Computational Finance with MATLAB

Ivaylo Krumov
University of Luxembourg
Email: ivaylo.krumov.001@student.uni.lu

This report has been produced under the supervision of:
Giacomo di Tollo
University of Luxembourg
Email: giacomo.ditollo@ext.uni.lu

Abstract

This document is the final report at the end of Phase 2 of the project carried out by Ivaylo Krumov under the guidance of Giacomo di Tollo during his Bachelor Semester Project 5. It addresses the application of numerical methods and software in the optimization of computational finance problems. The report describes the scientific aspects of the project, consisting of the acquired knowledge relevant to the scientific question at hand and to the development of the technical deliverable, and the technical aspects, involving the application of this knowledge to solve fundamental portfolio optimization problems through the use of computational software. Ultimately, the project aims to provide valuable insight into the practicality of applying this software in financial scenarios in the real world.

1. Plagiarism statement

I declare that I am aware of the following facts:

- As a student at the University of Luxembourg I must respect the rules of intellectual honesty, in particular not to resort to plagiarism, fraud or any other method that is illegal or contrary to scientific integrity.
- My report will be checked for plagiarism and if the plagiarism check is positive, an internal procedure will be started by my tutor. I am advised to request a precheck by my tutor to avoid any issue.
- As declared in the assessment procedure of the University of Luxembourg, plagiarism is committed whenever the source of information used in an assignment, research report, paper or otherwise published/circulated piece of work is not properly acknowledged. In other words, plagiarism is the passing off as one's own the words, ideas or work of another person, without attribution to the author. The omission of such proper acknowledgement amounts to claiming authorship for the work of another person. Plagiarism is committed regardless of the language of the original work used. Plagiarism can be deliberate or accidental. Instances of plagiarism include, but are not limited to:
  1) Not putting quotation marks around a quote from another person's work
  2) Pretending to paraphrase while in fact quoting
  3) Citing incorrectly or incompletely
  4) Failing to cite the source of a quoted or paraphrased work
  5) Copying/reproducing sections of another person's work without acknowledging the source
  6) Paraphrasing another person's work without acknowledging the source
  7) Having another person write/author a work for oneself and submitting/publishing it (with permission, with or without compensation) in one's own name ('ghost-writing')
  8) Using another person's unpublished work without attribution and permission ('stealing')
  9) Presenting a piece of work as one's own that contains a high proportion of quoted/copied or paraphrased text (images, graphs, etc.), even if adequately referenced

  Auto- or self-plagiarism, that is the reproduction of (portions of a) text previously written by the author without citing that text, i.e. passing previously authored text as new, may be regarded as fraud if deemed sufficiently severe.

2. Introduction

The vast field of finance and economics is dynamic and complex and, from an investor's point of view, it requires a great deal of understanding to make well-informed decisions to achieve profitable investment results. This is especially true when it comes to managing a portfolio, where even seemingly small actions can have a large impact on its overall value. In these scenarios the optimal balance between risk and return is always a priority objective.

Nowadays, financial markets have evolved to be more complex and unpredictable than ever before. For this reason, the sheer utility that various computational tools can bring is particularly valuable to investors, as they can provide tremendous assistance for optimal asset allocation or risk assessment in a portfolio. In particular, MATLAB stands out as a useful tool incorporating a variety of modules that should allow an investor to effectively deal with problems related to portfolio optimization.

This report will explore the extent to which MATLAB is suitable for portfolio optimization by aiming to answer the fundamental scientific question "Can we reliably use MATLAB to help optimize investment choices?". Answering this question will involve a thorough research of the key theoretical scientific and mathematical concepts of financial optimization and risk management, as well as basic understanding of the integrated utilities of MATLAB that offer relevant ways to optimize asset investments in a portfolio. Real-world applications of these findings will be especially emphasized, since ultimately this report, and more generally the project as a whole, is expected to serve as a comprehensive way to provide practical insight for investors looking to maximize their portfolio profits in realistic scenarios. By the end of this exploration readers will have the necessary theoretical and practical knowledge to solve fundamental portfolio optimization problems with MATLAB, in preparation for more advanced applications in numerous investment situations.

3. Numerical Methods in Finance and Economics

To be able to give an answer to our scientific question, namely "Can we reliably use MATLAB to help optimize investment choices?", we will need to gain a basic understanding of the theoretical foundations of optimization techniques in the field of finance and economics. To that end, we will first explore the concept of numerical methods, as they are essential in solving optimization problems efficiently. Numerical methods are mathematical tools commonly used in finance and economics to solve complex problems by approximating solutions through computations instead of precisely obtaining analytical solutions. Approximations are frequently favored over true optimal solutions because financial markets are dynamic and complex. As such, real-time decision-making may not be feasible in situations where true optima require heavy computations. Furthermore, precise answers tend to not be very useful since financial models include assumptions and simplifications that might not account for all market nuances. To put it simply, approximated solutions can be computed considerably more quickly and can be more resistant to even the smallest changes in the data or the model. Moreover, this approach is particularly powerful in the field of finance and economics due to several other factors which make it quite difficult to find precise solutions such as the sheer volume of the used financial data and the need to optimize multiple parameters at once.

In portfolio optimization problems, numerical optimization algorithms, ranging from basic ones like mean variance optimization and value at risk (VaR) or more sophisticated methods such as conditional value at risk (CVaR), are used to create suitable financial models and construct diversified investment portfolios that offer the best possible balance between risk and return. Furthermore, numerical methods provide crucial assistance when it comes to risk assessment and management. Common risk measures such as the aforementioned VaR and CVaR, which estimate the maximum potential loss with a given probability over a specified time horizon, heavily depend on numerical computations. By evaluating portfolios' downside risk using statistical analysis and historical market data, these techniques help investors protect themselves from unfavorable market moves and prevent or at least mitigate possible losses.

To better understand the role of numerical methods in finance and economics, and more specifically in portfolio optimization problems, we will take a closer look at the key concepts in the portfolio optimization process.

3.1. Portfolio selection

Portfolio selection can be defined as the procedure of strategically distributing investments among a range of assets based on a variety of factors, such as historical data, risk assessments, and investor preferences. The goal of this is to minimize losses and maximize profits while taking into account the volatility of financial markets and the inherent uncertainty of investing in assets.

In particular, modeling uncertainty is an essential aspect of portfolio selection. However, to be able to create a suitable model, we need to first understand the concept of uncertainty itself. This is where knowledge of probability and statistics comes into play and a good explanation is provided by [Paolo Brandimarte (2013)]. Depending on the impact of investor actions on the market, uncertainty can be defined as exogenous (low impact) or endogenous (high impact). Exogenous uncertainty is applicable for small investors or in markets with a large asset supply. Meanwhile, uncertainty is endogenous in thinner markets where trading an asset can have a noticeable impact on its price. For such small markets large trades can significantly affect asset prices, therefore requiring strategies that take this into consideration. Another way of classifying uncertainty is by objective and subjective descriptions. An objective view assumes a true representation derived from historical data and statistical analysis, whereas subjective uncertainty is generated by investors who have particular interests that result in skewed uncertainty evaluations, further changing whenever new data becomes available.

