|  |
| --- |
| Photo displaying partial image of two pie charts on a canvas-textured page |
| Genetic Algorithms in Portfolio Management  A term project on the application of Genetic Algorithms in the domain of portfolio management and optimization |
| |  |  |  | | --- | --- | --- | | Debraj Chatterjee, 19IM10039 | Spring 2021 | Genetic Algorithms in Engineering Process Modeling (MT21104) | |

**Abstract**

Portfolio management is one of the most interesting topics in finance as it has great practical application for a wide range of entities. Portfolios of investments are held by venture companies, institutional investors, banks and last but not the least, every single individual retail investor. Therefore, it is all the more important that extensive research is done on this topic. The aim of portfolio management research is optimizing a portfolio for various factors such as minimum variance (risk), maximum return, extensive diversification, etc.  
In this regard there has been development of various algorithms and techniques for portfolio optimization. Genetic algorithms are also catching upon in this domain and recently some institutions are starting to implement them for the same.

This project is an attempt at building an optimum portfolio using genetic algorithms, specifically differential evolution.

**Introduction**

For most people investments should be less risky. More than focusing on large returns, they focus more upon minimizing the level of risk of their portfolio. The risk of a portfolio in a classical sense is the total variance of the annual returns given by the portfolio. This is the so-called minimum variance portfolio. The variance of the portfolio will depend upon the riskiness (individual variances) of the assets in it and the correlations among the different assets.

This project is aimed at building a minimum variance portfolio using genetic algorithm techniques, specifically differential evolution. The algorithm will decide the weights of the different assets in the portfolio in an evolutionary way. The problem has been solved for two assets (which is the simplest base case) and also for three assets (which is as good as the general case of multiple assets, greater than two in number).

**Problem Formulation**

The decision variables of the problem are the weights of the different assets in the portfolio. Therefore, each candidate solution is a portfolio. This candidate solution is a vector which has the weights of the assets as its components.

   
where is the number of assets in consideration in the portfolio and  is the weight of asset 

Obviously, the range of each  is , as it is the fraction (weight) that the asset holds in the total portfolio investment budget.

There are two types of constraints in the problem formulation –

1. The non-negativity constraint which specifies that the weights of the assets should be non-negative, which is logical as negative investment cannot be made into an asset.



1. The constraint which specifies that the sum total of all the fractions (weights) of the assets in the portfolio should be unity.



The constraints (1) and (2) together take care of the fact that the weight of each asset should not be greater than unity as



Hence, explicitly stating this as a separate constraint is not necessary. However, we will still specify it explicitly for faster convergence.

The function to be optimized (minimized) is the total variance of the portfolio. It is given by



Where  represents the individual variance in returns (riskiness) of asset  and   
 represents the covariance of asset  and asset.  
Covariance is a measure of how the returns of one asset, vary with respect to the returns of another asset in the same year. For diversification, it is desired that the covariance of the portfolio is not  , which would mean that if one asset makes losses, the other one will too. The optimum choice is to have the assets as independent to each other which would make the covariance term as .

**Developing Typical Genetic Algorithm Parameters**

Common Genetic Algorithm parameters that have to be developed and specified for all genetic algorithms are the

1. Initial Population (including the population size)
2. Fitness Function (integrating the constraints)
3. Average fitness of the population and Best fitness (fitness of the fittest population member)
4. Termination Condition

**Initial Population -**

The initial population is created by creating a population member (solution candidate – a portfolio), by randomly assigning weights to each asset in the portfolio such that the total weight becomes unity, and this process is repeated over and over to create the entire population.   
For Differential Evolution, it has been studied that having a initial population size equal to 10 times the number of variables is optimal. Therefore, in this case, the optimal population size is 10 times the number of assets.

 ----(4)

**The Fitness Function –**

The fitness function in the problem is actually a measure of how “less risky” the portfolio is. We will try to maximize the “less riskiness” of the portfolio, or maximize this fitness. In this sense, we are actually minimizing the total risk of a portfolio through successive generations. Hence, we have to make appropriate modifications to the function in equation (3).  
To achieve this, we take the negative of the function in equation (3) as one part of the fitness, as maximizing the negative of a function is equivalent to minimizing the original function.  
The inverse of the original function could also have been taken, however it has been avoided as if the risk becomes zero, then division by zero would become undefined.



Also, a penalty should be applied to the fitness for violating constraint (1) and constraint (2).

For constraint (1), the penalty applied is as follows –

The penalty term is taken to be . A count  is kept of the number of variables that violate this constraint. The penalty term is multiplied by the number of variables that violate the constraint and the total is subtracted from the fitness.

 ----(6)  
where 

For constraint (2), the penalty applied is as follows -

The penalty term is taken to be . This penalty term is multiplied by the absolute value of the difference of the sum of the weights of the assets, with 1. This ensures that the constraint becomes adaptive and as the sum of the weights gets closer and closer to 1, the penalty becomes lesser and lesser and finally 0, when the sum is 1.



Combining the equations (5), (6) and (7), we get the total fitness function as

 --(8)

**Average Fitness and Best Fitness –**

Average fitness of the population is the average of the fitnesses of all the members of the population.

-----(9)

The best fitness is the fitness of the fittest individual in the population.

----(10)

**Termination Condition –**

There is no specific termination condition in genetic algorithms. The termination condition is according to the satisfaction of the user. However, a few common measures are implemented some of which are –

* Number of Generations
* Average fitness of the population relative to the best fitness
* Best fitness of the population

In this problem, I have taken the termination condition to be the number of generations. This has been taken as an overestimate and in fact, the average fitness becomes equal to the best fitness of the population by the termination number of generations in a lot of trials done by me.  
Therefore, it is a pretty good termination condition as the population will not evolve any further.

The number of generations for evolution taken by me is 1000.

