Tanzania Journal of Science 47(3): 1236-1242, 2021
ISSN 0856-1761, e-ISSN 2507-7961

High Speed Hill Climbing Algorithm for Portfolio Optimization

Collether John
University of Dar es Salaam, College of Information and Communication Technology, Department of Electronics and Telecommunications, P. O. Box 33335, Dar es Salaam, Tanzania.
E-mail: nyalusicj@yahoo.com; coletha@udsm.ac.tz
Received 27 May 2021, Revised 4 Aug 2021, Accepted 5 Aug 2021, Published Aug 2021
DOI: https://dx.doi.org/10.4314/tjs.v47i3.31

Abstract

Portfolio can be defined as a collection of investments. Portfolio optimization usually is about maximizing expected return and/or minimising risk of a portfolio. The mean-variance model makes simplifying assumptions to solve portfolio optimization problem. Presence of realistic constraints leads to a significant different and complex problem. Also, the optimal solution under realistic constraints cannot always be derived from the solution for the frictionless market. The heuristic algorithms are alternative approaches to solve the extended problem. In this research, a heuristic algorithm is presented and improved for higher efficiency and speed. It is a hill climbing algorithm to tackle the extended portfolio optimization problem. The improved algorithm is Hill Climbing Simple–with Reducing Thresh-hold Percentage, named HC-S-R. It is applied in standard portfolio optimization problem and benchmarked with the quadratic programing method and the Threshold Accepting algorithm, a well-known heuristic algorithm for portfolio optimization problem. The results are also compared with its original algorithm HC-S. HC-S-R proves to be a lot faster than HC-S and TA and more effective and efficient than TA.

Keywords: Portfolio optimization, Hill climbing algorithm, Threshold percentage, Reducing sequence, Threshold Acceptance algorithm.

Introduction

Algorithms for portfolio optimization are used to guide an investor's selection of financial assets. The aim is to achieve maximum return and minimum risk. The pioneering work of Markowitz (1952) introduced the mean–variance optimization as a model for the problem of asset allocation and diversification for maximum return with minimum risk. The allocation is made by considering the trade-off between risk, measured by the variance of the future asset returns, and return. To apply the Markowitz model, strong assumptions on market situations have to be made so as to solve the problem using standard methods like quadratic programming (Sharpe 2000). Introducing constraints of the realistic market like minimum and maximum holding sizes, the standard methods cannot be used to solve the complex problem.

The complex portfolio optimization becomes difficult to solve because the objective functions have multiple local optima, have discontinuities or not well behaved (Dueck and Scheuer 1990, Gilli et al. 2011). On the other hand, to shape the model such that it can be solved by standard methods leads to giving up the relevant aspects of the original problem. Heuristics are able to compute solutions even to problems that are infeasible for standard methods like the extended portfolio optimization problem (Dueck and Winker 1992, Crama and Schyns 2003,

Maringer 2008). They have been shown to be capable of handling non-convex optimization problems with all kinds of constraints (Kirkpatrick et al. 1983, Gilli and Kellezi 2000, Gilli and Schumann 2012). Threshold Accepting algorithm is the main established heuristic algorithm in portfolio optimization (Dueck and Scheuer 1990, Winker and Maringer 2007, Gilli and Winker 2008, Gilli and Schumann 2010). The algorithm proposed below is benchmarked with Threshold Accepting algorithm under standard portfolio optimization problem. The objective of the research is to improve a hill climbing algorithm to a better efficiency.