Gaining insight about the impact of large price changes on markets can be particularly beneficial for portfolio selection. This is something that is addressed by [Joseph Andria, Giacomo di Tollo, Jaan Kalda (2022)] in-depth, discussing the statistical properties of financial time series when it comes to significant price fluctuations by putting an emphasis on how large market movements can greatly impact the prices of assets. The reliability of portfolio models is affected by their level of adaptability to such substantial market changes that can impact uncertainty, so understanding the dynamic nature of markets is a crucial aspect of the process of selecting a truly reliable portfolio.

Having this general understanding of uncertainty, we can continue with an example of how it can be modeled for the price of an asset, as described by [Paolo Brandimarte (2013)]. A variety of models exist for this purpose, from basic ones like the binomial model, which has two potential future asset prices (either an increase or decrease in value, where an increase can be defined as p and a decrease as $1 - p$ or vice versa), to complex ones that allow for several future states. These types of models are known as discrete-state models, as they utilize discrete probability distributions to represent uncertainty in relevant state variables, such as interest rates. Bushy tree models in general balance computational complexity and representation accuracy by representing uncertainty over discrete time periods and are suitable for strategies involving *buy and hold*, where assets are traded in the present and the outcome is awaited at some point in the future.

There are situations where uncertainty is better represented using continuous distributions (normal or lognormal). This approach makes modeling uncertainty simpler and may make analytical formulations easier. Comparably, differential equations representing deterministic processes are frequently used in continuous-time models, which are an improvement over discrete-time models as time steps approach zero. However, these equations need to include random elements to account for uncertainty, which are typically represented by stochastic processes like the Wiener process.

An essential aspect of portfolio selection is making sure that assets can be optimally allocated. As explained by [Paolo Brandimarte (2013)], the mean-variance portfolio optimization method is a common approach that explores the distribution of wealth among different risky assets, each with its own standard deviation and expected return. Generally, it may make sense to select assets with high expected returns and low risks but ignoring their correlations can lead to the loss of possible gains, particularly in cases when returns are negatively correlated. Determining portfolio weights (e.g. $w1$ and $w2$ in a scenario with only two assets), which are typically subject to constraints (e.g., $w1+w2 = 1$), is crucial to solving asset allocation problems. The rate of return of a portfolio is calculated as a weighted sum of the individual returns of each asset, similarly to the expected return. In the simple case of a portfolio with two assets, the portfolio's variance takes into account individual asset variance and covariance, while with a larger number of assets it involves a covariance matrix.

In general, the most efficient portfolios are the ones that yield the highest expected return for a given level of risk. Oftentimes several efficient portfolios exist and investors choose one based on their preferred risk tolerance and return profile. The mean-variance optimization approach serves as the foundation of portfolio selection, which helps investors create portfolios that match their expectations for returns and risk tolerance.

3.2. Risk Measurement

As an investor, the concept of risk is fundamental to understand when investing in assets, since it directly impacts the potential return on the investment. In general, when we talk about risk we refer to the uncertainty of future outcomes and the likelihood of not achieving the expected returns, or in other words the probability that an asset we have invested in loses value. Naturally, to get the most value out of an investment one would need to minimize the level of risk as much as possible. This calls for utilizing a suitable approach to measure the risk, the result of which can then be implemented in a portfolio optimization model to account for the measured risk when looking for optimal asset allocation in a portfolio. [Paolo Brandimarte (2013)] describes several risk measurement methods, most notably Standard Deviation, Min Abs Deviation, Value at Risk and Conditional Value at Risk, each of which provides a different perspective on potential losses. These are traditional evaluation approaches, which are frequently applied in real-life scenarios, such as credit risk assessment. Furthermore, as emphasized by [Eliana Angelini, Giacomo di Tollo, Andrea Roli (2008)], they can even be enhanced with advanced techniques like neural network integration to analyze risk more comprehensively and thus provide a deeper understanding of it. However, first and foremost this requires good knowledge of how risk measurement methods work to represent risk in a portfolio model. To gain more insight on these measurements, we are now going to take a more in-depth look at each of them, including what makes each one different from the rest and in what kinds of situations they are advantageous over other risk measuring approaches.

3.2.1. Standard Deviation. A simple and light on computations risk measurement approach, Standard Deviation is a crucial part of mean-variance portfolio optimization. It estimates risk by first comparing the mean return of each asset investment relative to its volatility, calculating the difference between the two, i.e the return's deviation. Each computed deviation is then squared so that negative values are ignored and all the results are averaged to get the variance. Finally, the square root of the variance is taken to scale it down, effectively calculating the Standard Deviation. The higher the Standard Deviation, the greater the variance and therefore a higher probability of the expected return deviating from the mean return, meaning a higher risk level.

Standard Deviation assumes that returns are normally distributed, which conveniently makes computations fast and not as heavy as other risk measurement methods. This is oftentimes sufficient to provide a good general representation of risk measure that is easy to read and understand. Nevertheless, this approach may not always be quite as accurate as an investor might expect, exactly due to the normal distribution assumption, since in some situations mean distributions are not necessarily equally risky. Moreover, it is not suitable for application in very specific scenarios, because it does not account for additional factors in more extreme investment events, but it still remains as a reliable way to measure risk in the general case.

3.2.2. Min Abs Deviation. An approach that shares similarities to Standard Deviation is Minimum Absolute (Min Abs) Deviation. It is similar in the sense that risk is measured by performing almost the same computations. The difference in Min Abs Deviation lies in the second step, where each asset return's deviation's absolute value is taken instead of the squared deviation value. Next, these absolute values are averaged, which returns the Min Abs Deviation value. This method gives equal weight to all deviations no matter their size, making it more capable of handling extreme investment events than Standard Deviation.

The fact that Min Abs Deviation provides a more meaningful and intuitive measure representation of mean deviations makes it well suited for scenarios in which the mean distributions are skewed or non-normal or when there are not frequent asset price fluctuations. However, on top of being more computationally expensive due to the lack of convenient squaring operations, this balanced perspective on risk comes at the cost of a simplified view which makes it less able to reliably assess the risk of portfolios in highly volatile markets with frequent changes to asset prices.

