Article can be accessed online at http://www.publishingindia.com

A Cuckoo Optimisation Algorithm for Solving Financial Portfolio Problem

Slimane Sefiane*, Hadj Bourouba**

Abstract

Over the years, different solution methods to financial portfolio optimisation problems have been developed and applied. In recent years, however, there has been an increasing use of heuristic methods as alternative to other methods. In this study, a newly developed heuristic method called Cuckoo Optimisation Algorithm (COA) is presented to solve financial portfolio optimisation problems. The results on a five stock application example show that the proposed cuckoo algorithm solves the portfolio optimisation problem more optimally than genetic algorithm and ant colony algorithm.

Keywords: Financial Portfolio Optimisation, Approximate Methods, Cuckoo Optimisation Algorithm

Introduction

Financial portfolio optimisation can be viewed in this work as the process of identifying the combination of assets to invest so as to satisfy two main objectives simultaneously: maximisation of return and minimisation of risk subject to different constraints. Efforts have been made on developing methods for solving optimisation problems including financial portfolio optimisation problems. These methods can be classified into two categories: exact optimisation methods and heuristic/ approximate optimisation methods. While exact methods provide exact results, these methods have been successfully applied to problems only when both the objective function and the constraints are linear. However, when taking into account real life conditions (i.e., cardinally and transaction costs constraints, market dynamic), financial portfolio optimisation becomes a problem in which the objective function and/ or the constraints are nonlinear and cannot be solved by the use of exact methods. Brabazon and O'Neil (2008), for example, comment that exact methods (also called analytical methods) are largely based on a reduction of the constraints and their effectiveness is limited to simple scenarios of the problem, an extension of the problem by introducing more complex risk measures. Loosening several artificial assumptions requires a new efficient approach, which cannot be developed on the basis of exact methods due to the irregularity of the objective function and the search space. Harlow (1991) criticised exact methods as they are based on artificial assumptions and fail to encompass more complex, real life problems. Kochenberger, Glover, Alidaee, and Rsgo (2004) comment that exact methods degrade rapidly with problem size, and have meaningful application to general unconstrained quadratic program problems with no more than 100 variables. For larger problems, approximation methods are usually required. In the same way, Puchinger and Raidl (2005) state that while exact methods focus on finding an optimal solution for every instance of a combinatorial optimisation problem, the computational time, however, often increases dramatically with the instance size, and often only small or moderately-sized instances can be practically solved. In this case, the only possibility for larger instances is to trade optimality for computational time, yielding heuristic algorithms. Abido and Elazouni (2011) argue that approximate methods can be successfully applied to a wide range of optimisation problems with little or no modifications in order to adapt to any specific problem. They add that these approximate methods are usually easier to implement than the exact methods. Elham and Bijari (2014) argue that applying

* Institute of Economic and Commercial Sciences, University (Center) of Relizane, Algeria. Email: ssefiane@hotmail.com
** Faculty of Economic and Commercial Sciences, University of Mostaganem, Algeria. Email: hadjbourouba@yahoo.fr

exact methods in finding global solutions to continuous problems necessitate their linearisation which is too difficult to practice. Heuristic methods have been tackling the problems within reasonable computational time. Simon (2008) stressed that approximation methods are becoming popular tools for modeling uncertainty and reducing the computational expense.

In view of the argument against exact methods, it is therefore justified to see the use of approximate methods to optimisation problems as an alternative to exact methods being increasing recently. The use of approximate methods such as genetic and ant colony optimisation algorithms in solving financial portfolio optimisation problem for example is preferred over exact methods (see for example, Drezewski & Siwik, 2010; Hochreiter, 2013; Lipinski, Winezura, & Wojcik, 2007; Coello, Lamont, & Vanveldhuizen, 2007; Senel, Pamukcu, & Yanik, 2008; Anagrostopolos & Manimas, 2011; Taimoor & Shoemaker, 2016).

This paper uses a newly developed evolutionary approximate method inspired by nature called Cuckoo Optimisation Algorithm (COA) to solve the portfolio optimisation problem. Because this method is still emerging in the field of finance, the present study constitutes a further development to the existing literature in this field, namely the problem of portfolio optimisation, and provides a framework for future researches.

The Optimisation Algorithm by Cuckoo

The cuckoo optimisation algorithm (COA) is a novel meta-heuristic approximate algorithm that was developed in 2011 by Rajabioun (2011) and it has been used to solve a range of real life optimisation problems.