Objective function

In the standard Markowitz model below, the goal is to maximize the expected return, R, while diminishing incurred risk, σ, (measured as standard deviation/variance) (Markowitz (1952)). Given return ($R_{\mathbf{p}}$) of a portfolio and variance ($\sigma^2_{\mathbf{p}}$) of portfolio, the equation to maximize is

Max $(\lambda.E(R_{\mathbf{p}}) - (1-\lambda).\sigma^2_{\mathbf{p}})$ (1)

Subject to

- Expected return:
  $E(R_{\mathbf{p}}) = \sum_i w_i \, E(R_i)$ (2)
- Portfolio return variance:
  $\sigma^2_{\mathbf{p}} = \sum_i \sum_j w_i w_j \sigma_i \sigma_j \rho_{ij}$ (3)
  $\rho_{ij} = 1$ for $i = j$
- Constraints:
  $\sum_i w_i = 1$ (4)
  $0 \leq w_i \leq 1$ (5)

where the expected return of each asset is $E(R_i)$, each asset variance is σ_i, and each asset weight is w_i.

From the Equation (1), the trade-off between return ($R_{\mathbf{p}}$) and risk ($\sigma_{\mathbf{p}}$) of portfolio is reflected. The efficient line/frontier is then identified by solving the above problem for different values of $\lambda \in (0, 1)$: If $\lambda = 1$ the model will search for the portfolio with highest possible return regardless of the variance. With $\lambda = 0$, the minimum variance portfolio (MVP) will be identified. Higher values of λ put more emphasis on portfolio's expected return and less on its risk (Markowitz (1952). Equations (4) and (5) are the constraints on the weights that they must not exceed certain bounds.

The most important constraints are budget and return constraints since they characterize the main part of the portfolio problem (Di Tollo and Roli 2008). The return constraint is when the investor requires a certain level of profit from his investment with minimum risk. The budget constraint is when the investor has to invest all the money/capital in the portfolio. However, return constraints can only be satisfied for a historical portfolio (Markowitz 1952, Markowitz 1959, Korn 1997, Sharpe 2000, Prigent 2007).

Materials and Methods

To design the method HC-S-R, we have to understand the design of HC-S which is used by HC-S-R.

The Algorithm HC-S

The hill-climbing algorithm is denoted as HC-S. Here HC stands for Hill Climbing, S stands for Simple search of neighbourhood, according to the neighbourhood functions defined below. In each step, it attempts to improve the solution by changing the relative weight of a single asset. ThP stands for Threshold Percentage. It refers to the size of a step in the proposed hill climbing method.

Solution representation

The solution is represented by a vector of numbers ($y_i, \ldots, y_n$). The element in position *i* represents the relative weight of the capital invested in stock *i*. The vector of numbers ($y_i, \ldots, y_n$) are normalized to make sure that the weights in all the assets add up to 1. The percentage/weight to be invested in stock *i* is x_i, where: $x_i = y_i / \sum_{i=1}^{n} y_i$. One advantage of using this representation is that the vector, y, may take any number without violation of budget constraint that the weights add up to 100%.

Neighbourhood definition for the Hill Climbing Algorithm HC-S

HC-S algorithm is proposed for portfolio optimization. The current solution has two neighbours or possible candidate solutions. Elements of vector y in the range of 0 to 100 are randomly generated. The number of elements of y is equal to the number of asset/stocks. The randomly picked position in y is denoted as pos. ThP is a small percentage, which we refer to as *threshold percentage*, by which elements of y will be varied to get the next neighbour. The neighbourhood definition is to pick just one position (pos) in the current solution, y, at random. After picking the random position in the current solution, one neighbour is obtained by adding ThP to that position and another is obtained by subtracting ThP on the same position. This gives two neighbours (two possible candidate solutions) to be compared with the current solution, at random. The first better candidate solution (neighbour) to be picked is taken to be the current solution out of the possible candidate solutions. If no better solution is found, another position, pos, in y is picked at random. The procedure is repeated for a pre-set number of iterations, or until local maximum. Given mean returns of all stocks in column vector denoted as *retasset*, given assets' co-variances/ deviations matrix, denoted as *dev*, and given λ as the level of risk aversion; Figure 1 is the pseudo code for the procedure for HC-S proposed.