3.2.3. VaR and CVaR. Value at Risk (VaR) and Conditional Value at Risk (CVaR) are alternative and easy to understand risk measures that aim to assess the maximum possible loss for a portfolio over some time interval within a certain confidence level. According to [Paolo Brandimarte (2013)] "VaR is implicitly defined as the quantile of the probability distribution of future wealth", meaning that as a quantile-based measure it focuses on the tail-end of the probability distribution of losses. Its value is easily interpretable since it directly shows the worst loss an investor might expect based on the given confidence level. VaR can be relatively straightforward to compute under certain conditions like short time spans and assumptions of normal distributions. This is done by first calculating the standard deviations of assets and then finding the quantile of the normal distribution. For longer periods of time VaR is very frequently affected by additional factors, such as daily return and drift due to expected return, which add more complexity to its calculation but need to be taken into account to obtain an accurate result.

CVaR extends VaR by accounting for both the quantile threshold and the estimated average of the losses that exceed the VaR threshold, thus making it a more comprehensive metric suitable for risk measurement in a portfolio with extreme asset price fluctuations. Just like VaR, CVaR focuses on the tail-end of the probability distribution of losses, so it is capable of handling more extreme losses much better. This utility is particularly useful in situations where an investor wants to know the worst-case scenario loss for a certain portfolio, especially one with large movement in asset prices.

This exploration of risk measures shows that the risk measurement process can be quite diverse in terms of the range of methods that can be utilized. Generally, a good rule of thumb is to select the most specific risk measure that best fits the investor's situation in regard to portfolio characteristics, environment of the asset market (e.g the degree of price fluctuations) and the investor's own level of risk tolerance (e.g Standard Deviation for higher tolerance, Min Abs Deviation for moderate tolerance and VaR and CVaR for lower tolerance). Each risk measure possesses strengths and weaknesses in certain scenarios and ultimately it is the investor's choice which one to put into use for assessing the risk and potential losses of given portfolio, which in itself is a skill of its own.

3.3. Covariance Matrix

To have a full view of the characteristics of a portfolio and its assets, portfolio optimization would not be complete without a way that mathematically represents the relations between assets. This representation is done through the use of a covariance matrix, which provides a mathematical model for the movement of asset returns relative to one another and is a fundamental aspect of any portfolio model.

As the name suggests, a covariance matrix relies on the concept of covariance - a key metric for assessing how much two variables change together. From a mathematical standpoint, the covariance value is said to be positive whenever the two variables move together in one direction and, conversely, it is deemed as negative whenever they move together in opposite directions. This is directly applicable to asset prices, where the covariance between different assets indicates whether or not they tend to increase or decrease in value at the same time. A positive covariance means that when one asset's price goes up, the price of the other asset is likely to do so as well. Similarly, a negative covariance suggests that if an asset decreases in value then the other one is also prone to a decrease. In general, a strong positive relationship between two assets is dictated by a higher positive covariance, while a higher negative covariance implies a strong negative relationship.

In practice, the covariance matrix itself is relatively straightforward to read and understand. The matrix for a portfolio consisting of n assets is a $n\text{X}n$ square matrix, where each entry on the main diagonal is the covariance of the respective asset with itself, while every other entry represents the covariance of the two assets corresponding to the row and column. The computation of the elements of the matrix involves an analysis of historical data on asset prices. In essence, the covariance of each pair of assets is obtained by calculating the average product of both assets' deviations from their respective mean returns over some period of time. This process consists of comparing the return of each asset at a specific point in time to the overall mean return, followed by an averaging operation of the products of the pair's deviations, ultimately resulting in the covariance value for the two assets.

By its very nature, a covariance matrix provides a simplified historical view of the simultaneous movements of asset prices, which might not always be very reliable due to the possibility of real-world scenarios having more complex asset relationships and the dynamic nature of markets, thus often requiring frequent updates of covariance values. Nevertheless, having this perspective is essential in order to gain a complete understanding of the full picture of a portfolio, as investors can take advantage of this knowledge and decide whether or not it is worth investing in certain assets.

With the prerequisite theoretical knowledge covered in-depth, we can go back to answering the scientific question "Can we reliably use MATLAB to help optimize investment choices?". As a matter of fact, MATLAB's Financial Toolbox is the key utility that will provide us with a complete answer to this question, since it contains all the tools that are relevant to the process of portfolio optimization in different situations. This includes various risk measurements such as the discussed Standard Deviation and VaR, the ability to process a covariance matrix for a given portfolio and sophisticated optimization algorithms with parameters that can be fine-tuned to facilitate the investor's specific requirements and constraints. Therefore, a point can be made that MATLAB is fully suitable for portfolio optimization problems of varying complexity and scale, as it has diverse tools related to portfolio optimization and risk assessment, allowing an investor to put into practice all of the theoretical knowledge covered until now. To further explore this in action, the next section will go into some of the utilities of the Financial Toolbox in more depth in order to solve some portfolio optimization problems that can be relevant in real-world scenarios.

4. MATLAB for Finance and Economics

The produced technical deliverable for this project is a program created with the goal of supporting the answer to the scientific question by putting into practice the described theoretical knowledge from the scientific deliverable. As a result, this practical visualization is expected to provide concrete and clear examples of MATLAB's utility in the area of finance and economics, thus validating the theoretical information presented in the scientific portion of this work.

The main functionality of the developed program is to give users, who invest in a number of assets, a range of useful tools which can compute various financial metrics for their portfolios, such as risk, rate of return and VaR, depending on the particular scenario. In this way the program aims to be useful for investors who actively seek to minimize their losses in various situations with the assistance of reliable tools tailored for this job.

On a technical level, the program consists of a simple user interface, fully developed in MATLAB. The interface integrates a main menu interface, which presents several choice selections for different portfolio-optimization related tools in the form of MATLAB functions, and a separate window interface for each implemented function, which expects its own unique input based on the scenario the particular function is suited for. The current version of the program supports 5 functions, each of which is now going to be discussed in more detail to gain insight about their purpose and usefulness for specific portfolio optimization situations. It is important to note that these functions have been inspired by code snippets provided in chapters 2.4.3 and 2.4.5 of [Paolo Brandimarte (2013)]'s work, serving as better realized versions brought up to modern MATLAB syntax and specifications.

4.1. Function 1 - Stock portfolio optimization

This is a practical implementation of basic stock portfolio optimization using MATLAB's $Portfolio$ class and the built-in $estimateFrontier$ function. Function 1 is composed of 2 parts - definition of the portfolio parameters and optimization process.

The function arguments utilized in this implementation are only 3 (though many more portfolio parameters are available to be set) - $ExpRet$ is an array containing the expected return of an arbitrary number of stocks, $CovMat$ is the covariance matrix of stock returns and $numPortfolios$ indicates the total number of efficient portfolios to be estimated, computed from most efficient to least efficient up to that limit.