This is because I have observed that the population converges by so many generations and there is no further scope of evolution. The whole population becomes equal to the same value.

**The Differential Evolution Algorithm**

Differential Evolution is a real encoded genetic algorithm which is primarily based upon mutation (differential). Therefore, I have to develop the differential mutation specifically for this problem.

**Crossover –**

The crossover in differential evolution is unique. A random point (variable) is chosen as the starting point for the crossover and is exempted from the crossover process. The rest of the variables undergo the crossover process depending upon whether the crossover probability is favorable at that point.

I have chosen the crossover factor as  and crossover is carried out whenever for a variable, a random number generated between 0 and 1 (the mask) is less than or equal to CF.

If the crossover probability is favorable, parent 2 from the original population in the generation which has been chosen randomly before starting the process, passes its variable value to the modified candidate, else the parent 1 (parent of the to be formed modified candidate) passes the variable value to the modified candidate.

By repeating this process, a population of modified candidates of the same size of the population in the generation is formed.



**Mutation -**  
For Differential Evolution, the mutation happens by perturbing a modified population member  by a mutation factor  multiplied by the vector differential of two other randomly chosen original population members . This process is done upon the modified candidate formed after crossover. This ultimately forms the intermediate candidate.



The mutation factor taken by me is  (a slower learning rate) as it proved to be good results after a lot of trial and error. A traditional value like 0.5 or greater was resulting in the variables to go much above their feasible range as the mutation factor would be essentially of the same order as that of the variables, that is, O(1).

Thus, for each population member an intermediate member is generated and stored after crossover and mutation.

**Fitness Evaluation and Population Updating –**

To evolve the population for repeating the whole process in the next generation, the fitness of the population members and their respective intermediate members is evaluated and compared.  
If the fitness of the intermediate candidate  is better than that of the original population member  in the generation, then the population member is swapped out with the intermediate population member, thus increasing the overall fitness of the original population and updating it.

I have also followed this process and evaluated the fitness of the intermediate and original population members according to equation (8).



The average fitness and best fitness of the population is then updated according to equations (9) and (10). This completes one whole generation (iteration) of the population.  
The process is repeated till the termination condition (maximum number of generations is reached).

**Procedure**

In this section I will elaborate the complete qualitative procedure that I have undertaken in my approach, starting from initialization, modeling and executing the differential evolution algorithm.

1. **Initialization of the problem and taking user inputs**

* At first, the number of assets that are to be taken into consideration for the portfolio is taken as input from the user, following which the names of the assets are asked, and the same is stored in a list.
* A general object ‘Asset’ is created which will contain the name of the asset, its annual returns history, its average return and its average risk (variance) as its properties.
* The functions for calculating the variance and covariance are defined.
* Input is taken from the users about the number of years of past returns that is to be considered. The returns are stored and based on this, the past returns for each asset are asked for, and the variance and average return are calculated and stored as properties of the object ‘Asset’, using the above defined functions.
* Using this, the variances and standard deviations of each asset are stored in two separate lists.
* An object ‘Portfolio’ is defined which will contain the list of weights of each asset in it as its attribute.

1. **Developing Common Parameters related to Genetic Algorithms**

* **Building the Initial Population -** A function is defined that builds up the initial population of portfolios (candidate solutions). For this, weights are randomly assigned to each asset in a portfolio candidate solution, till the sum is 1. This is repeated for the population number defined in (4) to build the initial population.
* **Function for the fitness –** A function is defined that calculates the risk of a portfolio using (3). Since it has to be minimized, the negative value is taken and the appropriate penalty for violating the constraints, as explained in the problem formulation is also incorporated, using the form of equation (8).
* **Functions for Average and Best Fitness –** Functions for calculating the best and average fitness of the population is defined based on equations (9) and (10).

1. **The Differential Evolution Algorithm**

* **Initial Population Creation -** An initial population is created and displayed using the respective function
* **Differential Evolution -** The steps for the Differential Evolution Algorithm are specified and it is executed.
* **Population Fitness Evaluation and Updating –** The best and average fitness of the population is updated according to equations (9) and (10).
* **Output -** The output gives the final population, the average fitness, the best fitness, the fittest population member (best portfolio) and displays the weights that should be assigned to the different assets (according to that in the best portfolio – fittest population member)

**Results**

**2 Assets –**

The first run of trials considers two assets – Stocks and Gold. The number of years of past returns studied is three (2018, 2019, 2020).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Returns in Year (%) | | | |
| Assets |  | **2018** | **2019** | **2020** |
| **Stocks** | 30 | 20 | 25 |
| **Gold** | 5 | 9 | 7 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Run Number | Weights | | Best Fitness | Average Fitness |
|  | **Stocks** | **Gold** |  |  |
| 1 | 0.11777 | 0.86623 | -16003.0834 | -16003.08343 |
| 2 | 0.2871 | 0.3050 | -407978.5447 | -407978.5447 |
| 3 | 0.4586 | 0.4012 | -140161.1739 | -140161.1739 |
| 4 | 0.3414 | 0.3926 | -266038.9539 | -266038.9539 |
| 5 | 0.4445 | 0.7899 | -234445.6649 | -234445.6649 |
| 6 | 0.2443 | 0.8189 | -63167.4406 | -63167.4406 |
| 7 | 0.3792 | 0.3216 | -299198.1422 | -299198.1422 |
| 8 | 0.5653 | 1.1247 | -690035.4335 | -690035.4335 |
| 9 | 0.7479 | 0.3787 | -126579.6782 | -126579.6782 |
| 10 | 0.2967 | 0.3692 | -334138.9239 | -334138.9239 |
| 11 | 0.2430 | 0.8065 | -49499.3824 | -49499.3824 |
| 12 | 0.4126 | 0.6147 | -27368.7545 | -27368.7545 |
| 13 | 0.3602 | 0.6703 | -30489.7135 | -30489.7135 |
| 14 | 0.4653 | 0.7097 | -174939.1438 | -174939.1438 |
| 15 | 0.2489 | 0.6157 | -135436.5341 | -135436.5341 |
| Mean | **0.3742** | **0.6123** |  |  |

