A MULTIOBJECTIVE GENETIC ALGORITHM FOR PORTFOLIO SELECTION WITH INTEGER CONSTRAINTS

By

K. P. Anagnostopoulos, G. Mamanis
Democritus University of Thrace, Greece

Abstract

In this paper we develop a computational procedure in order to find the efficient frontier, i.e. a non-decreasing curve representing the set of Pareto-optimal or non-dominated portfolios, for the standard Markowitz mean-variance model enriched with integer constraints. These constraints limit both the portfolio to contain a predetermined number of assets and the proportion of the portfolio held in a given asset. The problem is solved by adapting the multiobjective algorithm NSGA (Non-dominated Sorting Genetic Algorithm) that ranks the solutions of each generation in layers based on Pareto non-domination. The algorithm was applied in 60 assets of ATHEX and a comparison with a single genetic algorithm was realized. The computational results indicate that the procedure is promising for this class of problems. JEL Classifications: C61, C63, G11.

Keywords: Markowitz Model, Multiobjective Optimization, NSGA, Portfolio Selection.

1. Introduction

Every investor faces the problem of choice the appropriate assets in which he will invest his funds. To support such decisions, H.M. Markowitz set up some fifty years ago a quantitative framework, in which the selected portfolio is optimum with respect to both the expected return and the variance of return and maximizes the so-called utility function (Markowitz 1952, Markowitz 1990). The optimal portfolio offers the highest level of expected return for a given level of risk and the minimum level of risk for a given level of return. All such portfolios are called efficient and constitute the efficient frontier. The assumption that asset returns follow the normal distribution allows the finding of efficient frontier via quadratic programming.

However, Markowitz mean-variance model has been criticised not only for the main assumptions it is based upon, but also because it neglects some important aspects of portfolio performance in real life situations. As a result, some other measures of risk have been used, e.g. Value-at-Risk (Benati and Rizzi 2007, Gilli and Kellezi, 2002); and additional constraints were introduced in the standard model in order, for example, to avoid very small holdings, to restrict
the total number of holdings and/or to take into consideration the round lot of
assets that can be bought or sold in a bunch (Mitra 2003).

Since these additional constraints lead to sets of discrete variables and
constraints, the resulting optimization problem becomes quite complex as it
exhibits multiple local extrema and discontinuities (Chang et al. 2000, Crama
and Schyns 2003, Gilli and Kellezi 2002, Jobst et al. 2001). In such situations,
especially in large-scale instances of the problem, classical optimization meth-
ods do not work efficiently and heuristic optimization techniques are the only
alternatives for finding optimal or near-optimal solutions in a reasonable
amount of time. Thus, researchers have experimented with the application of
heuristic optimization techniques for finding the efficient frontier of the stand-
ard Markowitz model enriched with practical constraints. However, it must be
noted that, although many metaheuristic algorithms have been developed in the
past (Blum and Roli 2003), “few authors seem to have investigated the applica-
tion of local search metaheuristics for solving the portfolio selection problems”
(Crama and Schyns 2003).

One of the first attempts for the use of heuristic optimization techniques
to portfolio selection was made by Mansini and Speranza (1999). They have
formulated the optimum portfolio choice with round lots as a mixed integer
programming problem and they have proposed heuristics for its solution based
upon the idea of constructing and solving mixed integer sub-problems, which
consider subsets of the available investment choices. Chang et al. (2000) have
extended the standard Markowitz model to include cardinality constraints as
well as upper and lower bounds on the proportion of the portfolio invested in
each asset. For finding the cardinality constrained efficient frontier the authors
have applied three heuristic algorithms based upon genetic algorithms, tabu
search and simulated annealing. For the same problem, Anagnostopoulos et al.
(2004) have also proposed a GRASP algorithm enhanced by a learning mecha-
nism and a bias function for determining the next element to be introduced in
the solution. Crama and Schyns (2003) have also applied a simulated annealing
algorithm but they have extended the model to contain not only cardinality con-
straints and upper and lower bounds, but also trading and turnover constraints.
Jobst et al. investigated the shape of the efficient frontier of the means-variance
model including buy-in thresholds, cardinality constraints and round lot restric-
tions using a branch-and-bound algorithm combined with heuristics (Jobst et al.
2001).