The function first uses the values of $ExpRet$ and $CovMat$ to create a $Portfolio$ object. A MATLAB $Portfolio$ must always incorporate some constraints, but since no specific constraints are needed in this example, the default ones are set with the built-in $setDefaultConstraints$ function. The problem is then optimized to find an efficient frontier using $estimateFrontier$, which is the set of portfolios offering the maximum expected return for a given level of risk. In other words, the n most efficient portfolios are computed, where n corresponds to the value of the $numPortfolios$ argument. This is followed by a plot of the efficient frontier with $plotFrontier$ to visualize the risk-return trade-off of the computed portfolios. Finally, $estimatePortMoments$ computes the risk (standard deviation) and the mean of return for the resulting portfolios to provide more clarity to their risk-return profile.

4.2. Function 2 - Creating constraint matrix

This particular function is not as practical as the rest, since it only creates a constraint matrix for portfolio optimization without actually doing anything with it afterwards. Function 2 is later reused in the implementation of Function 3, so the sole purpose for it existing as a separate function is to verify that a constraint matrix is created properly with a given input without the need to do so explicitly in another function.

The function takes 6 arguments - number of assets $NAssets$, minimum and maximum investment constraints for the assets, $AssetMin$ and $AssetMax$ respectively, a $Groups$ matrix representing how assets are grouped together and minimum and maximum constraints for the groups, $GroupMin$ and $GroupMax$ respectively. Both $AssetMax$ and $AssetMin$ are expected to receive a vector as a value, where each individual entry of the vector is a value representing the constraint for the corresponding asset. To create the constraint matrix, the built-in function $portcons$ is used, which requires group parameters as input alongside the expected asset number and asset constraints. In this case groups essentially serve as categories, which assets may or may not share. This is reflected in the $Groups$ matrix, where the number of rows represents the number of different groups and each row contains values indicating whether the corresponding assets belong to that group or not (1 if they do, 0 if they do not). Additionally, the group constraints $GroupMin$ and $GroupMax$, just like the asset constraints, expect a vector as input value, which contains as entries the investment constraint value for each group. All these values are then passed as parameter values to $portcons$, generating a matrix of constraints using linear inequalities. The inequalities are of the type $A * Wts <= b$, where Wts is the matrix representing the portfolio weights, A defines the coefficient matrix, i.e. coefficients for each asset or asset group, and b is the constraint vector that indicates the upper bounds for the values.

4.3. Function 3 - Portfolio optimization with constraint matrix

This function is essentially an extension of Function 1, combining it with Function 2 to facilitate a more optimized approach of asset allocation due to the presence of specific constraints. As such, the expected inputs are the exact same 6 arguments of Function 2 alongside the 3 arguments of Function 1, all of which have already been described in the previous two subsections.

Similarly to Function 1, Function 3 can be defined as a composition of 2 parts, those being the portfolio parameter definition, which now additionally includes the creation of the constraint matrix, and the optimization process, which follows the same overall steps. The function first constructs a constraint matrix from the given $NAssets, AssetMin, AssetMax, Groups, GroupMin$ and $GroupMax$ values, as described previously. The coefficient matrix A and constraint vector b from the resulting linear inequalities are then retrieved separately because this is later needed when setting the portfolio to use the constraint matrix instead of the default constraints. Following that, the Portfolio object is initialized using the values of the expected returns $ExpRet$ and the covariance matrix $CovMat$. After that the constraints are finally set by overriding the portfolio's default constraints with the $setInequalities$ function and the previously extracted values of A and b. Finally, the optimization computations are done just as in Function 1, estimating the efficient frontier for the specified number of portfolios $numPortfolios$ and calculating the risk (standard deviation) and return for each portfolio. For the purpose of clarity, the risk, return and weights are printed in the standard output and the resulting efficient portfolios are visualized as a plot.

4.4. Function 4 - Portfolio optimization with risky and risk-free asset

This is a different implementation of portfolio optimization, in the case where no constraint matrix is considered and instead there exist multiple risky assets and a risk-free asset. Similarly to Function 1 and Function 3, this function takes an array of the expected returns $ExpRet$ for an arbitrary number of assets, which are only risky assets, and their covariance matrix $CovMat$. Additionally, there are 3 other input arguments - $RisklessRate$ defines the rate of return of the risk-free assets, $BorrowRate$ indicates the rate at which funds can be borrowed to account for the potential cost of borrowing (must be equal or greater to the risk-free rate) and $RiskAversion$ is a value usually in the range from 2.0 through 4.0, which represents the coefficient of the investor's risk aversion level.

Once these parameters are defined, the function proceeds to the optimization computations, where first the risk level, rate of return and weights for the risky assets are computed with the built-in $portopt$ function which directly performs portfolio optimization given the values of $ExpRet$ and $CovMat$. The reason for utilizing $portopt$ instead of the previously seen $estimateFrontier$ is convenience, since it does not require a predefined portfolio object as input and also this time we have no real use for only plotting the efficient portfolios like we have done before because we are rather looking for the optimal investment allocation between the risky and risk-free assets. Having calculated the values of the risk, return and weights, the built-in $portalloc$ function is called, which computes the optimal investment allocation by taking these 3 values as arguments for the risky assets alongside the $RisklessRate$ for the risk-free asset, $BorrowRate$ and $RiskAversion$. It then creates a plot representing the efficient frontier and the capital allocation line, highlighting the optimal risky portfolio on the frontier and the optimal overall portfolio where the risk-free asset is introduced.

4.5. Function 5 - Value at Risk computation

The final function of the program calculates the VaR of a portfolio with 2 assets. This function can be used in conjunction with the previous portfolio optimization implementations, as it can assess the optimized portfolio's VaR based on the portfolio's risk, expected returns and the computed weights. It takes as input the standard deviations, i.e. risk, of each asset, respectively $s1$ and $s2$, their correlation coefficient rho, an array $ExpRet$ containing their individual expected returns, another array $PWts$ of the allocated weights to each asset, a third one representing the portfolio's expected return $PRoR$, the investor's $confidenceLevel$ (a value greater than 0 and less than or equal to 1), which corresponds to the probability of losses exceeding the VaR, and the combined value of the portfolio assets $PValue$.

A covariance matrix $CovMat$ is constructed using their risk levels and correlation coefficient by mathematically determining the extent to which the returns of the two assets move together based on historical data. The expected returns, asset weights and the derived covariance matrix are then used to compute risk for the portfolio with the built-in function $portstats$. Finally, $portvrisk$ integrates the values of the portfolio's risk, return and combined value alongside the confidence level to calculate the worst expected loss, which is essentially the VaR value.