An example of the initial population (corresponding to Run 1 of the above table) is –

[0.5726977534100763, 0.42730224658992366]

[0.3984621525543277, 0.6015378474456723]

[0.3729229867529382, 0.6270770132470618]

[0.6412524850378274, 0.3587475149621726]

[0.8774308399848905, 0.12256916001510954]

[0.9598933274490431, 0.0401066725509569]

[0.39190649023245727, 0.6080935097675427]

[0.02588228402683257, 0.9741177159731674]

[0.1168321416593191, 0.8831678583406809]

[0.2046924907955915, 0.7953075092044085]

[0.3157881704981076, 0.6842118295018924]

[0.48737852909029933, 0.5126214709097007]

[0.21746906217782802, 0.782530937822172]

[0.5106936725488136, 0.4893063274511864]

[0.8218962362473979, 0.17810376375260206]

[0.02731124627829362, 0.9726887537217064]

[0.27385043235635054, 0.7261495676436495]

[0.36322318828519295, 0.636776811714807]

[0.10180230412799907, 0.8981976958720009]

[0.2064644498297209, 0.7935355501702791]

The first number in the array represents the weight of stocks in the portfolio and the second number represents the weight of gold.

And the corresponding final population and output is

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

[0.11776725 0.86622967]

The average fitness is:

-16003.08343241763

The Best fitness is:

-16003.083432417632

The weights of the assets in the best portfolio are:

[0.11776725 0.86622967]

Weightage of Stocks in your portfolio should be

0.1177672452932667

Weightage of Gold in your portfolio should be

0.8662296713615073

As can be seen, all the population members in the final population are alike and hence, the population has converged.

From the 15 trial runs, we get that the expected weight **of Stock should be 0.3742 or** **37.42%.**  
The expected weight of **Gold should be 0.6123 or** **61.23%.**

**3 Assets –**

The second run of trials considers three assets – Stocks, Gold and Silver. Here also, the number of years of past returns studied is three (2018, 2019, 2020).   
This is as good as any general case of  assets, since here covariances between multiple sets of assets are in play.  
So, this is the real test of the working of the technique.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Returns in Year (%) | | | |
| Assets |  | **2018** | **2019** | **2020** |
| **Gold** | 5 | 9 | 7 |
| **Stocks** | 30 | 20 | 25 |
| **Silver** | 18.4 | 3.1 | 7 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Run Number | Weights | | | Best Fitness | Average Fitness |
|  | **Gold** | **Stocks** | **Silver** |  |  |
| 1 | 0.5192 | 0.4940 | 0.0778 | -90936.9254 | -90936.9254 |
| 2 | 0.8081 | 0.3901 | 0.0558 | -254032.9495 | -254032.9495 |
| 3 | 0.7139 | 0.4924 | 0.2559 | -462227.5674 | -462227.5674 |
| 4 | 0.5074 | 0.0436 | 0.2124 | -236583.9227 | -236583.9227 |
| 5 | 0.3382 | -0.1147 | 0.4185 | -1357929.9165 | -1357929.9165 |
| 6 | 0.2669 | 0.5596 | 0.0322 | -141323.1512 | -141323.1512 |
| 7 | 1.0717 | 0.7620 | 0.0881 | -921798.6031 | -921798.6031 |
| 8 | 0.1414 | 0.0582 | 0.0121 | -788279.3301 | -788279.3301 |
| 9 | 0.7350 | 0.7667 | 0.2064 | -708164.1479 | -708164.1479 |
| 10 | 0.4715 | 0.3392 | -0.0281 | -1217350.4562 | -1217350.4562 |
| 11 | 0.4723 | 0.3049 | 0.1594 | -63364.9595 | -63364.9595 |
| 12 | 0.2827 | 0.5805 | 0.1013 | -35609.2563 | -35609.2563 |
| 13 | 0.4475 | 0.4777 | 0.3592 | -284333.1913 | -284333.1913 |
| 14 | 0.0509 | 0.3698 | 0.2178 | -361521.3218 | -361521.3218 |
| 15 | 0.8728 | 0.0289 | 0.3902 | -291932.0358 | -291932.0358 |
| Mean | **0.5133** | **0.3702** | **0.1706** |  |  |

An example of the initial population (corresponding to run 1 of the above table) is –

[0.9197731648375017, 0.046001176447359134, 0.03422565871513916]

[0.5568896800582841, 0.38229626940130995, 0.060814050540406006]

[0.8481511542572217 0.11035490777580167, 0.041493937966976646]

[0.7012048630103263, 0.11926203307252496 0.17953310391714883]

[0.1371796469575267, 0.7051984884872637, 0.15762186455520955]

[0.5099526108715285, 0.33851844310408863, 0.15152894602438283]

[0.934369650974655, 0.03262505874487836, 0.033005290280466726]

[0.9883664135503825, 0.0025393506433978776, 0.009094235806219708]

[0.5959804837119314, 0.1652340259371574, 0.23878549035091123]

[0.0934700013012596, 0.5271886250348342, 0.3793413736639062]

[0.7082046686009743, 0.17122777292241007, 0.1205675584766156]

[0.3356892282554955, 0.08660103061129187, 0.5777097411332126]

[0.8315406107939466, 0.08334447985257876, 0.08511490935347465]

[0.4695340073466979, 0.516113866617908, 0.014352126035394042]

[0.7631932845754698, 0.15826183918347908, 0.07854487624105122]

[0.6340400142137288, 0.22652968608101312, 0.1394302997052581]

