Distributed Financial Risk Assessment

Anirudh Garg ag9563@nyu.edu
Nikhil Kommineni nk3853@nyu.edu
Sai Preetham sb9509@nyu.edu
Sujana Maithili sc10648@nyu.edu

1 Abstract

This project aims to implement a distributed framework for financial risk assessment using Monte Carlo simulations to estimate Value at Risk (VaR) and Conditional Value at Risk (CVaR). By leveraging Spark's [1] distributed computing capabilities, we efficiently parallelize processing of large-scale market data, simulating millions of scenarios to generate robust risk metrics. The framework also incorporates techniques to optimize computational performance and ensures scalability for real-world financial datasets, enabling timely and reliable risk evaluation for decision-making.

All computations are run on the NYU dataproc cluster hosted on GCP.

2 Introduction

Financial markets are characterized by their complexity and volatility, making accurate risk assessment critical. Traditional risk models often lack scalability and fail to adapt in real time to dynamic market conditions. To address these limitations, this work proposes a distributed framework that leverages Spark's parallel processing capabilities to analyze large-scale financial data efficiently. By improving the precision of risk metrics like VaR and CVaR, our system provides actionable insights for portfolio management and strategic decision-making, empowering stakeholders to navigate uncertainties with confidence.

Monte Carlo simulation forms the backbone of our approach, estimating risk by simulating millions of random market scenarios. This method captures the probabilistic behavior of portfolios under diverse conditions, addressing the limitations of traditional variance-covariance and historical simulation methods.

Financial analysts can use the system to assess portfolio risk and generate actionable insights for crafting investment strategies. Portfolio managers can leverage the calculated VaR and CVaR metrics to optimize asset allocation and ensure risk-adjusted returns. Additionally, risk management teams can utilize the framework to estimate necessary capital reserves and design effective risk mitigation strategies, thereby enhancing overall financial stability and resilience.

2.1 Value at Risk

Value at Risk(VaR) is a measure of investment risk that provides a reasonable estimate of the maximum probable loss in value of an investment portfolio over a particular period of time. A VaR of $1 million with a 5% probability and two weeks reasonably justifies that the portfolio stands only a 5% chance that it losses more than $1 million dollars over two weeks.

2.2 Conditional VaR

Conditional Value at Risk (CVaR), sometimes known as expected shortfall, is a related investment risk metric that was recently proposed as a better alternative to VaR. CVaR uses the expected loss instead of a cut-off value to indicate the risk in a portfolio. A CVaR of $1 million with a 5% $q-value$ and two weeks indicates that the average loss in the worst 5% of outcomes is $1 million.

3 Background

3.1 Terminology

We make use of the following specific terms related to the financial domain in this report:

- *Index*: An imaginary portfolio, often built to model the market trends. For example, the NASDAQ Composite Index includes about 3,000 stocks and similar instruments listed in the New York Stock Exchange.
- *Market factor*: A value that can be used as an indicator of macroaspects of the financial climate at a particular time. Common examples include, the value of an index, the GDP of the US, or the exchange rate between the dollar and the euro.

3.2 Methods for Calculating VaR

Estimating VaR requires us to propose a model for how the portfolio functions and choosing the probability distribution its returns are likely to take. Institutions employ a variety of approaches for calculating VaR, all of which tend to fall under a few general methods.

- **Variance-Covariance**: The Variance-Covariance method assumes that each instruments returns are normally distributed, which allows deriving a closed-form solutions to the estimated losses. This is a low computationally expensive method.
- **Historical Simulation**: Historical simulation extrapolates risk from historical data by using its distribution instead of relying on summary statistics. For example, to determine a 95% VaR of a portfolio, we could look at that portfolio's performance for the last 100 days and estimate the statistic as its value on the fifth-worst day. This method is limited by the historical data under consideration and is not immune to events like market crashes, which are important for risk calculations.
- **Monte Carlo Simulations**: Monte Carlo Simulation simulates the portfolio performance under random conditions. Such a simulation technique is often used when we can't derive a closed form solution for the probability density function. The density function can still be estimated by repeatedly sampling simpler random variables that it depends on and seeing how it plays out in aggregate. A typical outline looks like:
  - Defines a relationship between market factor data and the instrument values by mining historical market data.
  - Defines easy-to-sample-from probability distributions for various market factors we aim to use.
  - Runs trials consisting of random market conditions.
  - Computes the portfolio returns and the corresponding losses after each trial. We then build an empirical distribution over the outputs. If we run 100 trials and want to estimate the 5% VaR, we would choose it as the loss from the trial with the fifth-greatest loss. To calculate the 5% CVaR, we would find the average loss over the five worst trials.