In any case, the construction of the efficient frontier via quadratic program-
ning requires the optimization problem to be solved several times for various
values of return. In this paper we confront the standard Markowitz model with
cardinality constraints as a bi-objective optimization problem in order to find the efficient frontier in a single execution of the algorithm. The problem is solved by a multiobjective genetic algorithm, which uses a non-dominated sorting procedure to select the best parents. To the best of our knowledge, none of the related studies in the literature use a proper multiobjective algorithm to construct the Pareto front within the context of a portfolio selection problem such as the one considered in this work. The algorithm was applied in 60 assets of ATHEX and a comparison with a variant of the single (as opposed to multiobjective) genetic algorithm, which has been proposed by Chang et al. (2000), was realized. The computational results indicate that the procedure is very promising for this class of problems.

The rest of this paper is organized as follows. In Section 2, after a short review of the Markowitz model, the portfolio selection is defined as a multiobjective combinatorial problem. An adaptation of the Nondominated Genetic Algorithm (NSGA) for solving the problem is presented in Section 3. Section 4 is devoted to our numerical results, and some concluding remarks are presented in Section 5.

2. The formulation of the problem

2.1 The Markowitz mean-variance model

The problem of optimally selecting a portfolio among N assets was formulated by H.M. Markowitz in 1952. H.M. Markowitz based on the assumption that every investor has the desire to achieve a predetermined return and to minimize risk on investment. Mean or expected return is employed as a measure of return and standard deviation or variance of return is employed as a measure of risk. Among all portfolios there are special ones for which it cannot be said that one is better than the other. All such portfolios that are Pareto-optimal (or non-dominated) offer the maximum level of return for a given level of risk, or equivalently, the minimum level of risk for a given level of return. The investor should select a portfolio among the efficient portfolios. The proper choice among efficient portfolios depends on the willingness and ability of the investor to assume risk.

However, the main problem is to find this efficient frontier. Under the assumption of the normality of returns, this can be done by solving a quadratic optimization problem for all possible values of $\rho$, i.e. the desired level of return. The set of all optimal solutions constitutes the mean-variance frontier. It is usually displayed as a curve in the plane where the vertical axis denotes portfolio’s expected return, while the horizontal axis represents the variance of this return. Mathematically, the problem can be formulated as follows:
The objective function (1) minimizes the total variance (risk) associated with portfolio, while equation (2) ensures that the portfolio has an expected return of $\rho$. Equations (3) and (4) describe budget and non-negativity constraints respectively. Budget constraint ensures that 100% of the budget is invested in the portfolio, while non-negativity constraints ensure that no asset has a negative proportion.

An alternative form of the model is often used in practice (see, for example, Anagnostopoulos et al. 2004, Chang et al. 2000) by removing the return constraint and replacing the objective function (1) by

\[
\min \lambda \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} - (1 - \lambda) \sum_{i=1}^{N} w_i r_i
\]

Values of $\lambda$ satisfying $0 \leq \lambda \leq 1$ represent an explicit tradeoff between risk and return, and generate solutions between the two extremes $\lambda = 0$ and $\lambda = 1$. To draw the efficient frontier, the problem is repeatedly solved using several values of $\lambda$.

2.2 The multiobjective optimization model

For more realistic portfolio selection several extensions of Markowitz standard model have been proposed. In real financial decision-making, it is useful to avoid very small holdings, and to restrict the total number of assets. These
requirements can be modeled as threshold and cardinality constraints. In general, both lead to sets of discrete variables and constraints.

Threshold and cardinality constraints can be added to the model using a binary variable $z_i$, which is equal to 1 if the asset $i$ ($1 \leq i \leq N$) is held in the portfolio and 0 otherwise. Introducing finite upper and lower bounds $\varepsilon_i, \delta_i$ for the stock weight $w_i$, threshold constraints are represented by the following inequality
\[
\varepsilon_i z_i \leq w_i \leq \delta_i z_i, \ i=1,...,N
\]

To facilitate portfolio management or to control transaction costs, some investors may wish to limit the number of assets held in their portfolio. The cardinality constraint, which limits the portfolio to contain predetermined number of assets $K$, can be added to the model by counting the binary variables $z_i$. This constraint is expressed by the following equation
\[
\sum_{i=1}^{N} z_i = K
\]

When such constraints are added, the resulting mixed integer program becomes larger in size and computationally more complex than the standard mean-variance model.