[0.020060126039616488, 0.5397336360978335, 0.44020623786255]

[0.7788950113118839, 0.10727981190623236, 0.11382517678188375]

[0.1623407275261236, 0.7253939237111507, 0.11226534876272565]

[0.02433009889575044, 0.9620062767884089, 0.013663624315840672]

[0.3358207063863834, 0.22058760545293365, 0.44359168816068295]

[0.7384207384826588, 0.06238730204142197, 0.19919195947591928]

[0.32905126946390917, 0.1964191820958386, 0.4745295484402522]

[0.598781008146106, 0.27308966689008196, 0.12812932496381202]

[0.47535647182460805, 0.1519495810851561, 0.37269394709023584]

[0.4862427261851996, 0.2825736332054425, 0.23118364060935792]

[0.40475553367694206, 0.08929165191888308, 0.5059528144041748]

[0.6310662211473015, 0.03397029146992531, 0.33496348738277315]

[0.7424183568267059, 0.08497877865611236, 0.17260286451718176]

[0.11064646253618071, 0.3613171320802245, 0.5280364053835949]

The first number in the array represents the weight of Gold in the portfolio, the second number represents the weight of Stocks, and the third number represents the weight of Silver.

And the corresponding final population and output is –

The final population is:

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

[0.51915644 0.49396094 0.07781954]

The average fitness is:

-90936.92543867834

The Best fitness is:

-90936.9254386783

The weights of the assets in the best portfolio are:

[0.51915644 0.49396094 0.07781954]

Weightage of Gold in your portfolio should be

0.5191564428566038

Weightage of Stock in your portfolio should be

0.49396093891844284

Weightage of Silver in your portfolio should be

0.07781954333813927

As can be seen, all the population members in the final population are alike and hence, the population has converged.

From the 15 trial runs, we get that the expected weight of **Gold should be 0.5133 or** **51.33%.**  
The expected weight of **Stocks should be 0.3702 or** **37.02%**. The expected weight of **Silver should be 0.1702 or 17.02%**

**Discussion**

**2 Assets Case -**

By using gradient based methods on Excel Solver, the optimum distribution of weights is **28.57%** **Stocks** and **71.43%** **Gold**.

There is some error in the results of the genetic evolutionary way from the gradient based method, however this is expected, as it is a probabilistic method. It should be used to get an initial good solution and not an optimum solution. When used in conjunction with a gradient based solver, this initial good solution can be supplied to the gradient based solver to optimize. This teamwork reduces the computation time as the gradient based solver may take a lot of time to optimize in the absence of a good initial guess.

However, one thing that the evolutionary method concludes is that for building a minimum risk portfolio containing stocks and gold, gold should be given higher weightage over stocks.

This conclusion also agrees with the real-world notion that stocks are a risky investment option and that gold is safer. Hence, this can be considered as a small success.

**3 Assets Case –**

By using gradient based methods on Excel Solver, the optimum distribution of weights is **71.42%** **Gold** and **28.57% Stocks** and **0.0000027% Silver.** The overwhelmingly low weightage given to Silver is understandable as the input data of past returns of Silver are much more volatile (risky) than both Gold and Stocks and hence, its contribution to making a minimum risk portfolio should be minimum.

There is some error in the results of the genetic evolutionary way from the gradient based method, however this is expected, as it is a probabilistic method. It should be used to get an initial good solution and not an optimum solution. When used in conjunction with a gradient based solver, this initial good solution can be supplied to the gradient based solver to optimize. This teamwork reduces the computation time as the gradient based solver may take a lot of time to optimize in the absence of a good initial guess.

The results of the evolutionary approach imply one aspect that the weightage of Gold (51.33%) should be the highest, Stocks (37.02%) lower, and Silver (17.02%) the lowest. This is also the qualitative result of the gradient based optimization employed by the Excel Solver.

So, we can say that by performing fairly well even in the case of three assets, our genetic model is working satisfactorily well.

**Conclusion -**

A evolutionary approach combined with a gradient based approach is able to provide us with the optimum weights of assets for a minimum risk portfolio.

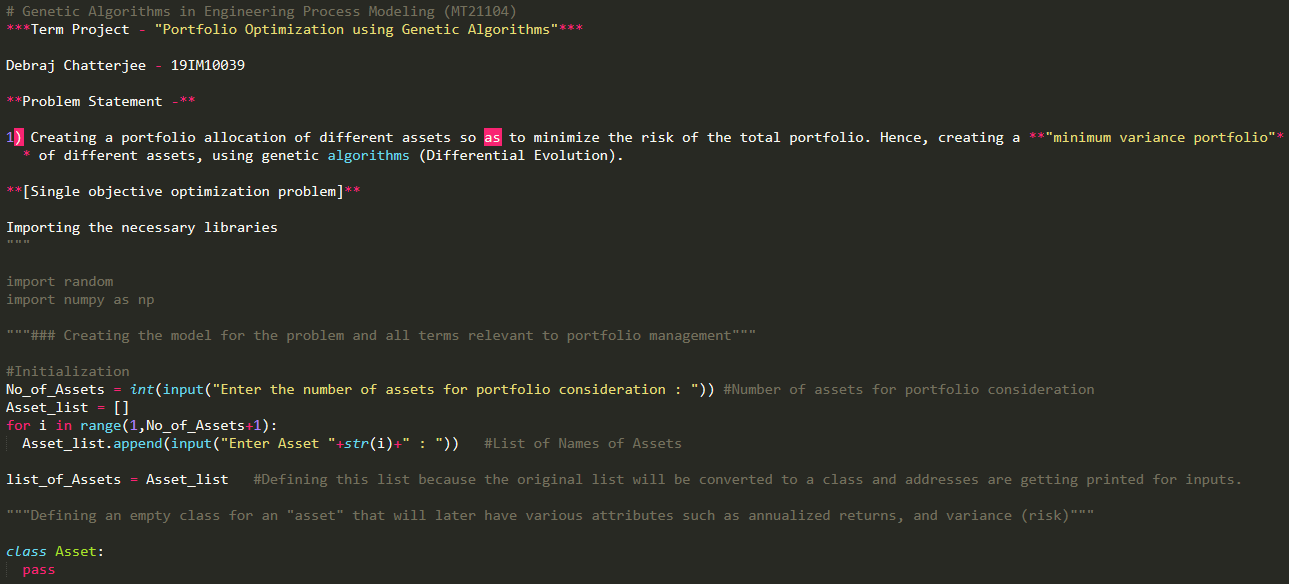
However, even if we only use the evolutionary approach when deciding our portfolio, we will get the priority ranking of the different assets in terms of the weights that should be assigned in the total portfolio, while going for a minimum risk portfolio.   
This is a substantial success for the evolutionary approach that I have applied!

