Creating Software for Value at Risk

Interim Report

BSc Final Year Project

Author
Benjamin Shearlock

Supervisor
Dr. Volodya Vovk

Department of Computer Science
Royal Holloway, University of London

Declaration

This is my own work etc etc

- word count
- my name
- submission date
- signature
- etc

Contents

1 Abstract

In the realm of financial risk management, understanding and evaluating the level of risk associated with any investment or portfolio is of extremely high importance. Perhaps the most universally regarded metric used for this purpose is Value at Risk (VaR). VaR provides a quantitative estimate of the potential losses that a portfolio may incur over a certain period of time (specified time horizon) at a given confidence level.

Original widespread use of VaR came about in the early 1990s, the concept first being introduced by J.P. Morgan in 1994, since it helped provide an estimate of the maximum loss an investor is willing to accept for any given investment. Its historical roots can be traced back to the financial industry's increasing need for a standardized and comprehensive measure of risk following the 1987 stock market crash, so they sought a comprehensive way to assess risk in complex portfolios [3].

Mathematically, VaR is expressed as follows:

$$VaR(N, X) = -\text{Percentile}(L, 1 - X) \tag{1}$$

Where:

$$\begin{aligned} N &: \text{Time horizon (in days)} \\ X &: \text{Confidence level (in percentage)} \\ L &: \text{Loss distribution over } N \text{ days} \end{aligned}$$

This formula captures the loss at the $(100-X)$th percentile of the loss distribution over the specified time horizon [6].

VaR can be mathematically computed through various methods, each with its own strengths and limitations. The most common approaches include the historical simulation method, parametric method, model building method and Monte Carlo simulation [1], to list a few. For historical simulation, past data is used to estimate future risk by examining the historical returns of an asset or portfolio, while model building employs mathematical models to predict portfolio performance. The choice of algorithm depends on data availability, computational resources, and specific requirements.

To implement VaR calculations, various pieces of software are essential. In this project, I will be utilising Visual Studio Code (VSCode) as the integrated development environment (IDE) and Python for its rich ecosystem of libraries. Python libraries like NumPy, Matplotlib, and Kivy will be invaluable for data manipulation, visualisation, and user interface design [9].

My inaugural proof of concept program will be created to visually demonstrate VaR calculations for two initial methods, these being historical simulation and model building techniques to estimate VaR for a sample portfolio. Historical simulation would involve collecting historical data and computing VaR based on past performance (can involve acquiring stock data through an API), while model building would use a predefined model to forecast future losses. Visualisation tools like Matplotlib can help in presenting the results graphically, enabling me to generate informative charts and graphs. NumPy will facilitate data manipulation and efficient mathematical operations [10]. Additionally, Kivy, a Python framework for developing multi-touch applications, will be used to

create the interfaces for the visual representation of VaR, as well as giving it the option to possibly be viewed on other devices.

Later on into the project, if there is enough time, I think it may be worth exploring some more advanced topics like the variance-covariance of returns, specifically employing GARCH (Generalized Autoregressive Conditional Heteroscedasticity) models. GARCH models provide a more nuanced understanding of volatility and can enhance the accuracy of VaR estimates [6].

The objective of my project is to gain a deeper understanding of VaR, develop a functional program to calculate and visualise it, and potentially extend the research to incorporate more advanced risk management techniques if possible. This project is a stepping stone towards a deeper understanding of the risk side of finances and will contribute to enhancing the knowledge and skills necessary for effective financial decision-making, since this is not a topic that I have delved much into before, but I have aways been very interested in learning more about it. This gives me a fantastic opportunity to learn about the financial sector, as well as create something that is applicable and useful to real life.

2 Specification

3 Chapter 1 - Introduction

3.1 What is VaR?

3.2 Aims and Goals

3.3 Milestone Plan

Due to unfortunate circumstances, I've had rough delays to the start of this Project, so Weeks 1-3 will be noted as being a vague learning of the fundamentals behind Value at Risk. I will express this and what I hopefully plan to accomplish with the Project in the timeline below. In the first term, I want to research and create a working program to compute Value at Risk for small portfolios, that has a serviceable GUI that can be expanded on later. I will also make sure to have amply researched about back-testing and how to incorporate it into my program in some capacity. For the second term, I will research and implement applying Value at Risk for a portfolio of derivatives, as well as looking into using the Monte Carlo simulation and allowing for the computation of all this with as many stocks as necessary. I will finalise the GUI and plan to look into completing some of the extensions provided for the project, however this will depend on the overall developmental scope of the project at the time, so they will not be specified here.

Weeks 1-3

Project Research

- Research the fundamentals of Value at Risk (VaR)
- Research best coding language to use (Python)
- Familiarize myself with LaTeX and prepare IDE & Git for Project specified use

Week 4

Finalize Plan and Start Coding

- Complete Project Plan
- Continue researching VaR and Python
- Begin project coding

Week 5-7

Coding and Data Preparation

- Continue to work on the VaR program (No GUI)
- Start collecting and organizing sample data for small portfolios so it can be used by the program
- Finalizing understanding of the two computational methods needed, this being model-building and historical simulation

Week 8

Back-Testing Research & Implementation

- Investigate methods and techniques for VaR back-testing
- Start integrating back-testing into the project

Week 9-10

GUI Development

- Initiate the development of the GUI
- Ensure the GUI is robust for its current task as well as expandable for future enhancements

Week 11

Interim Report and Presentation Preparation

- Fine-tune programs and report so they are at a satisfactory level, will also allow for easier preparation for the interim presentation
- Prepare for the interim presentation