Cuckoo optimisation algorithm is a nature inspired method based on the breeding and the eggs laying behaviour of a bird family called "cuckoos" and simulating this behaviour for solving optimisation problems. Rajabioum (2011) tested the performance of this algorithm via a benchmarking study, and found that this algorithm has high speed of convergence and reaches the global solution easily. He added that the COA is superior than other nature inspired computing algorithms because of its multiple functions operators (including egg laying and immigration operators). Other optimisation algorithms have the operators which are defined for one specific objective, but in COA, defined operators do multiple functions simultaneously. The study by Burnwal and Deb (2013) on the job scheduling optimisation problem of flexible manufacturing system showed that the CS algorithm is better than genetic algorithm and particle optimisation swarm methods in terms of efficiency. Bhandari, Singh, Kumar, and Singh (2014) argue that CS is capable of obtaining the optimised solution in case of complicated problems. Hence, CS may be helpful for non-linear problems and multi-objective optimisations. The cuckoo search has been proven to deliver excellent performance in function optimisation, engineering design, neural network training, and other continuous target optimisation problems and has solved the knapsack and nurse-scheduling problems (Abusrhan and Al Daoud, 2013). In comparison to other evolutionary algorithms; Xing and Gao (2014) revealed that cuckoo search algorithm is very easy to implement as it dependents only on a relatively small number of parameters to define and determine the algorithm's performance.

In the COA algorithm, a given solution is called "habitat". Habitat represents a current position of the cuckoo in the search space. Each cuckoo habitat produced a number of eggs; some of these eggs grow and become mature cuckoos. Laying of eggs is based on the current position of the habitat of the cuckoo and on the reproduction distance proportional to the total number of eggs of the cuckoo itself with two other variables limiting the interval where the values of the variables are drawn (denoted Varhi and Varmax).

At each iteration, each cuckoo lays a set of eggs; some of them survive and migrate to seek the best habitat. The rest will be killed by their hosts (the owner of the nest bird), or destroyed because of their bad qualities.

Optimisation Steps with Cuckoo Algorithm

The cuckoo optimisation algorithm (COA) is a population-based method which begins with an initial population of cuckoos. These cuckoos lay some eggs in the nest of another species "habitat". A random group of potential solutions is generated that represent the habitat in the COA. In the fitness function will be evaluated the parameters of the candidate components. The steps to search the optimum solution are given as follows:

Step 1: Generating initial cuckoo habitat

COA like genetic algorithm starts usually with initial population that includes values of problem variables that formed as an array that is called "habitat" and it is analogous to `chromosome' in the genetic algorithm. In an **N**var-dimensional optimisation problem a habitat is an array of 1 × **N**var and is defined as follows.

Habitat = [x1; x2; x3; : : : ;xNvar]

Each of the variable values (x1, x2, x3, . . . ,xNvar) is a real number. One habitat in COA is one solution. In financial portfolio optimisation, one habit corresponds to one candidate solution consisting of invested assets.

The profit of a habitat is obtained by evaluation of profit function *fp* at a habitat of {X1, X2, XNvar}.

Step 2: Egg Laying Radius (ELR)

This step consists of dedicating some eggs to each cuckoo and initialises their Egg Laying Radius (ELR).

In an optimisation problem with upper limit of Varhi and lower limit of Varlow for variables, each cuckoo has an egg laying radius (ELR) which is proportional to the total number of eggs, number of current cuckoo's eggs and also variable limits of Varhi and Varlow. So, ELR is defined as:

ELR = αx

$$\frac{\text{Number of current cuckoo's eggs}}{\text{Total number of eggs}} \text{ x } (\text{Var}\mathit{hi} - \text{Var}\mathit{low})$$

Step 3: Cuckoo's egg laying

Each cuckoo starts laying eggs randomly in some other host bird's nests within her ELR. This egg laying process is analogous to generation of new children in genetic algorithm and determination of new particle positions in PSO Slingh and Rattan (2014). In financial portfolio optimisation problem, this corresponds to generation of new candidate solutions with random changes in variables depending upon ELR. After all the cuckoos have laid their eggs, some of them that are less similar to host bird's own eggs are detected by host birds and thrown out of their nests. Those with least profit values are eliminated. The profit value of a habitat is given by:

Profit = *fp* (habitat) = *fp* {x1, x2, …..xNvar}. This calculation of profit values corresponds to the objective function (i.e., fitness function) of the portfolio optimisation problem.