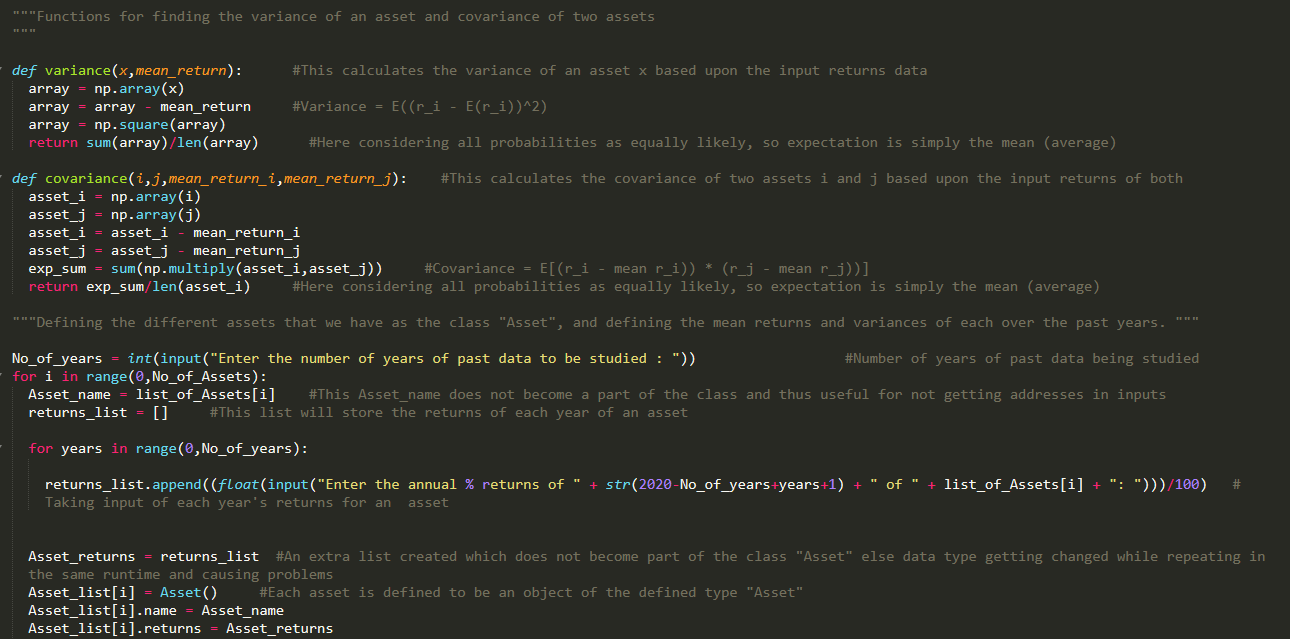
**References**

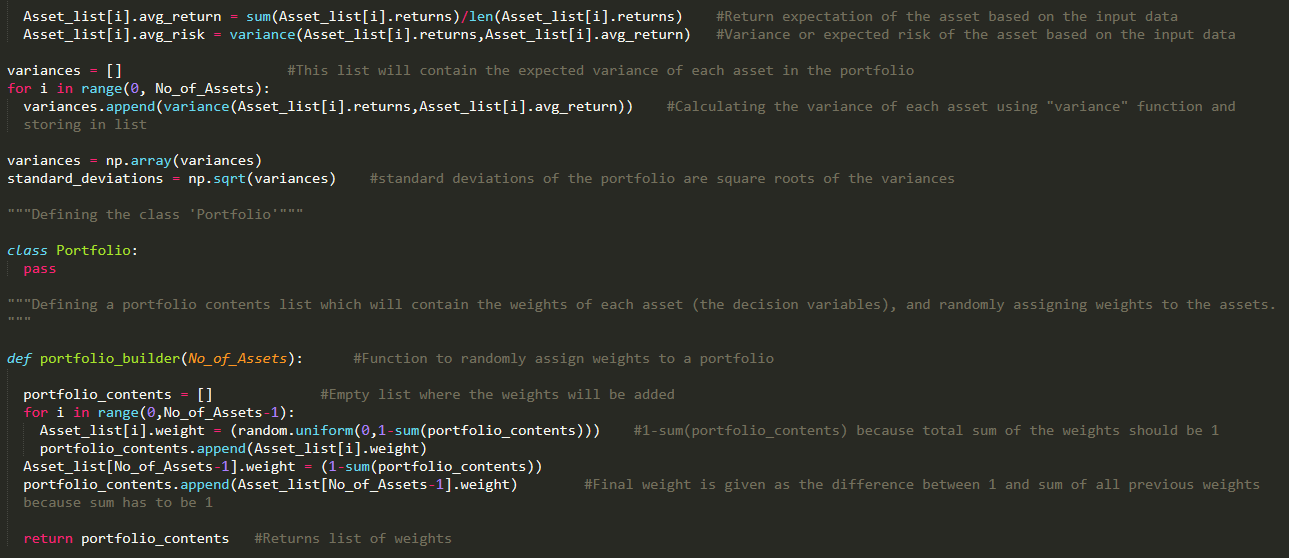
Félix Roudier  
Portfolio Optimization and Genetic Algorithms  
Master’s Thesis Department of Management, Technology and Economics - DMTEC Chair of Entrepreneurial Risks -   
ERSwiss Federal Institute of Technology (ETH) Zurich Ecole Nationale des Ponts et Chauss´ees (ENPC) Paris