Procedure HC-S (ThP, λ, retasset, dev)	
Randomly generate initial current solution y	
Begin	
Repeat	
Pick random position, (pos), in current solution y	
yplus = y	% Generate yplus from current solution
yminus = y	%Generate yminus from current solution
yplus(pos) = yplus(pos)*(1 + ThP)	% Get a neighbour of current solution %
yminus(pos) = yminus(pos)*(1 - ThP)	% Get second neighbour of current solution %
y = **move_to_neighbour**(y, yplus, yminus, λ, retasset, dev)	% Pick a better neighbour solution %
Until stopping criterion	% Stopping criterion; no neighbour is better than current solution or pre-set maximum number of iterations reached%
End	

Figure 1: Procedure of HC-S.

Figure 2 is a Pseudo code for a function for searching for better neighbouring solution. It is applied by the procedure of HC-S of Figure 1.

Function Move_to_neighbour (y, yplus, yminus, λ, retasset, dev)	
Begin	
$x_i = y_i / \sum_{i=1}^{n} y_i$	% Find weights, x, of all the assets n in portfolio%
$xplus_i = yplus_i / \sum_{i=1}^{n} yplus_i$	% Find weights, xplus, of all assets n %
$xminus_i = yminus_i / \sum_{i=1}^{n} yminus_i$	% Find weights, xminus, of all assets n%
xvalue = **objectvalue** (x**,** λ**,** retasset, dev,fitvalue)	% Calculate objective value of portfolio x and denote as xvalue. %
xplusvalue = **objectvalue** (xplus**,** λ**,** retasset, dev, fitvalue)	% Calculate objective value of portfolio xplus and denote as xplusvalue%
xminusvalue = **objectvalue** (xminus, λ**,** retasset, dev, fitvalue)	%Calculate objective value of portfolio xminus and denote as xminusvalue. %
if xplusvalue>xvalue **then** y=yplus	% Return yplus if it is better than y. %
end if	
if xminusvalue>xvalue **then** y=yminus	% Return yminus if it is better than y. %
end if	
return y	
End	

Figure 2: Function to search for better neighbouring solution.

Following, Figure 3 is a Pseudo code for a function to calculate the objective value (Equation (1)). It is applied by the function for searching for better neighbouring solution of Figure 2.

Function Objectvalue (x, λ, retasset, dev, fitvalue)	
Begin	
retpor t= scalar/dot product(retasset, x)	%Calculate effective expected return of portfolio%
risk = x*dev*x'	% Calculate effective risk/variance of portfolio %
fitvalue = λ*retport – (1 - λ)*risk	%Calculate objective/objective value according to equation (1) above. %
return fitvalue	
End	

Figure 3: Function to calculate objective/fitness value.

Design of the Hill Climbing Algorithm HC-S-R

Neighbourhood definition for the Hill Climbing Algorithm HC-S-R

HC-S-R is like HC-S, except that the ThP is reduced over time. In other words, it searches the neighbourhood with finer and finer steps. Elements of vector y are randomly generated. The number of elements of y is equal to number of asset/stocks. The randomly picked position in y is denoted as pos. ThP is a small percentage, which we refer to as threshold percentage, by which elements of y will be varied to get the next neighbour.

Similar to HC-S, the neighbourhood definition of HC-S-R is to pick just one position (pos) in the current solution, y, at random. After picking the random position in the current solution, one neighbour is obtained by adding ThP to that position and another is

obtained by subtracting ThP on the same position. This gives two neighbours (two possible candidate solutions) to be compared with the current solution, at random. The first better candidate solution (neighbour) to be picked is taken to be the current solution out of the possible candidate solutions. If no better solution is found, another position, pos, in y is picked at random. The procedure is repeated for positions picked at random for a pre-set number of iterations, or until local maximum.

In HC-S-R ThP is reduced over time. That is after a pre-set number of iterations or on reaching local maximum, ThP is repeatedly reduced to be half the previous value until it reaches the pre-set minimum ThP value, denoted as minThP. Given mean returns of all stocks in column vector denoted as *retasset*, given assets' co-variances/deviations matrix, denoted as *dev*, and given λ as the level of risk aversion; Fig 4 is the pseudo code for the procedure for algorithm HC- S-R proposed.

