │   └── simulationController.js
│   ├── simulations/
│   │   ├── monte_carlo.py
│   │   ├── markov_chain.py
│   │   └── genetic_algorithm.py
│   └── app.js
├── .env
├── package.json
├── package-lock.json
├── README.md
└── requirements.txt
```