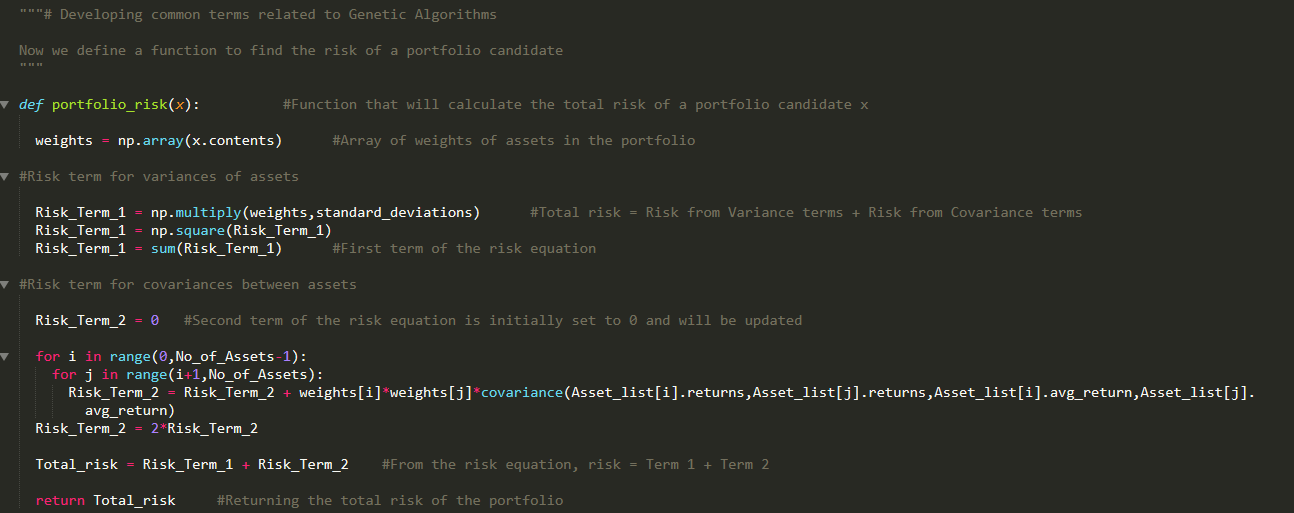
**Appendix**

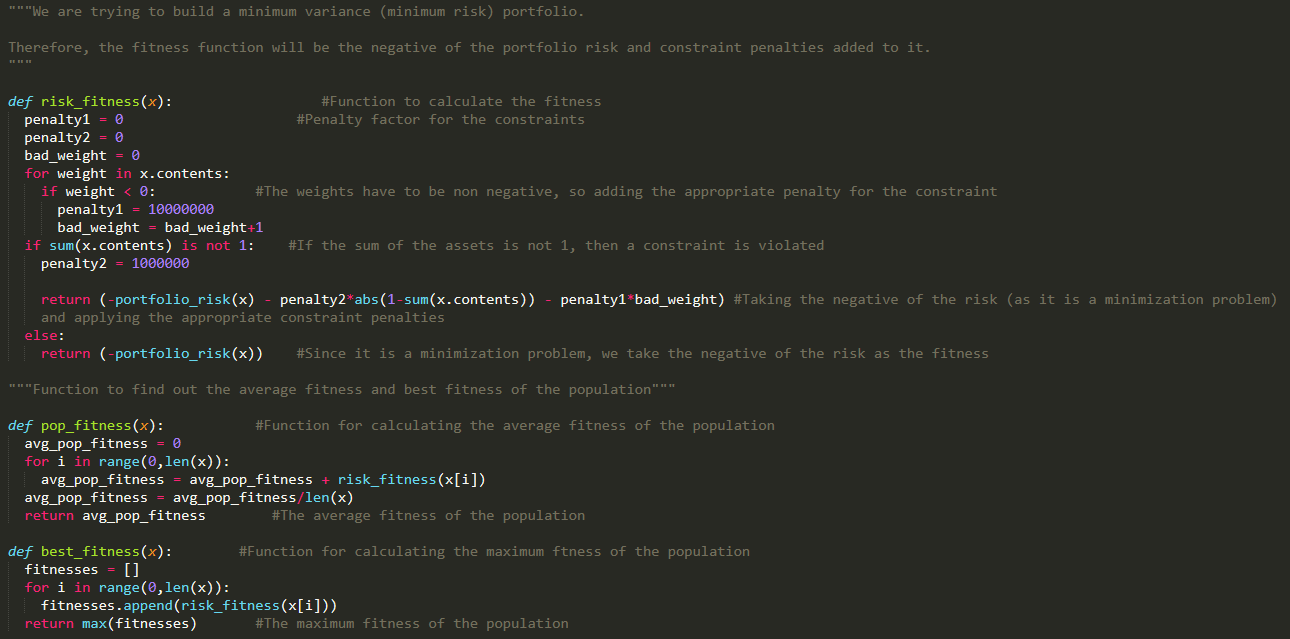
**Python Code for the Project**

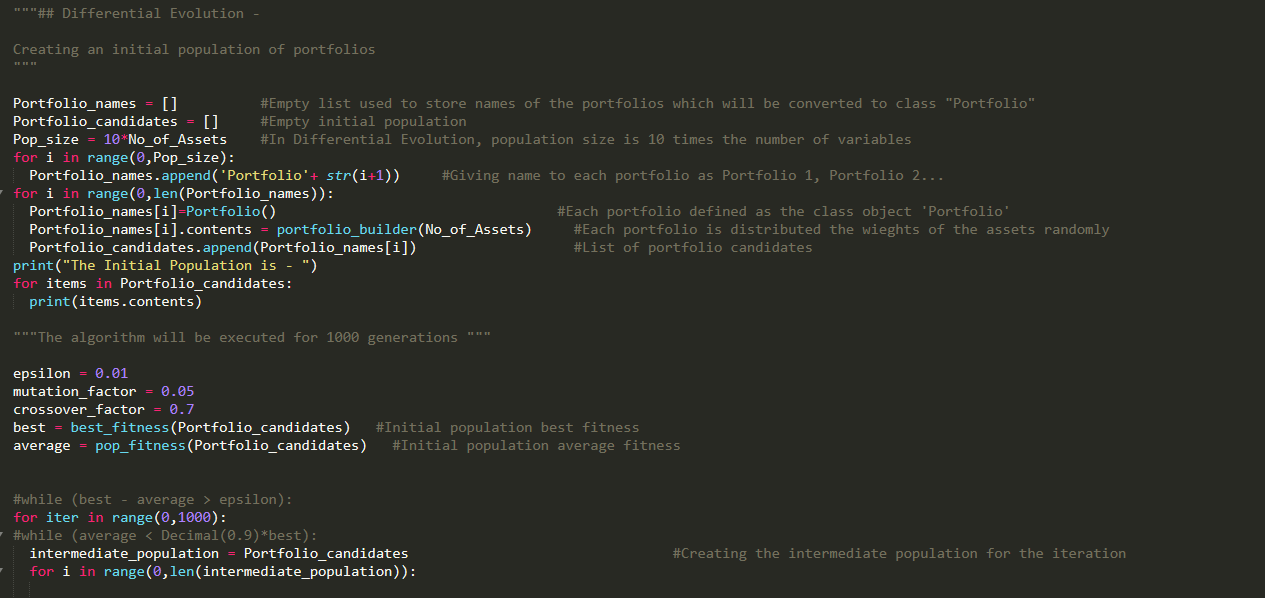