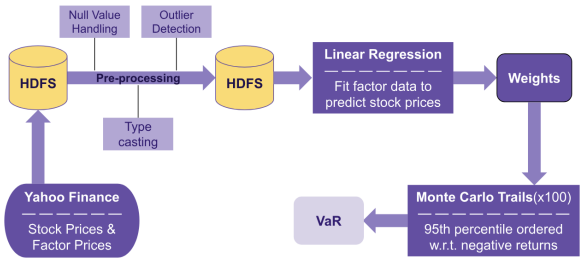

Figure 1: Data pipeline

4 Proposed Method

The proposed method for estimating financial risk using Monte Carlo simulation consists of the following stages[1]:

4.1 Data Collection and Cleaning

The historical financial data is collected using Python's `yfinance` [2] library. The data includes the following columns: Date, Price, Adjusted Close, Close, High, Low, Open, and Volume. After downloading the data using the list of stock ticker names obtained from each country's stock exchange websites, the first three rows contain metadata, column names, and some poorly formatted rows in the second and third rows. To handle this, we filter out the first three rows and read the remaining data into a DataFrame with the specified columns. Since we are only interested in the Price and Close columns, we select these two columns for further processing in our code.

Table 1: Sample Stock Data

Date	Price	AdjClose	Close	High	Low	Open	Volume
2000-01-04	0.247	1.27	1.27	1.11	1.11	1.26	17828000
2000-01-05	0.238	1.22	1.27	1.10	1.10	1.22	20038000
2000-01-06	0.249	1.28	1.47	1.22	1.26	1.47	38700000

The collected data is cleaned to handle missing values and inconsistencies using techniques such as forward and backward filling. We also take the start and end dates into consideration. For our use case, we filter out any stock data that is not listed before the start date and only retain data that is available prior to that date. Since we are using data from different countries and comparing it against U.S. indices and bond yields, there may be cases where certain values are missing due to holidays in specific countries. To address this, we fill in the missing values with the nearest available data from the closest date.

Table 2: Sample Index Data

Date	Price	Adj Close	Close	High	Low	Open
2000-01-03	6.457	6.457	6.473	6.411	6.411	6.457
2000-01-04	6.396	6.396	6.450	6.388	6.434	6.450
2000-01-05	6.489	6.489	6.489	6.396	6.427	6.489

4.2 Determining Factor Weights

VaR is concerned with losses over a specific time horizon, not the absolute prices of instruments, but rather how those prices change over a given period. In our calculation, we will define this period as two weeks. We begin by calculating the 2-week returns of stocks and factors, and then we use linear regression [3] to model the stock return data based on the factor return data.

To account for the possibility that instrument returns may be nonlinear functions of factor returns, we can introduce additional features derived from nonlinear transformations of the factor returns. Specifically, we will add two new features for each factor return: the square and the square root. Despite these transformations, our model remains linear in nature, as the response variable is still a linear function of the features.

Although we will perform multiple regressions, one for each instrument, the number of features and data points in each regression is small, so we don't need to utilize Spark's distributed linear modeling capabilities.

Model Representation: The return of an instrument i in trial t is given by:

$$r_{i,t} = c_i + \sum_{j=1}^{n} w_{i,j} f_{t,j}$$

where:

- c_i is the intercept term,
- $w_{i,j}$ are the regression weights,
- $f_{t,j}$ are the randomly generated transformed factor values in trial t.

Feature Transformation: Nonlinear transformations are applied to factor returns, including squared and square root terms:

$$\text{Features} = \{f, f^2, \sqrt{|f|}\}.$$

The regression parameters are estimated using Ordinary Least Squares:

$$\mathbf{W} = (\mathbf{x}^\top \mathbf{x})^{-1} \mathbf{x}^\top \mathbf{y},$$

where $\mathbf{W}$ is the weight matrix, $\mathbf{x}$ is the vector of factor features and $\mathbf{y}$ is the vector of instrument returns.

4.3 Sampling

To simulate market conditions, we need to generate random factor returns by selecting a probability distribution over factor return vectors and sampling from it. A helpful tool for this is the multivariate normal distribution, which accounts for the correlations between factors. By using this distribution, each sample is a vector, where the distribution of values along each dimension is normal, given the values for all other dimensions. However, the variables are not independent in their joint distribution, making the multivariate normal ideal for capturing these dependencies.