4.6. Practical examples

In this section the actual utility of the program will be shown by presenting a single example usage for each of the 5 functions. Beginning with Function 1, three different assets are considered with corresponding expected returns of 12%, 15%, and 10% (Fig. 1). The covariance matrix includes values that indicate different levels of return fluctuations and correlations between asset returns, where a negative correlation implies that asset returns do not all move in the same direction. To explore a wider range of investment combinations, the number of efficient portfolios to compute is set to 30. The result is a plot of the efficient frontier (Fig. 2), illustrating a curve that demonstrates the trade-off between risk (Standard Deviation of Portfolio Returns) and expected return (Mean of Portfolio Returns). The risk level rises as we move from left to right along the curve, but so does the possible return. Therefore portfolios that are located on the left end of the curve are less risky but also have lower returns, while those on the right end have higher risk but can also expect much higher potential returns.

Next up, Function 2 constructs a constraint matrix given the following input (Fig 3.): assuming a portfolio with 4 assets, we want to set a minimum investment of 5% and maximum investment of 40% for each asset, and we also want to group together the first two in a single group and the last two in another, where at least 30% of the portfolio should be invested in each group but no more than 70%. The resulting constraint matrix (Fig 4.) reflects these parameters, with the positive values (1) along the diagonal of the matrix consisting of the first 3 columns corresponding to the minimum proportion of each asset and the negative ones (-1) to the maximum ones. Meanwhile, the last 2 columns represent the maximum and minimum constraints, indicating the group with either 1 or -1 in the first column and either its corresponding upper (if the number is positive) or lower (if the number is negative) bound in the second.

For the example of Function 3 (Fig. 5), since it basically integrates the entirety of Function 2, we will use the same values for building the constraint matrix. The rest of the input is also similar to the input of Function 1, but with adjusted to take into account 1 additional asset, since the current constraint matrix considers 4 assets. In particular, for our fourth asset we have an expected return of 9% and we further have an updated covariance matrix accounting for the new asset. The plot of the efficient frontier (Fig. 6) is similar to the one from the example of Function 1, showing a curve that represents the gradually increasing risk and return of the computed efficient portfolios.

Moving to Function 4 (Fig. 7), we have 3 assets with corresponding returns 10%, 12% and 14% and a covariance matrix similar to the one used in the first example. The risk-free rate is set to a rather conservative value of 5% and so is the borrowing rate, while the risk aversion is 3, which indicates it is a moderate amount (given the usual risk aversion range of 2 through 4). The result is again a graph that reflects the efficient frontier (blue curve) but this time it incorporates the risk-free asset by including the capital allocation line (colored in magenta). The point where the capital allocation line and the efficient frontier intercept is marked as the optimal risky portfolio, which is on the left end of the blue curve. The optimal overall portfolio is shown to be further up the capital allocation line, marked with a red plus. This indicates a higher expected return than the optimal risky portfolio, due to the inclusion of the risk-free asset.

Finally, for Function 5 we have the following inputs (Fig 9.): we assume we have standard deviations of 0.02 and 0.02 for assets 1 and 2 respectively, a correlation coefficient between them of 0.7, no expected individual returns for any of the assets, corresponding weights of 60% and 40%, no expected return for the portfolio, a confidence level of 1% and a total portfolio value of 10 000 000. The computed VaR is the value 350 654.02. From this we can infer that there is only a 1% chance that the portfolio will lose more than 350,654.02, therefore the investor can be highly confident that no loss will exceed this amount despite the portfolio having no expected return.

5. Conclusion

This report has presented the concept of portfolio optimization and the capabilities of MATLAB's Financial Toolbox module to optimize a portfolio's asset investment allocation and measure its risk, finding the most efficient portfolios with optimal balance between risk and return. It has done so by first exploring in-depth theoretical topics relevant to portfolio optimization as the project's scientific deliverable, followed by a demonstration and thorough explanation of a program developed with the use of Financial Toolbox tools, serving as the technical deliverable.

The question "Can we reliably use MATLAB to help optimize investment choices?", established in the beginning of the report, has been answered clearly through the presentation of the theoretical information and the practical program, which support the claim that MATLAB is indeed a reliable tool very well suitable for use in problems regarding optimization of investment choices. The completeness of both sections is also within expectations, providing a deep dive of the notions which portfolio optimization depends on and full descriptions of the functions that comprise the final program. One minor addition to the technical deliverable that could have served as an improvement is the inclusion of a more complex function, tailored to deal with a more extreme financial scenario. Nevertheless, for the purpose of this project, the program's basic nature sufficiently presents the fundamental utilities of MATLAB to assist investors with their investment choices.

To sum up, the findings of this project confirm that MATLAB is a dependable computational software that can be applied in real-world scenarios where the optimal asset return of a portfolio is sought after, as well as for measuring the risk and potential losses. The project has provided insights into portfolio management and a handful of the tools contained in MATLAB's Financial Toolbox, which are expected to serve as valuable fundamental knowledge for investors to make informed decisions in their financial endeavours.

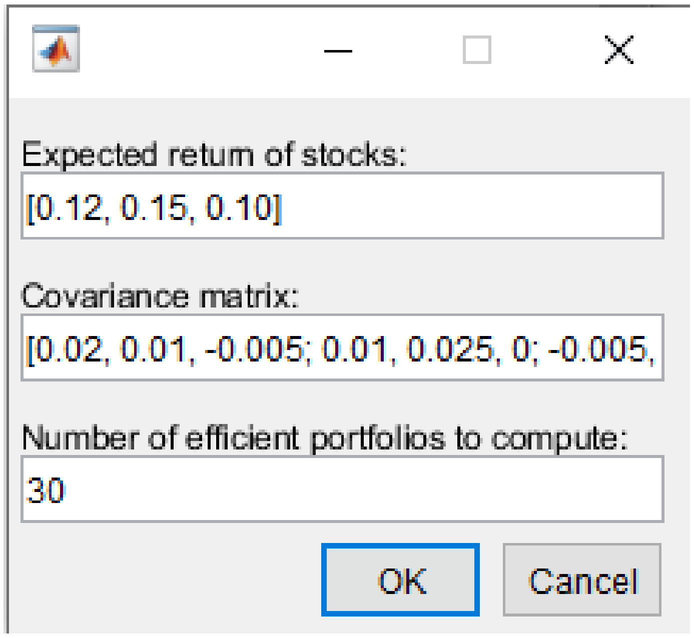
References

[Paolo Brandimarte (2013)] Numerical Methods in Finance and Economics: a MATLAB based introduction, Second Edition

[Joseph Andria, Giacomo di Tollo, Jaan Kalda (2022)] The predictive power of power-laws: An empirical time-arrow based investigation. Chaos, Solitons & Fractals Volume 162, September 2022, 112425

[Eliana Angelini, Giacomo di Tollo, Andrea Roli (2008)] A neural network approach for credit risk evaluation The Quarterly Review of Economics and Finance, Volume 48, Issue 4, 2008, ISSN 1062-9769, https://doi.org/10.1016/j.qref.2007.04.001.