Procedure HC-S-R (ThP, minThP, **λ, retasset, dev)**	
Randomly generate initial current solution y set minThP	
Begin	
Do while ThP>minThP	
Repeat	
Pick random position, pos, in curent solution y	
yplus = y	%Generate yplus from current solution
yminus = y	%Generate yminus from current solution
yplus(pos) = yplus(pos)*(1 + ThP)	% Get a neighbour of current solution %
yminus(pos) = yminus(pos)*(1 - ThP)	%Get second neighbour of current solution
y **= move_to_neighbour** (y, yplus, yminus, λ, retasset, dev)	% Pick a better neighbour solution %
Until stopping criterion	% Stopping criterion was: no neighbour is better than current solution or pre-set maximum number of iterations reached%
ThP=ThP/2	
End while	
End	

Figure 4 Procedure of HC-S-R.

The function Move_to_neighbour (y, yplus, yminus, λ, retasset, dev) is the same as that of HC-S above.
The function Objectvalue (x, λ, retasset, dev) is also the same as that of HC-S above.

Results

Benchmarking HC-S, and HC-S-R using Threshold Accepting

HC-S, and HC-S-R are tested on 100 assets portfolio. They are used to solve the Markowitz model, Equations (1), (2), and (3) under basic constraints (4) and (5). The results are compared with Threshold Accepting, which is a well-established Hill Climbing algorithm in portfolio selection and optimization. **HC-S** denotes Hill Climbing-Simple and **HC-S-R**: denotes HC-S with Reducing ThP. **HC-S (9e+5)** is HC-S with 9e+5 iterations, while **HC-S-R (0.1, 0.01, 9e+5)** is HC-S-R with starting ThP = 0.1, half ThP every 9e+5 iterations, until ThP is below 0.01. The above number of iterations was given on every ThP, but the program was to stop on reaching a local optimum.

Table 1 shows experimental results on the portfolio optimization on 100 stocks from DAX stock exchange; taken after 100 runs. The

results show the values of objective function, number of functional evaluations required to reach final objective value, and average time in seconds for one run to converge to local maximum (final solution). The best final objective value is the highest objective function value obtained in all the 100 runs. Final objective values obtained in each run were recorded and so below are the mean, standard deviation (STD) and worst of final objective values in all the 100 runs. The mean and STD of number of functional evaluations to reach final objective value, of the 100 runs, are also given.

Table 1: Experimental results on portfolio optimization on 100 stocks, after 100 runs

Algorithm		**HC-S-R** (0.1, 0.01, 9e+5)	**HC-S** (9e+5)	**TA** (9e+5)
Best final objective value		0.000596	0.000596	0.000588
Final objective value	Mean	0.000594	0.000594	0.000563
	STD	7.32e-6	6.46e-6	3.46e-5
	Worst	0.000572	0.000559	7.2563e-5
No. of functional evaluations to final objective value	Mean	3.2e+4	2.7e+5	3.0e+5
	STD	850	6800	1770
Average time for 1 run (in sec.)		10.84	39.0	704.7

STD = Standard deviation.

Discussion

The best final objective values are higher and similar in HC-S-R, and HC-S showing that the methods are more robust than Threshold Accepting as they better escape local optima. The effect of "R" in HC-S-R is seen in speed to reach final objective value. This means, repeatedly reducing ThP (instead of fixing ThP) made the algorithm more efficient. The mean of number of functional evaluations for HC-S-R is 3.2e4 while that of HC-S is 2.7e5. Average time to converge to final objective value for HC-S-R was 10.84 seconds while that of HC-S was 65 seconds. Of the algorithms, HC-S-R required less number of functional evaluations to final objective as the mean and STD of number of functional evaluations to final objective of HC-S-R (3.2e4, 850) was the least of all. The time elapsed for HC-S-R to finish the 1 run was the smallest, making it the fastest of the algorithms. So HC-S-R is more efficient than HC-S.