The distribution is parameterized by a mean for each dimension and a covariance matrix that describes the relationships between each pair of dimensions. To start, we will fit a normal distribution to each factor's returns, but we will also consider the correlations between them. While more complex distributions could provide a better fit, for simplicity, we will focus on the multivariate normal. This approach ensures that

we account for the inter-dependencies between factors, which is crucial for accurately assessing risk.

Multivariate Normal Distribution: Factor returns $\mathbf{m}_t$ are sampled as:

$$\mathbf{m}_t \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

where:

- $\boldsymbol{\mu}$ is the vector of mean returns, and
- $\boldsymbol{\Sigma}$ is the covariance matrix of factor returns.

Covariance Matrix: The covariance matrix is computed as:

$$\Sigma_{ij} = \frac{1}{n-1} \sum_{k=1}^{n} \big(x_{k,i} - \mu_i\big)\big(x_{k,j} - \mu_j\big),$$

where:

- $x_{k,i}$ is the i-th factor's return in the k-th sample,
- μ_i is the mean return for the i-th factor, and
- n is the number of samples.

4.4 Running Trials

To parallelize the computationally intensive trials, we will use Spark. Each trial involves sampling a set of risk factors, predicting the returns for each instrument, and summing them to calculate the trial loss. We can parallelize the simulation along trials, instruments, or both, but we will focus on parallelizing by trials. We will run a total of 1,00,000 trials. We calculate the full return for each trial by averaging the returns of all instruments, assuming equal portfolio weights. For unequal holdings, a weighted average would be used which we used to calculate to find the VaR of famous investor portfolios.

Instrument Return Calculation: Using the pretrained regression model, the return for each instrument is:

$$r_i = c_i + \sum_{j=1}^{n} w_{i,j} f_j,$$

where f_j are the features derived from the sampled factor returns.

Portfolio Return: The total portfolio return for a trial is the sum of instrument returns divided by total no of instruments assuming that all the stocks are of equal weight:

$$L = \frac{1}{m} \sum_{i=1}^{m} r_i$$

where m is the number of instruments in the portfolio. From this we calculate the VaR and CVaR values.

Value at Risk (VaR): To calculate the 5% VaR, we find the return that is expected to be underperformed 5% of the time and outperformed 95% of the time. This can be done by taking the worst 5% of trials and then selecting the return of the best trial in this subset as the VaR.

Conditional Value at Risk (CVaR): Instead of selecting the best return from the worst 5% of trials, we compute the average return from this subset of trials.

5 Results

We evaluated the framework on datasets from major stock exchanges such as NYSE (US Stock Exchange), NSE (India Stock Exchange), HKSE (Hongkong Stock Exchange), and LSE (London Stock Exchange).

Table 3: Stock Exchange Data Overview

Exchange	Size	Symbols	Time Period
NYSE	1.1 GB	2800	2000 to 2024
NSE	870 MB	2021	2000 to 2024
HKSE	693 MB	1800	2000 to 2024
LSE	601 MB	1720	2000 to 2024

Our market factors include S&P 500, NASDAQ Composite, YYX (Treasury Yield 30-years), and FVX (Treasury Yield 5-years).

Table 4: Market Factor Data Overview

Index	Size	Time Period
S&P 500	660KB	2000 to 2024
NASDAQ	640KB	2000 to 2024
YYX	636KB	2000 to 2024
FVX	644KB	2000 to 2024

The following results summarize the computed risk metrics:

Table 5: VaR and CVaR for Stock Exchanges

Exchange	VaR	CVaR
NYSE	2.73%	4.35%
NSE	-4.2%	1.83%
HKSE	2.11%	2.14%
LSE	-3.73%	1.66%

The above results indicate that NSE (Indian Stock Exchange) is least risky in the short-medium term future. This could be due to India's growing economic stability and strong domestic demand create a favorable environment, making the NSE less susceptible to global market fluctuations. Moreover, India's large and growing domestic market, coupled with a relatively young and dynamic workforce, provides a strong foundation for economic stability. This intrinsic resilience of the Indian economy helps cushion the NSE against external shocks, contributing to its lower perceived risk in the short-to-medium term.

We also estimate VaR of famous investors like Warren Buffett, Bill & Melinda Gates, and George Soros, this provides insights into the risk profiles of portfolios. For our estimate we considered all investments that are $\geq$1%. Below table shows few of their investments.