Step 4: Cuckoos immigration

After the cuckoo groups are formed in different areas, the society with best profit value is selected as the goal point for other cuckoos to immigrate. When mature cuckoos live in all over the environment it's difficult to recognise which cuckoo belongs to which group. To solve this problem the mean profit value is calculated for each grouping of cuckoos. Then the maximum value is calculated; and the maximum value is calculated to determine the maximum of these mean profits to find out the goal group and consequently that group's best habitat is the new destination habitat for immigrant cuckoos.

Step 5: Eliminating cuckoos in worst habitats

Only those Nmax number of cuckoos survive that have better profit values, other demise

Step 6: Convergence

The optimisation process of cuckoo algorithm is terminated when more than 95% of all cuckoos converge to the same habitat.

Rajabioun (2011) summarised the COA algorithm as follows:

1. Initialise cuckoo habitats with some random points on the profit junction.
2. Dedicate some eggs to each cuckoo.
3. Define ELRJbr each cuckoo.
4. Let cuckoos to lay eggs inside their corresponding ELR.
5. Kill those eggs that are recognised by host birds.
6. Let eggs hatch and chicks grow.
7. Evaluate the habitat of each newly grown cuckoo.
8. Limit cuckoos' maximum number in environment and kill those who have in worst habitats.
9. Cluster cuckoos, find best group and select goal habitat.
10. Let new cuckoo population immigrate toward goal habitat.

11. If stop condition is satisfied stop, if not go to 2.

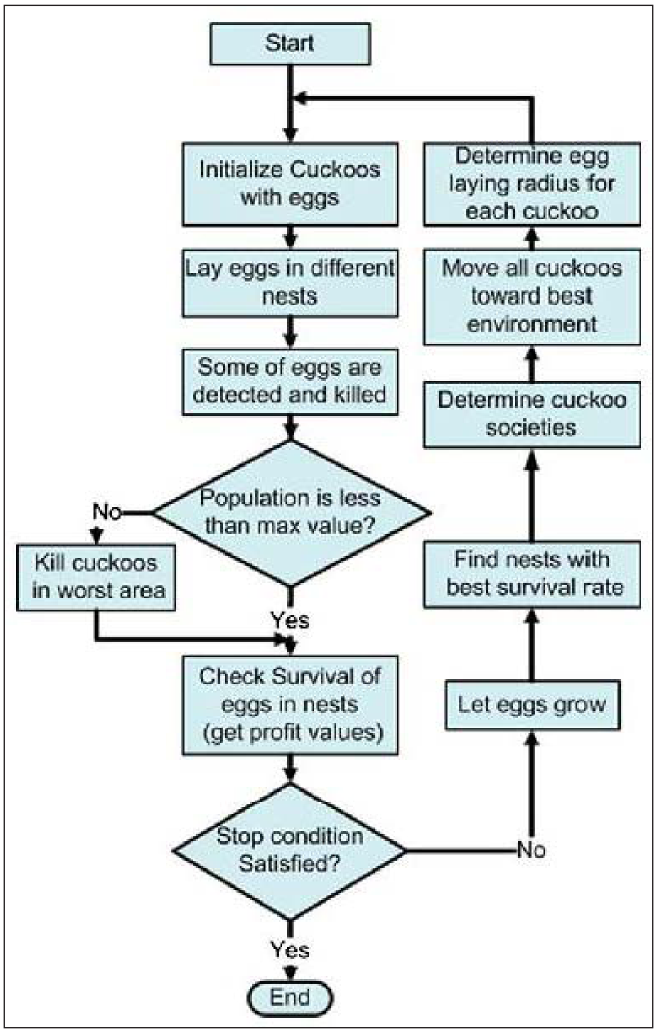

Fig. 1: Flowchart of Cuckoo Optimisation Algorithm

The Mathematical Formulation of the Portfolio Optimisation Problem

The objective of portfolio optimisation is to find out the portfolio weights invested in each asset as to maximise the portfolio return and to minimise the portfolio risk and at same time fulfilling the constraints. The mathematical formulation of the portfolio optimisation problem is based on the following equations:

$$E(w_i) = w_i r_i \qquad \text{...Eq. 1}$$

where

wi: denotes the weight of the individual asset

ri: denotes the expect return of asset *i*

The total expected return of portfolio *r* to be maximized can be written as

$$R\rho = \sum_{i=1}^{n} E(wi) \qquad \text{...Eq.2}$$

where

n: is the number of invested assets

$$\text{Max} \left\{ R\rho = \sum_{i=1}^{n} E(wi) \right\} \qquad \text{...Eq.3}$$