6. Appendix



Expected return of stocks:

[0.12, 0.15, 0.10]

Covariance matrix:

[0.02, 0.01, -0.005; 0.01, 0.025, 0; -0.005,

Number of efficient portfolios to compute:

30

Fig. 1. Example input for Function 1

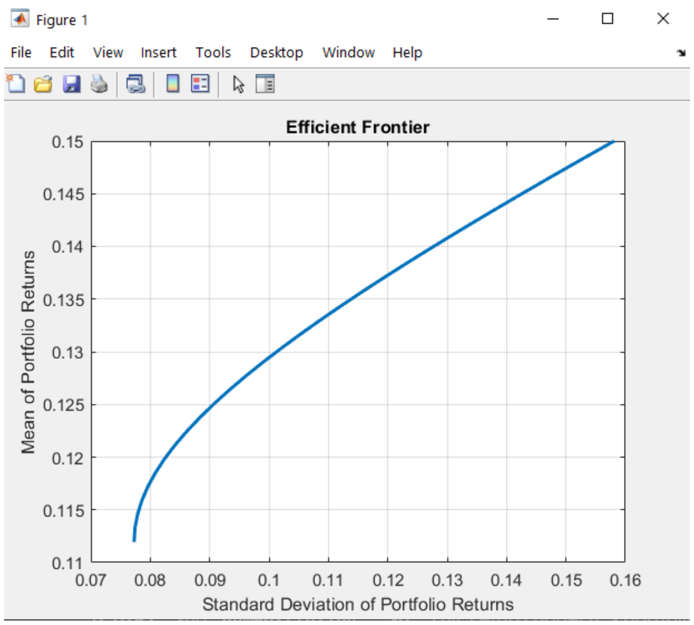
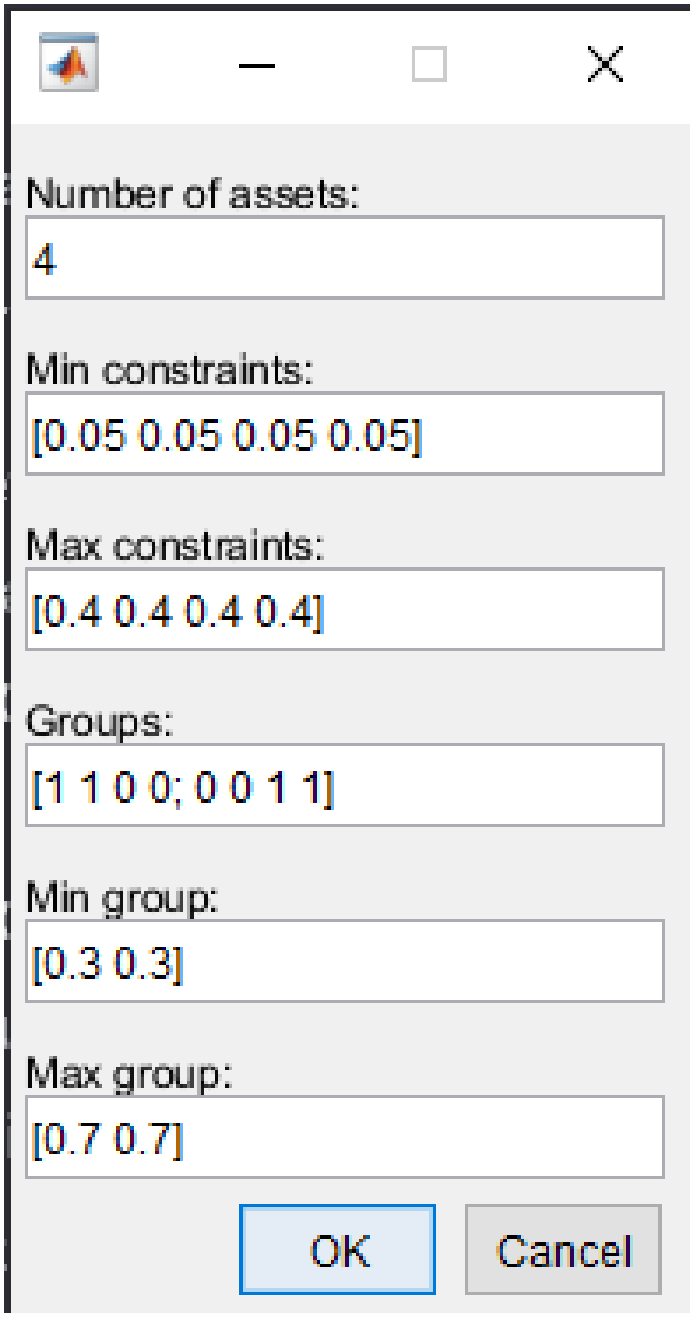


Fig. 2. Output plot for Function 1



Number of assets:
4

Min constraints:
[0.05 0.05 0.05 0.05]

Max constraints:
[0.4 0.4 0.4 0.4]

Groups:
[1 1 0 0; 0 0 1 1]

Min group:
[0.3 0.3]