The mean of final objective value of HC-S-R (0.000594) is the same as that of HC-S (0.000594). This indicates that HC-S-R and HC-S are similar in effectiveness to find better solutions.

The results on benchmarking HC-S-R and HC-S with TA are summarized in Table 2.

Table 2: Summary on benchmarking the algorithms with TA

Algorithm	**Effectiveness**	**Efficiency**
TA	Well established algorithm in portfolio optimization	Efficient
HC-S	More effective in finding better solution than **TA**	More efficient and quite faster time wise than **TA**
HC-S-R	Similar with **HC-S** in effectiveness to find better solution.	More efficient than **HC-S**

Conclusion

HC-S-R and HC-S have been described and implemented in portfolio optimization problem. Results show that the technique of reducing ThP made HC-S-R more efficient than HC-S. Also the small standard deviations observed in final objective values show that HC-S, and HC-S-R find solutions more robust than Threshold Accepting. Results also demonstrate that HC-S-R and HC-S manage to find significantly better solutions than Threshold Accepting, an established algorithm for portfolio optimization. It is therefore recommended for techniques to be used to increase the efficiency of other algorithms. Also the hill climbing algorithms produced can be combined with other algorithms like evolutionary algorithms, to give hybrid algorithms for better performance.

Acknowledgment

I thank my employer; University of Dar es Salaam for all support and encouragement.

References

Crama Y and Schyns M 2003 Simulated annealing for complex portfolio selection problems. *Eur. J. Oper. Res.* 150: 546–571.

Di Tollo G and Roli A 2008 Meta-heuristics for the portfolio selection problem. *Int. J. Oper. Res.* 5(1): 13-35.

Dueck G and Scheuer T 1990 Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing. *J. Comput. Phys.* 90 (1): 161-175.

Dueck G and Winker P 1992 New concepts and algorithms for portfolio choice. *Appl. Stoch. Models Data Anal.* 8: 159-178.

Gilli M and Kellezi E 2000 Heuristic Approaches for portfolio optimization. In proceedings of the *6th International. Conference on Computing and Economics* Barcelona Spain.

Gilli M and Winker P 2008 A review of heuristic optimization methods in econometrics. *Swiss Finance Institute Research* Paper No. 08-12.

Gilli M and Schumann E 2010 Portfolio optimization with threshold accepting: a practical guide In: Satchell SE (Ed) *Optimizing Optimization: The Next Generation of Optimization Applications and Theory*. Chapter 9, Elsevier.

Gilli M Maringer D and Schumann E 2011 Numerical methods and optimization in finance. Elsevier Inc. Chapters 12-13.

Gilli M and Schumann E 2012 Heuristic optimisation in financial modelling *Ann. Oper. Res.* 193(1): 129-158.

Kirkpatrick S, Gelatt Jr CD and Vecchi P 1983 Optimization by simulated annealing. *Science* 220 (4598): 671-680.

Korn R 1997 Optimal portfolios: stochastic models for optimal investment and risk management in continuous time. World Scientific Publishing Co. Pre. Ltd.

Maringer D 2008 Heuristic optimization for portfolio management *IEEE Comput. Intell. Mag.* 3(4): 31-34.

Markowitz H 1952 Portfolio selection. *J. Finance* 7(1): 77-91.

Markowitz H 1959 Portfolio selection: efficient diversification of investments. John Wiley and Sons New York City.

Prigent J 2007 Portfolio optimization and performance analysis Chapman & Hall/ CRC Press. New York.

Sharpe WF 2000 Portfolio theory and capital markets McGraw-Hill New York.

Winker P and Maringer D 2007 The threshold accepting optimisation algorithm in economics and statistics. In: Kontoghiorges EJ and Gatu C (Eds.) Optimisation, Econometric and Financial Analysis *Advances in Computational Management Science* Vol 9 (pp. 107-125). Springer, Berlin.