Under the following constraints:

$$\sum_{i=1}^{n} w_i = 1$$

$$W_i^{\min} \le W_i \le W_i^{\max}$$

For a positive portfolio return (what so ever are the weights values) let: $\sum_{i=1}^{n} r_i w_i \ge 0$

where

$Wi^{\max}$ and $Wi^{\min}$: maximum and minimum weights of asset ***i***

The portfolio variance ($\sigma^2\rho$) can be written as:

$$\sigma^2\rho = \sum_{i=1}^{n} (W_i^3 s^2(r_i)) + \sum_{i=1}^{n} \sum_{j=i+1}^{n} 2W_i W_j \operatorname{cov}(r_i, r_j) \qquad \text{... Eq.4}$$

where

σ^2 *(ri)*: variance of asset ***i***

Cov (ri, rj): Covariance between asset ***i*** and asset ***j***

Financial Portfolio Optimisation Using the COA Method

The goal of the portfolio optimisation problem in this work consists of selecting the portfolio weights invested in each asset so as to maximise the portfolio return and minimise the portfolio risk simultaneously. To meet this condition, the multi-objective function (the fitness function) to minimise is presented as an algebraic summation of the portfolio variance and the portfolio return, where the least value of the objective function defines the best solution (i.e., best portfolio). In addition to this, a penalty function method is used to handle the constraints.

The multi-objective function; which is the total function to minimise therefore, can be expressed as follows:

$TF\rho = \sigma^2 - R\rho$ ... Eq. 5

The penalty function that is used to handle the constraints also needs to be minimised (i.e., the lower the value of the penalty function, the better the performance).

Pf (wi, rk) = TF

$$TF\rho \frac{1}{\mu k} \sum_{j=1}^{n} [hj(wi)]^2 + \mu k \sum_{j=1}^{m} \left[\frac{1}{gi(wi)} \right]^2 + \quad \text{... Eq. 6}$$

where μk is the penalty coefficient.

gi(wi) > 0 i= 0, …. m

hj(wi) = 0 j= 0, ….n

Table 1: The Simulation Example of Five (05) Stock Portfolios

Year	*Stock 1*	*Stock 2*	*Stock 3*	*Stock 4*	*Stock 5*
2007	-.15	.29	.38	.18	-.10
2008	.05	.18	.63	-.12	.15
2009	-.43	.24	.46	.42	.15
2010	.79	.25	.36	.24	.10
2011	.32	.17	-.57	.30	-.25

The mean return for each asset and the covariance matrix are given in the tables 2 and 3 below:

Table 2: The Mean Returns for Each Asset

	Stock 1	*Stock 2*	*Stock 3*	*Stock 4*	*Stock 5*
Mean return (*ri*)	.116	.226	.252	.204	.11

Table 3: The Covariance Matrix

	Stock 1	*Stock 2*	*Stock 3*	*Stock 4*	*Stock 5*
Stock 1	.21728	-.003376	-.053492	-.009264	.01064
Stock 2	-.003376	.00253	.008468	.002376	-.00456
Stock 3	-.053492	.008468	.22247	-.031128	-.02392
Stock 4	-.009264	.002376	-.031128	.04068	.00276
Stock 5	.01064	-.00456	-.02392	.00276	.01675

Table 4: COA Parameters Setting

COA parameters	*Values*
Number of initial population/cuckoos (habitat size)	5
Minimum number of eggs for each cuckoo	2
Maximum number of eggs for each cuckoo	4
Maximum number of cuckoos	10
Maximum iterations of the algorithm	100
Control parameter of egg laying (radius coefficient)	4
Lambda variable in COA (motion coefficient)	2
Number of clusters	1
Population variance that cuts the optimisation	1e-103

Comparison of the Simulation Results

The COA algorithm is used to optimise a financial portfolio problem with 5 stock example (i.e.;, n= 5) for the same objective function and parameters as in the previous two studies that have already been dealt with by using genetic algorithm and ant colony algorithm (Sefiane & Benbouziane, 2013). The constraints are the same. The COA has been applied using MATLAB, and the results in the table below report the optimal performance.

Table 5: The Optimal Results Obtained by the COA

The Portfolio Weights	*The Computational Results*
W1 : .05	The optimum portfolio return : .22527
W2 : .32501	The portfolio variance (optimum risk/ least risk) : .049348
W3 : .48396	The objective function : 4.4885
W4 : .090977	The penalty function : 4.4912
W5 : .05	The execution time : 7.389

The number of initial population is set 50 for COA and GA algorithms and the number of iterations is set 100 for both algorithms. As can be seen the COA reaches the highest return, of course with a slightly higher risk than that of ACO algorithm, because the higher the return the higher the risk.