****

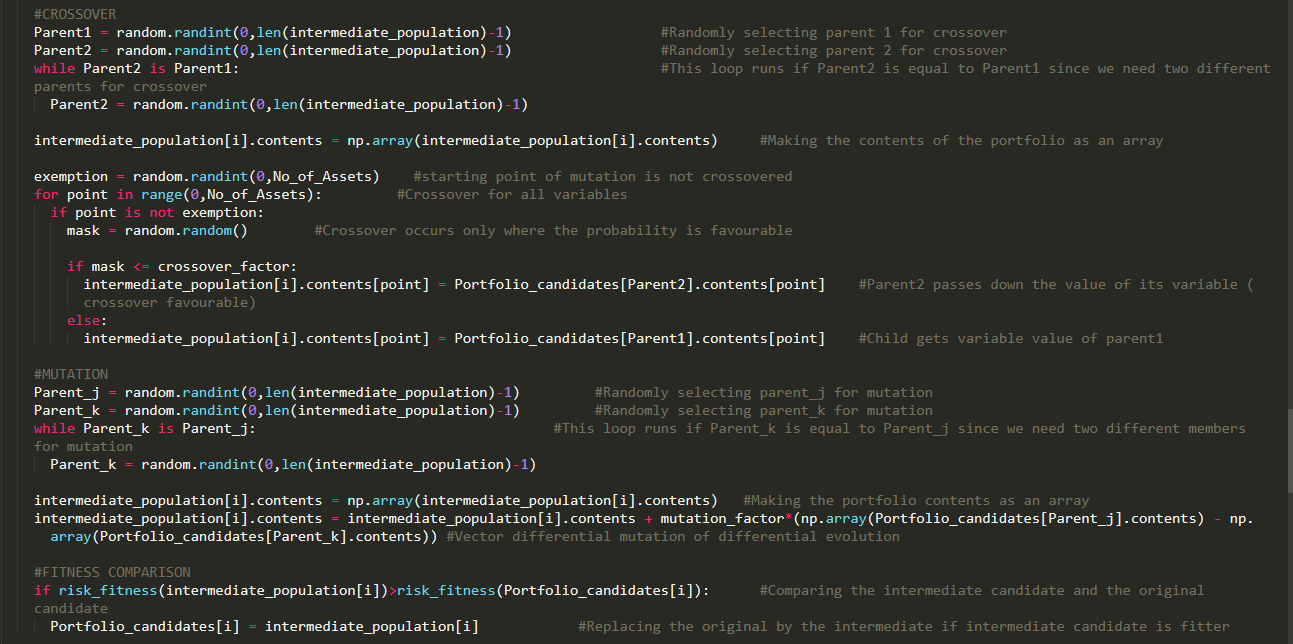
****

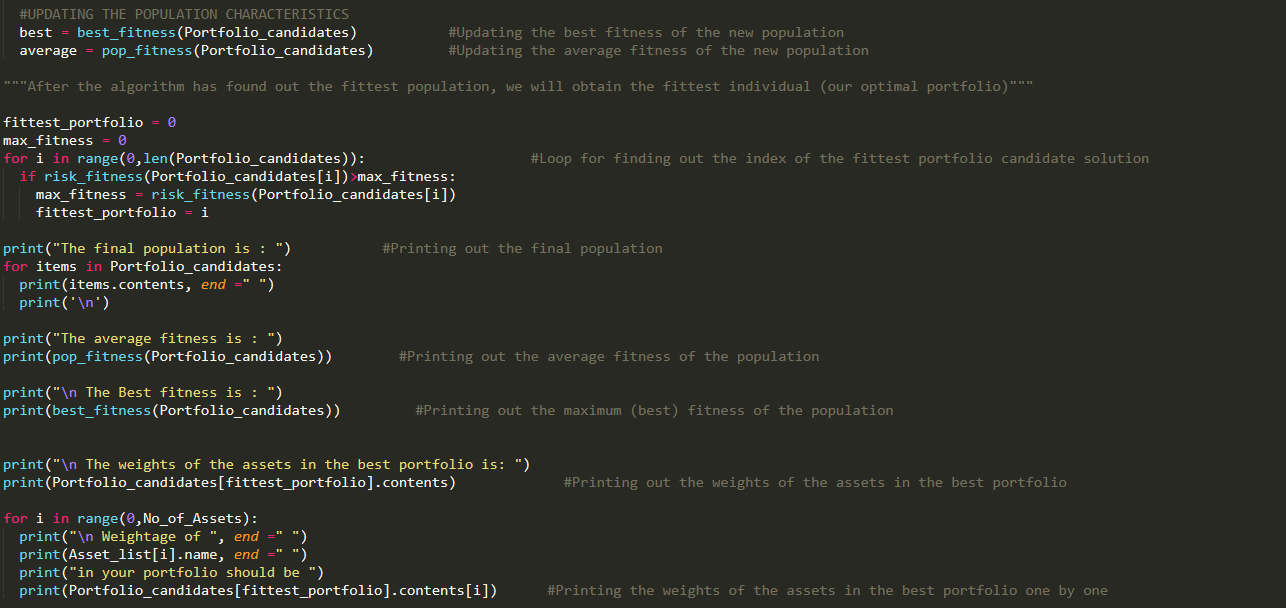
****

****

****