In this paper we reformulate the quadratic optimization problem into a two-objective optimization problem. This allows us to find the efficient frontier in a single execution of the algorithm. The vector objective function has as elements the portfolio return and the variance of return. Moreover our model has been enriched with threshold and cardinality constraints.

The problem to be solved is formulated as follows

\[
\text{opt } f(w) = [f_1(w), f_2(w)]
\]
subject to
\[
\sum_{i=1}^{N} w_i = 1
\]
\[
\sum_{i=1}^{N} z_i = K
\]
\[
\varepsilon_i z_i \leq w_i \leq \delta_i z_i, \ i=1,...,N
\]
\[
z_i = 0 - 1
\]

The objective function $f_1(w)$ represents portfolio’s return while the objective function $f_2(w)$ represents portfolio’s variance of return. The $N$-vector $w$ denotes the set of decision variables $w_i$. 
3. The multiobjective algorithm

Multiobjective genetic algorithms have gained much attention last years in solving optimization problems with multiple objectives (Coello Coello 2000, Deb 1999). The primary reason of these studies is the unique feature of genetic algorithms to use a population of solutions. This allows multiple Pareto-optimal solutions to be found in a single simulation run. It appears that the first who tried to use genetic algorithms for finding the Pareto frontier in a multiobjective optimization problem was Schaffer (1985). Although his Vector Evaluated Genetic Algorithm (VEGA) gave encouraging results, it suffered from biasness towards some Pareto-optimal solutions. To overcome this problem, it is suggested the use of both techniques, a non-dominated sorting procedure to move a population toward the Pareto front and some kind of niching technique to keep the GA from converging to a single point on the front. Based on this suggestion a number of independent GA implementations have been proposed, for example the MultiObjective Genetic Algorithm (MOGA) (Fonseca and Fleming 1993) and the Niched-Pareto Genetic Algorithm (NPGA) (Horn et al. 1994).

Srinivas and Deb (1995) proposed the Nondominated Genetic Algorithm (NSGA) which is based on several layers of classifications of individuals. Before selection, a procedure ranks the solutions of each generation in layers based on Pareto non-domination. Firstly, the nondominated individuals are identified so that to constitute the first nondominated front; and they are assigned a large dummy fitness value, which is proportional to population size, to provide an equal reproductive potential to all these nondominated individuals. To maintain diversity in the population classified individuals are shared with their dummy fitness values. Sharing is achieved by dividing each individual’s dummy fitness value by a niche count which is proportional to the number of individuals one has in its neighborhood. The parameter niche count for every individual $i$ in the front is calculated by the following equation

$$c_i = \sum_{j=1}^{M} Sh(d_{ij})$$

where $Sh(d_{ij})$ is the sharing function, $d_{ij}$ is the phenotypic distance between individuals $i$ and $j$, and $M$ is the number of individuals in the current front.

Sharing function is expressed by the equation

$$\Phi(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\sigma_{sh}}\right)^{\alpha} & \text{if } d_{ij} < \sigma_{sh} \\ 0 & \text{otherwise} \end{cases}$$