Weeks 1-2

Reflection and Research

- Spend time to reflect on the progress of the project so far, make any changes that I think are warranted after having the winter break time to think about
- Research the Monte Carlo simulation method for VaR, as well as how I could start implementing derivatives as portfolios into the project

Week 3-4

Start Implementing New Features

- Start implementing the Monte Carlo simulation method and continue derivative implementation
- Start researching Eigen & Cholesky decomposition to allow for however many stocks are needed within a portfolio

Week 5-7

GUI Finalisation

- Decide on the final visual product I want to represent with the GUI and start implementing it (if progress on this needs to continue into the next period, then it will be done so)
- Set the program up to work portably/allowing it to work on mobile OS's as well as different desktop OS's

Week 8-9

Extend Project Scope (if time permits)

- Explore additional features or enhancements for the project, possibly decided upon at the start of Term 2
- Implement as many as can be appropriately managed, with all additional time spent within this period being used to ensure the project is at its most refined state

Week 10-11

Perfect Final Report

- Make sure the program has been achieved to the best of its ability
- Finalise and perfect the final report

4 Chapter 2 - Understanding VaR

4.1 Historical Simulation

4.2 Model Building (Variance-Covariance)

4.3 Coding VaR Methods

4.4 Back Testing

5 Chapter 3 - Graphical User Interface

5.1 What is Kivy?

5.2 Conceptualisation of Initial Design Ideas

5.3 Initial GUI Creation

5.4 Testing Current Implementation

6 Chapter 4 - Evaluation

6.1 Current Work

6.2 Future Work

7 Bibliography

1. Alexander, C. (2008). *Market Risk Analysis: Value-at-Risk Models.* Hoboken, NJ: Wiley. [Online] Available at: `https://ebookcentral-proquest-com.ezproxy01.rhul.ac.uk/lib/rhul/reader.action?docID=416450`.
   This book covers various aspects that I want to approach in this project, such as historical simulation, Monte Carlo simulation and various forms of testing that could be highly useful for my project.
2. Arbuckle, D. (2017). *Daniel Arbuckle's Mastering Python: Build Powerful Python Applications.* Birmingham, England: Packt. [Online] Available at: `https://learning.oreilly.com/library/view/daniel-arbuckles-mastering/9781787283695/?ar=`.
   I chose this reference as it helps with python packaging, being able to turn a python code and all its GUI and libraries into a single executable file, which is something I want to be able to do for this project.
3. Choudhry, M. (2006). *An Introduction to Value-at-Risk.* Chichester: John Wiley & Sons Limited. [Online] Available at: `https://learning.oreilly.com/library/view/an-introduction-to/9780470017579/`.
   Covers similar topics to the first reference, but seems to go into more detail on specific issues that I may need to address in this project.
4. Duffie, D. and Pan, J. (2019). *An Overview of Value at Risk.* [Online] Available at: `http://web.mit.edu/people/junpan/ddjp.pdf`.
   A much shorter reference, it is a paper that covers the basics of Value at Risk, which I will be using to help me understand the fundamentals of the topic since I have been able to follow it better then the other references.

5. Föllmer, H. and Schied, A. (2016). *Stochastic Finance: An Introduction in Discrete Time.* Berlin: de Gruyter. [PDF]
A recommended reference from my supervisor, it is a book that covers higher level concepts relating to my project but also provides an overview of stochastic finance in general, which I have seen to be useful in understanding the topic.

6. Hull, J.C. (2008). *Options, Futures, and Other Derivatives.* Upper Saddle River, NJ: Prentice Hall. [PDF]
This is my main reference for the project, as it is the main textbook suggested. It has a relevant chapter on Value at Risk, which explores many of the different aspects that I will need to look into for this project.

7. Pritsker, M. (1997). "Evaluating Value at Risk Methodologies: Accuracy versus Computational Time." *Journal of Financial Services Research* 12: 201-242. [Online] Available at: `https://doi.org/10.1023/A:1007978820465`.
Another short reference, this is a paper that compares the accuracy and computational time of various Value at Risk methodologies, which I will be using to help me understand the different methodologies and how I can apply them to my project in the most efficient manner.

8. Raman, K. (2015). *Mastering Python Data Visualization: Generate Effective Results in a Variety of Visually Appealing Charts Using the Plotting Packages in Python.* Birmingham: Packt Publishing Limited. [Online] Available at: `https://learning.oreilly.com/library/view/mastering-python-data/9781783988327/?ar=`.
This reference covers various aspects of data visualisation, which I will be using to help me understand how to best display important information in a visually appealing way for my project.

9. Ulloa, R. (2015). *Kivy - Interactive Applications and Games in Python - Second Edition.* Packt Publishing. [Online] Available at: `https://learning.oreilly.com/library/view/kivy-interactive/9781785286926/?ar=`.
This reference is not being used for its content on game development, rather Kivy is a framework that can help create a GUI that works on both desktop operating systems as well as mobile operating systems, which I think I would like to explore the possibility of when developing the application

10. Weiming, J.M. (2015). *Mastering Python for Finance: Understand, Design, and Implement State-of-the-Art Mathematical and Statistical Applications Used in Finance with Python.* Birmingham, England: Packt Publishing. [Online] Available at: `https://learning.oreilly.com/library/view/mastering-python-for/9781784394516/`.
This reference covers various ways of handling financial data within python, which I will ensure to help me understand how to best handle the data I will be using within my project.

8 Planning and Timescale

9 Appendix - Diary