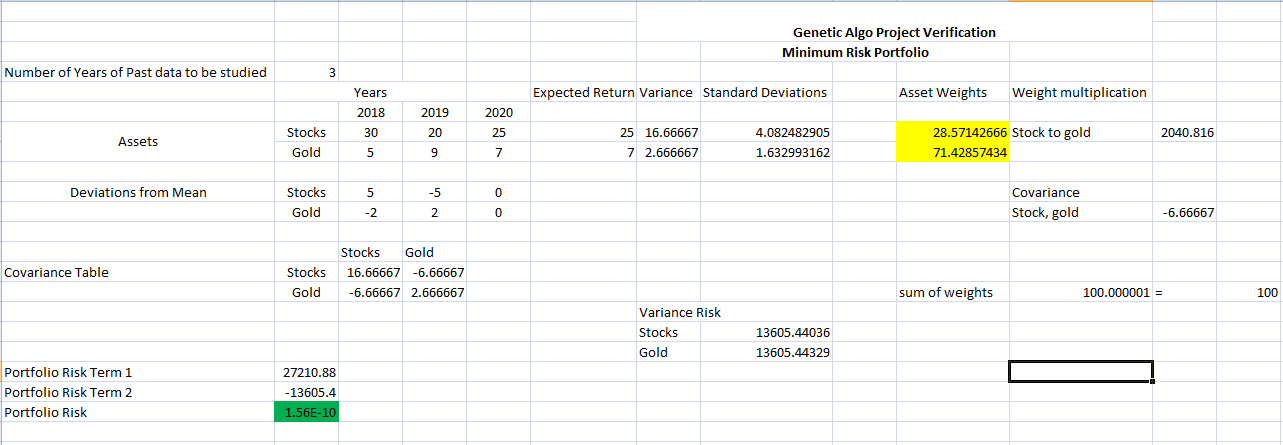
****

****

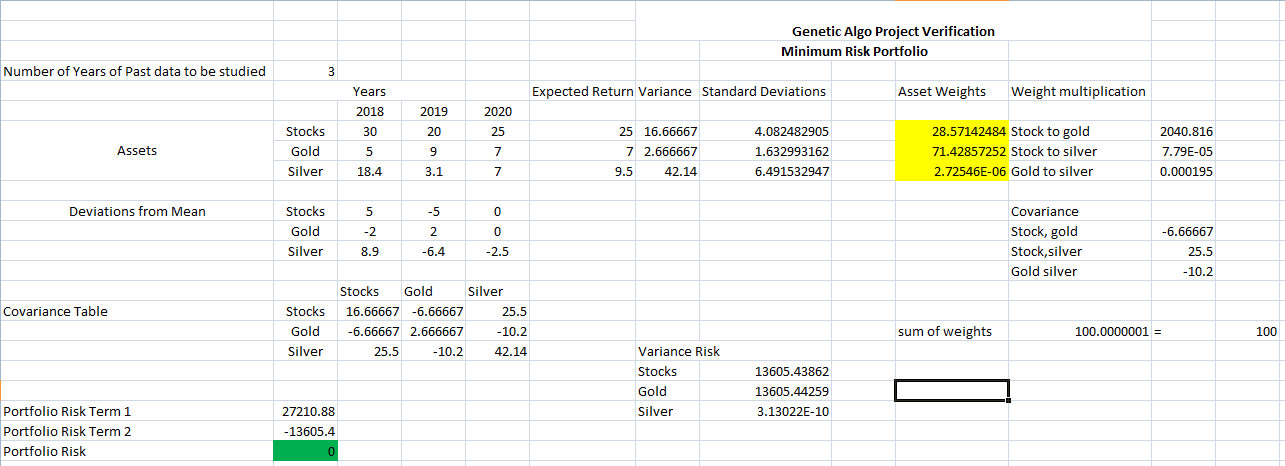
****

**Excel Solver Outputs**

**2 assets -**

****

**3 assets -**

****