where usually $\alpha = 1$, and $\sigma_{sh}$ is the maximum distance allowed between two individuals.
Sharing function plays an important role in NSGA’s performance, and it is strongly depended on the appropriate selection of the parameter $\sigma_{sh}$. The method proposed by Deb and Goldberg for estimating $\sigma_{sh}$ seems do not to work efficiently in our problem. This is probably due to the additional integer constraints which limit the search space. Thus, the algorithm was executed several times for different values of the parameter $\sigma_{sh}$, which was kept smaller than the initial value computed by Deb and Goldberg’s method, until the best efficient frontier was found. After sharing, these individuals are ignored temporarily and the second front of nondominated individuals is identified. These new set of points are assigned a new dummy fitness value which is kept smaller than the minimum shared fitness value of the first front (95% of the smallest shared fitness value of the previous front). The process continues until all individuals in the population are classified.

```
P ← randgeneratepopulation() /* Initial population P

 generation ← 0

 do while generation < maxgenerations
    find the vector of decision variables for each individual $i \in P$
    compute variance and return $\forall i \in P$
    $k \leftarrow 0$
    $D_k \leftarrow \emptyset$
    $F_k \leftarrow \emptyset$ /* the k-th front of individuals
    do until $P = \emptyset$ begin
       /* sorting procedure
       $k \leftarrow k+1$
       for all $i \in P$ and for all $j \neq i \in P$
          if for any $j$, individual $i$ is dominated by $j$ then
             $D_k \leftarrow D_{k-1} \cup \{i\}$
          else
             $F_k \leftarrow F_{k-1} \cup \{i\}$
          end if
       end for
       $P \leftarrow P - D_k$
       assign dummy fitness in each $i \in F_k$
       apply sharing function in $F_k$
    end do
    $P \leftarrow F_1 \cup \ldots \cup F_k$
    recombine $P$ according to shared fitness value
    mutate $P$
    generation ← generation + 1
 end do
```

Figure 1. Pseudocode of the algorithm
The population is then reproduced according to the shared fitness value. A stochastic remainder proportionate selection is used in this approach. Since individuals in the first front have the maximum fitness value, they always get more copies than the rest of the population. This allows the search for non-dominated regions and sharing helps to distribute the population over this region. The efficiency of NSGA lies in the way multiple objectives are reduced to a dummy fitness function using nondominated sorting procedure. Another aspect is that any number of objectives can be solved and both minimization and maximization problems can be handled (Srinivas and Deb 1995). The pseudocode of the algorithm is shown in Figure 1.

A crucial aspect in genetic algorithms is how to represent a solution. The chromosome is divided into two parts. The first part is a set $A$ of $K$ distinct assets and the second one is a set $B$ that includes $K$ real numbers associated with each asset $i$.

$$A = \{\alpha_1, \ldots, \alpha_K\}, \quad \alpha_i \in \{1, N\}$$

$$B = \{n_{\alpha_1}, \ldots, n_{\alpha_K}\}, \quad 0 \leq n_i \leq 1, \quad i \in A$$

Then, in order to find the proportion of each asset, the free portfolio proportion is calculated as follows

$$fpp = 1 - \sum_{i=1}^{K} s_{\alpha_i}$$

Thereafter, the proportion associated with each asset in the portfolio is calculated by the following equation

$$w_{\alpha_i} = s_{\alpha_i} + \frac{n_{\alpha_i}}{\sum n_{\alpha_i}} fpp$$

In this way all the constraints are satisfied.

The offspring are generated by uniform crossover as described below. If an asset is present in both parents it is present in the children with the corresponding associated value $n$. The remaining non-common assets are then selected randomly to fulfill children’s sets. An example can be seen in Table 1.

Children are also subject to mutation by multiplying by 0.9 or 1.1 (chosen with equal probability) the value $n_i$ of a randomly selected asset $i$. The next generation of individuals completely replaces the current population.


<table>
<thead>
<tr>
<th>TABLE 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crossover example</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parent 1</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.1</td>
<td>1</td>
<td>0</td>
<td>0.7</td>
<td>0.8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parent 2</th>
<th>10</th>
<th>20</th>
<th>5</th>
<th>15</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.9</td>
<td>0</td>
<td>0.6</td>
<td>0.4</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Offspring 1</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>5</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.1</td>
<td>0</td>
<td>0</td>
<td>0.6</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Offspring 2</th>
<th>10</th>
<th>20</th>
<th>40</th>
<th>50</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.9</td>
<td>1</td>
<td>0.7</td>
<td>0.8</td>
<td>0.4</td>
</tr>
</tbody>
</table>

4. Computational results

The algorithm has been implemented in Visual Basic and run on a personal computer Pentium 4 at 2.4 GHz. To construct the data set, 60 assets of big και medium capitalization from Athens Exchange were considered and weekly prices from 10-5-2005 to 12-5-2006 were used to calculate returns and covariances. The weekly return $r_i^t$ of the asset $i$ in the period $t$ was calculated according the equation

$$r_i^t = \frac{\tau_i^c - \tau_i^b + d_i^t}{\tau_i^c}$$

where $\tau_i^c$ (τ i^b) is the closing price of asset $i$ at the end (beginning) of period $t$ and $d_i^t$ is the dividend paid to shareholders in period $t$.