Table 6: Comparison of the Results Found by the Three Algorithms

Parameters/Algorithm	*COA*	*AOA*	*GA*
The portfolio optimal return	.22527	.2229	.2213
The portfolio variance	.04934	.0311	.0801
The objective function	4.4885	4.4863	4.5984
The execution time	7.389 sec.	4.0142 sec.	4.0170 sec.

A comparison of finding the optimal portfolio return of these three algorithms shows that the proposed algorithm (COA) solves the portfolio optimisation problem more efficiently, but a major disadvantage lies in the time execution that is larger (i.e., 7.389 sec.). These results therefore, support the study by Malik and Tayal (2014) that suggest Cuckoo search algorithm gives larger simulation time for small number of N(N in this study refers to the number of invested assets), and remains almost constant when the number of N increases. Unlike GA and ACO algorithms, however, where the simulation time increases with increase in N.

It should be recognised, however, that different users using the same algorithm may obtain different results in response to the same problem because, it all dependents on the problem modeling and on how the algorithm's performance is tested. The algorithm's performance results may differ from a problem situation to another because there is no single way to test the performance of an algorithm. The art of testing an algorithm's performance can influence the results. The performance of genetic algorithm for example can be tested in many and different ways as shown in Figs. 2 to 5.

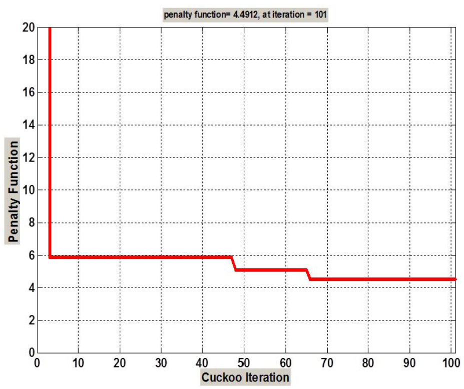

Fig. 2: The penalty function

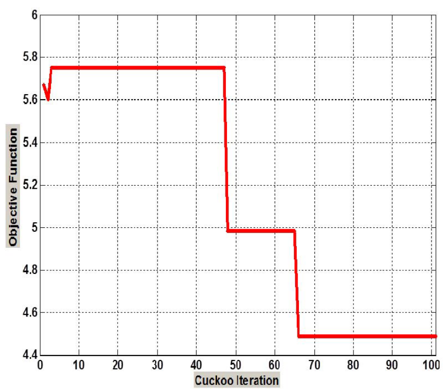

Fig. 3: The Objective Function

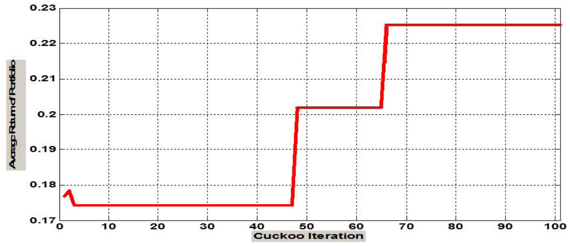

Fig. 4: The Portfolio Average Return

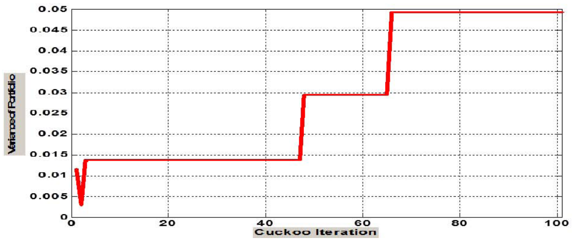

Fig. 5: The Portfolio Variance

Conclusion

In this paper, a method by cuckoo optimisation algorithm (COA) to solve financial portfolio optimisation problem is presented. The algorithm is used to find out the portfolio weights invested in each asset so as to maximise the portfolio return and minimise the portfolio risk and at the same time keeping the problem constraints within their limits. Hence, the objective of the optimisation problem using the COA method was set in this work to search for the portfolio return and the portfolio risk that minimise both the fitness function (i.e., the objective function) and the penalty function. The results obtained show that the proposed cuckoo optimisation algorithm outperforms the other algorithms such as genetic algorithm and ant colony optimisation algorithm in terms of accuracy, efficiency and simplicity.