Table 6: Top 10 Investments of Famous Investors

Warren Buffett	Bill & Melinda Gates	George Soros
AAPL (26.24%)	MSFT (27.64%)	SPY (22.32%)
AXP (15.44%)	BRK.B (22.6%)	SW (6.25%)
BAC (11.88%)	WM (14.84%)	AZN (3.79%)
KO (10.79%)	CNI (14.25%)	IWM (3.24%)
CVX (6.56%)	CAT (6.38%)	AER (2.76%)
OXY (4.94%)	DE (3.29%)	BABA (2.54%)
MCO (4.40%)	ECL (2.95%)	XLF (2.33%)
KHC (4.29%)	WMT (1.63%)	GFL (2.31%)
CB (2.93%)	FDX (1.54%)	IVV (2.27%)
DVA (2.22%)	KOF (1.22%)	HYG (2.21%)

We achieve the following VaR values. We observe George Soros's portfolio, has the lowest VaR of 0.84%, which reflects a better risk management strategy through diversification. It is interesting to note that both Warren Buffet and Bill & Melinda Gates, with significant investments in the technology sector, have much higher computed VaR scores than George Soros with most of his investments in traditional industries. It can also be noted that both Warren Buffet and Bill & Melinda Gates are playing a more risky investment game than George Soros, and are likely to aim for higher profit margins to justify the higher value at risk in their portfolios.

Table 7: VaR of Famous Investors Portfolios

Investor	VaR
Warren Buffett	4.92%
Bill & Melinda Gates	5.40%
George Soros	0.84%

Additionally, we also estimate the VaR of companies based on their time of listing. Companies listed prior to 2000 exhibit lower risk metrics, with a VaR of 2.73% and CVaR of 5.57%, suggesting greater stability in established firms. In contrast, newer companies listed in the last 4 years show a higher VaR of 5.27% and CVaR of 9.20%, reflecting their higher volatility and exposure to market uncertainties. This comparison highlights the impact of firm maturity wrt risk exposure. Older companies benefit from established market presence and consistent performance, whereas, newer firms, still in their growth phase, face greater uncertainty due to evolving business models and limited historical data.

Goodness: The evaluation of the model's goodness is based on two key aspects: internal consistency and how well the model matches reality.

- **Is the model internally consistent?** Bootstrapping allowed us to obtain confidence intervals for the Value at Risk (VaR) by repeatedly sampling with replacement from the set of portfolio return results of our trials. For example, for the HKSE, we can confidently say that VaR falls between [0.021055, 0.021272].

- **How well does our model match reality?** While the confidence interval gives us a sense of statistical consistency, it does not provide enough information to assess how well the model matches real-world performance. To evaluate this, Kupiec's proportion-of-failures (POF) test is used, which analyzes how the portfolio performed across various historical time intervals and counts the number of times the losses exceeded the VaR.

  We calculate the p-value from the Chi-Squared test, and for the HKSE, the p-value is $7.665867940431781 \times 10^{-12}$, which indicates that the current VaR model does not accurately reflect the real-world. This discrepancy can be

attributed to several factors. First, the factor selection may not adequately capture all the relevant market influences on the portfolio. Additionally, the model assumes linear relationships between market factors and asset returns, which may not be appropriate. Financial markets often exhibit non-linear relationships, where small changes in factors can lead to disproportionate effects on asset returns, especially under extreme conditions. Finally, the assumption of normality of returns may not hold in financial markets, as financial data tends to exhibit more extreme outcomes than a normal distribution would predict.

6 Conclusion

In this project, we estimate Value at Risk (VaR) and Conditional Value at Risk (CVaR), this is inherently challenging due to the complexity and unpredictability of real-world markets. In this project, we leveraged the power of Spark to run hundreds of thousands of trials in parallel, enabling us to simulate a wide range of market scenarios efficiently.

Spark's scalability and distributed computing capabilities were crucial in running these simulations and processing vast amounts of data, providing valuable insights into portfolio risk exposure. While our framework offers a simplified version of risk estimation, it captures essential market dynamics and helps in understanding the potential outcomes under various conditions.

In future we could enhance the model by incorporating additional risk factors, exploring non-linear relationships that account for extreme events. These improvements would help further align the model with the complexities of real-world financial markets, making it more accurate and reliable for risk assessment.

References

[1] Tathagata Das Ankur Dave Justin Ma Murphy McCauley Michael J. Franklin Scott Shenker Ion Stoica Matei Zaharia, Mosharaf Chowdhury. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. *NSDI'12: Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, 2012.

[2] Licensed Apache Software License (Apache) Ran Aroussi. Python yfinance package. *Pypi*, 2024.

[3] Burak Yavuz Evan Sparks Shivaram Venkataraman Davies Liu Xiangrui Meng, Joseph Bradley. https://www.jmlr.org/papers/volume17/15-237/15-237.pdf. *Journal of Machine Learning Research 17 (2016)*, 2016.