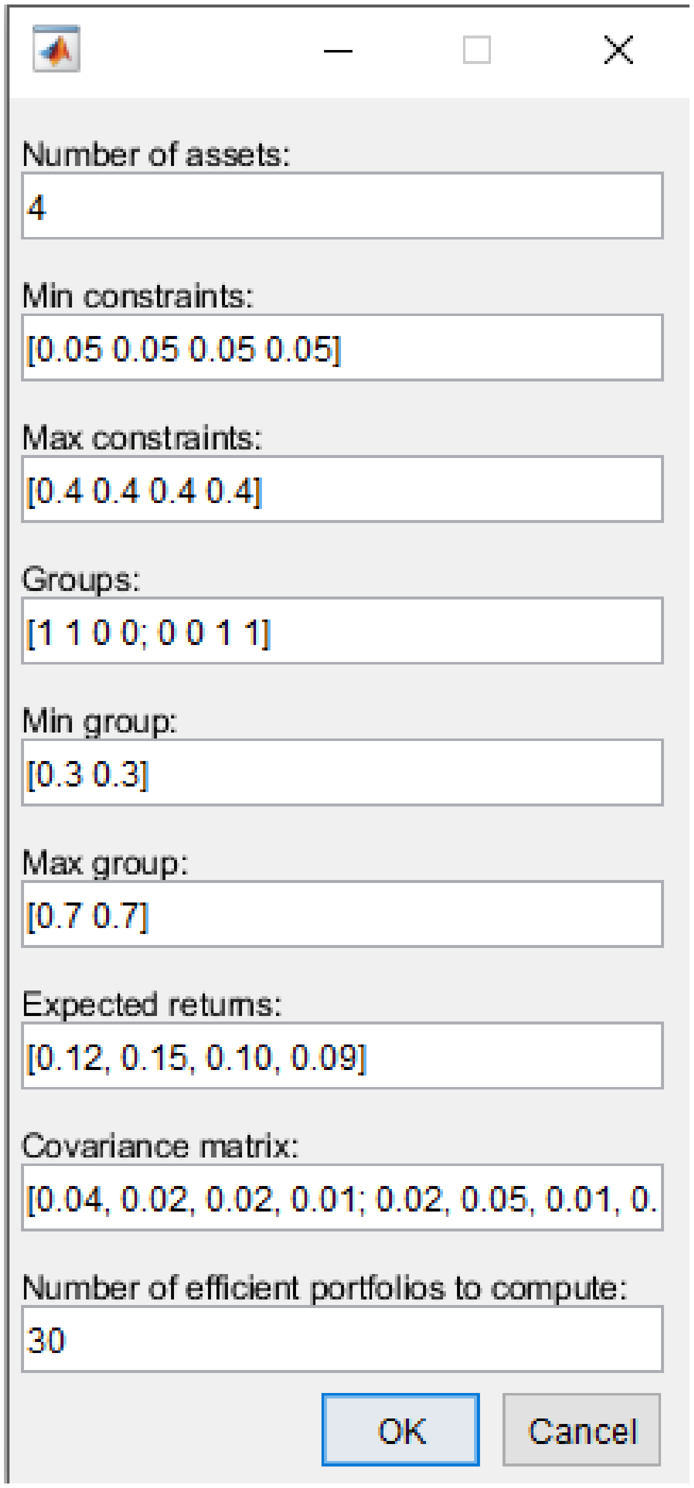
Max group:
[0.7 0.7]

Fig. 3. Example input for Function 2

```
ConstrMatrix =

    1.0000    1.0000    1.0000    1.0000    1.0000
   -1.0000   -1.0000   -1.0000   -1.0000   -1.0000
   -1.0000         0         0         0         0
         0   -1.0000         0         0         0
         0         0   -1.0000         0         0
         0         0         0   -1.0000         0
    1.0000         0         0         0    0.4000
         0    1.0000         0         0    0.4000
         0         0    1.0000         0    0.4000
         0         0         0    1.0000    0.4000
   -1.0000         0         0         0   -0.0500
         0   -1.0000         0         0   -0.0500
         0         0   -1.0000         0   -0.0500
         0         0         0   -1.0000   -0.0500
   -1.0000   -1.0000         0         0   -0.3000
         0         0   -1.0000   -1.0000   -0.3000
    1.0000    1.0000         0         0    0.7000
         0         0    1.0000    1.0000    0.7000
```

Fig. 4. Computed constraint matrix for Function 2


Number of assets:

4

Min constraints:

[0.05 0.05 0.05 0.05]

Max constraints:

[0.4 0.4 0.4 0.4]

Groups:

[1 1 0 0; 0 0 1 1]

Min group:

[0.3 0.3]

Max group:

[0.7 0.7]

Expected returns:

[0.12, 0.15, 0.10, 0.09]

Covariance matrix:

[0.04, 0.02, 0.02, 0.01; 0.02, 0.05, 0.01, 0.

Number of efficient portfolios to compute:

30

Fig. 5. Example input for Function 3 (note that it reuses the contraint matrix from the previous example)

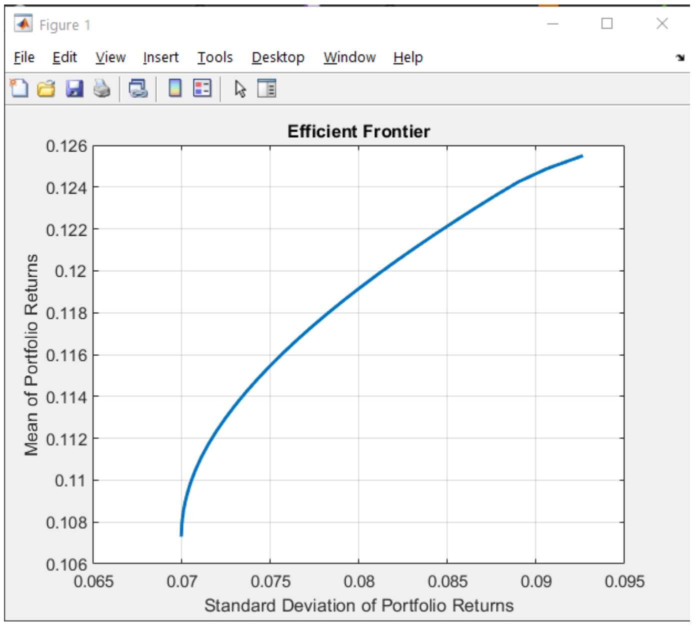
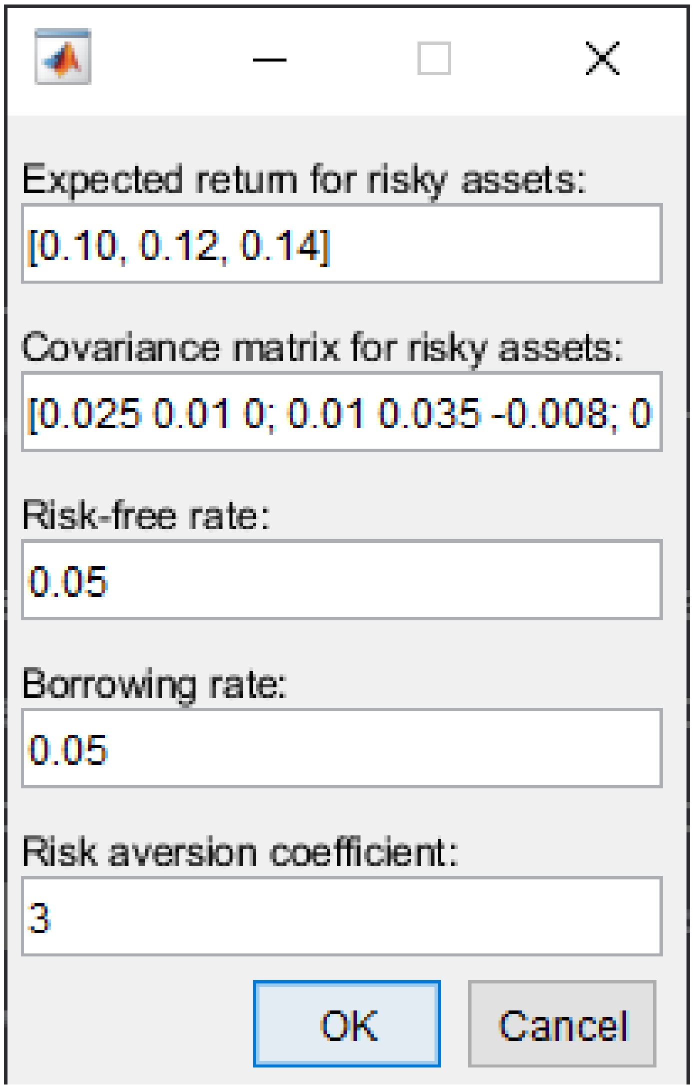