Although the proposed algorithm is applied on a very simple example, it points to the successful application of the model to financial portfolio optimisation problem.

Finally, further studies on financial portfolio optimisation problems might need to be undertaken to prove the practical efficiency of the cuckoo optimization algorithm.

References

Abido, M. A., & Elazouni, A. M. (2011). Multiobjective evolutionary finance-based scheduling: Entire projects' portfolio. *Journal of Computing in Civil Engeneering,* 25(1).

Abusrhan, A., & Daoud, A. (2013). A hybrid Algorithm Using a genetic algorithm and cuckoo search algorithm to solve the salesman problem and its application to multiple sequence alignment. *International Journal of Advanced Science and Technology,* 61, 29-38.

Anagnostopoulos, K. P., & Mamanis, G. (2012). Multiobjective evolutionary algorithms for complex portfolio optimization problems. *Computational Management Science,* 8.

Bhandari, A. K., Singh, V. K., Kumar, A., & Singh, G. K. (2014). Cuckoo search and wind driven optimation based study of satellite image segmentation for multilevel thresholding using Kapur's entropy. *Expert Systems with Applications,* 41.

Xing, B., & Gao, W. J. (2014). *Innovative computational intelligence: A rough guide to 134 clever algorithms.* Springer International Publishing, Switzerland.

Brabazon, A., & O'Neil, M. (2008). *Natural computing in computational finance.* Springer-Verlarg Berlin Heidelberg.

Burnwal. S., & Deb, S. (2013). Scheduling optimization of flexible manufacturing system using cuckoo search-based approach. *International Journal of Advanced Manufacturing Technology,* 64*(*5-8).

Coello, C. G., Lamont, B., & Vanveldhuizen, D. A. (2007). *Evolutionary algorithms for solving multi-objective problems* (2nd Ed.). Springer Science and Business Media. LLC.

Drezewski, R., & Siwik, L. (2010). Co-evolutionary multi-agent system for portfolio optimization. *Studies in Computational Intelligence,* 100.

Elham, S., & Bijari, M. (2014). Evaluation of the efficiency of cuckoo optimization algorithm. *International Journal on Computational Sciences & Applications (IJCSA),* 4(2).

Harlow, H. V. (1991). Asset allocation in a downside-risk framework. *Financial Analysts Journal,* 30-40.

Hochreiter, R. (2013). Solving dynamic optimization problems with revolutionary algorithms. *International Journal of Innovative Computing and Applications,* 5(3).

Kochenberger, G. A., Glover, F. Alidaee, B., & Rsgo, C. (2004). A unified modeling and solution framework for combinatorial optimization problems. *Operational Research Specktum,* 26(2).

Lipinski, P., Winezura, K., & Wojcik, J. (2007). *Building risk-optimal portfolio using evolutionary strategies.* In Giacobini M (eds) Proceedings of EvoWorkshops, Lecture Notes in Computer Science 4448.

Malik, K., & Tayal, A. (2014). Comparison of nature inspired meta-heuristic algorithms. *International Journal of Electronic and Electrical Engineering,* 7(8), 729-802.

Puchinger, J., & Raidl, G. R. (2005). *Combining Meta-heuristics and Exact Algorithms in Combinatorial Optimization: A Survey and Classification.* IWINAC'05 Proceedings of the First international work-conference on the Interplay Between Natural and Artificial Computation conference on Artificial Intelligence and Knowledge Engineering Applications: a bio inspired approach - Volume Part II.

Rajabioun, R. (2011). Cuckoo optimization algorithm. *Applied Soft Computing,* 11, 5508-551.

Sefiane, S., & Benbouziane, M. (2013). A meta-heuristic ant colony optimization method to portfolio optimization. *International Journal of Financial Management,* 3(4).

Senel. K., Pamukcu, A. B., & Yanik, S. (2006). An evolutionary approach to asset allocation contribution pension schemes. *Studies in Computational Intelligence*, 100.

Simon, D. (2008). Biography based Optimization. *IEEE, Transactions on Evolutionary Computation,* 12.

Slingh, U., & Rattan, M. (2014). Design of linear and circular antenna arrays using cuckoo optimization algorithm. *Progress in Electromagnetics Research,* 46, 1-11.

Taimoor, A., & Shoemaker, C. A. (2015). Multi **o**bjective optimization of computationally expensive multi-modal functions with RBF surrogates and multi-rule selection. *Journal of Global Optimization*, 64(1).