Fig. 6. Output plot for Function 3


Expected return for risky assets:

[0.10, 0.12, 0.14]

Covariance matrix for risky assets:

[0.025 0.01 0; 0.01 0.035 -0.008; 0

Risk-free rate:

0.05

Borrowing rate:

0.05

Risk aversion coefficient:

3

Fig. 7. Example input for Function 4

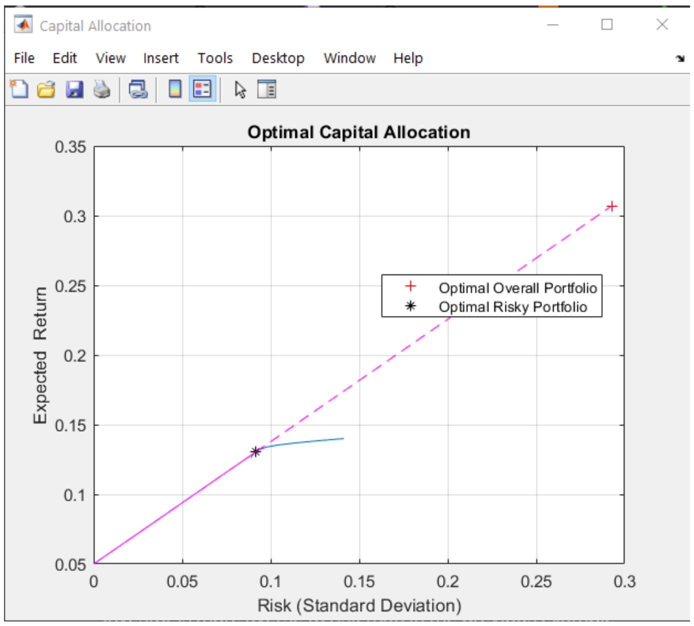


Fig. 8. Plot output for Function 4 (magenta line indicates the capital allocation line)

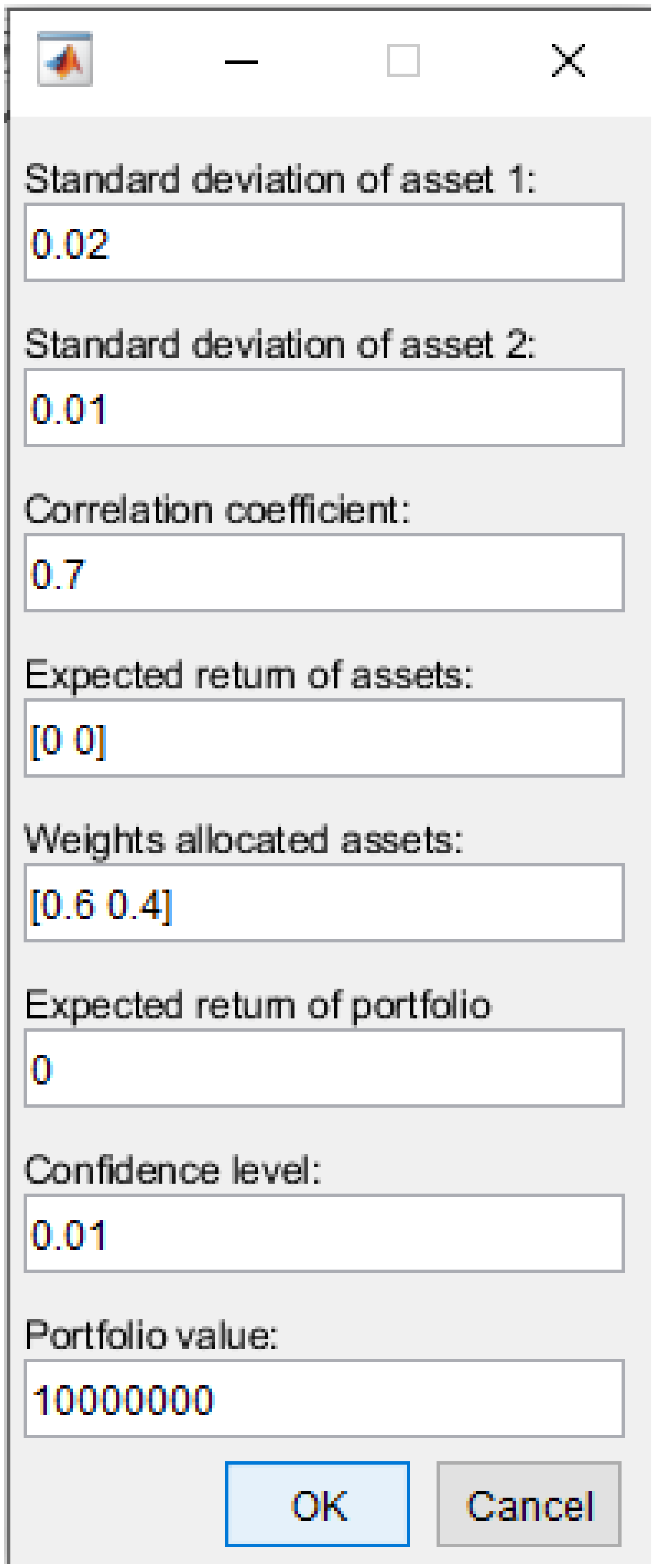
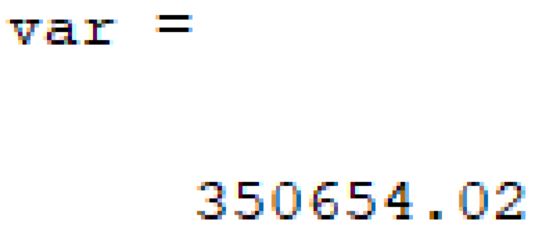

Standard deviation of asset 1:
0.02
Standard deviation of asset 2:
0.01
Correlation coefficient:
0.7
Expected return of assets:
[0 0]
Weights allocated assets:
[0.6 0.4]
Expected return of portfolio
0
Confidence level:
0.01
Portfolio value:
10000000

Fig. 9. Example input for Function 5



```
var =

     350654.02
```

Fig. 10. Computed VaR for Function 5