We tried to find the efficient frontier for different values of $K$ and especially for $K = 2, 5, 10$. For all these problems lower and upper bounds were 1% and 100% respectively, i.e., $\varepsilon_i = 0.1$, $\delta_i = 1 \quad \forall i \in A$
In order to see the algorithm performance, an initial population has been randomly generated (Fig. 1). Figures 2, 3 and 4 represent the cardinality constrained efficient frontier for $K = 2, 5, 10$ respectively. As we can see from these outputs, the algorithm has found many Pareto-optimal points with good distribution along the efficient frontier. The number of generated points and their distribution are crucial aspects in multiobjective optimization.

Figure 2. *The initial population*

![Initial Population](image1)

Figure 3. *The efficient frontier, $K = 10$*

![Efficient Frontier K=10](image2)
If the multiobjective algorithm converges in a small region near or on the true Pareto-optimal front, the purpose of multiobjective optimization is not served. This is because, in such cases, many interesting solutions with large trade-offs among the objectives and parameter values have been probably undiscovered. Table 1 illustrates this distribution of points for each problem instance, together with important parameters of the algorithm.
We have also implemented a variant of the genetic algorithm proposed in Chang et al. (2000). The differences between their genetic algorithm and our algorithm are, one the one hand, the complete replacing of the solutions (as in our multiobjective algorithm) versus the partial replacing and, on the other hand, the rank selection versus the tournament selection. Because of limited space, only some of the obtained results are presented.

![Graph](image)

**Figure 6. The efficient frontier generated by the single genetic algorithm, $K = 10$**

In order to compare the quality of solutions obtained by the multiobjective genetic algorithm and the single objective genetic algorithm, we use the technique proposed in (Jaszkiewicz 2000). The multiobjective genetic algorithm is considered not worse than the single objective if
where $s_i$ is the scalarizing function, $w^{sl}$ the best solution obtained by optimization of $s_i$ with the single GA and $w^{ml}$ the best solution on $s_i$ selected from the set of Pareto-optimal solutions generated by the multiple objective GA. Figure 6 shows the solutions obtained by optimizing 81 objective functions ($l = 1, \ldots, 81$) with single objective GA, defined for values $\lambda = 0$ to 1 with step 0.0125 (see Equation 5). Since

$$\sum_{l=1}^{L} \left( s_i \left( w^{ml} \right) - s_i \left( w^{sl} \right) \right) \leq 0$$

we may compare the computational requirements of the two approaches. The effectiveness index is equal to

$$EI = \frac{CT_m}{CT_s} = 25$$

where $CT_s$ is the average running time the single objective GA spent on optimization of $s_i$ and $CT_m$ the running time the multiobjective GA needs to generate the Pareto-optimal solutions $s_1 \ldots s_L$. These results are based on the Pareto front generated from the multiobjective algorithm with 200 generations. If the generations are equal to 50 (although the Pareto front is slightly inferior), the equation is still verified and the $EI$ is equal to 6,025. Thus we can conclude that the generation of the Pareto-optimal solutions with NSGA is competitive both from the computational effectiveness point of view and the quality of the Pareto front.

5. Conclusions

Constraints in the size of the portfolio and in lower and upper bounds on the proportion of the portfolio held in a given asset transform the standard Markowitz model in a mixed integer optimization problem and create discontinuities in the efficient frontier. In this paper we adapt the multiobjective algorithm NSGA for finding the cardinality constrained efficient frontier. We argue that the proposed procedure solves efficiently the cardinality constrained portfolio optimization problem as it generates in relatively short computational time a large number of Pareto-optimal solutions, which are uniformly distributed along the efficient frontier. Even if the efficient frontier is not continuous and, then, competition among solutions may lead to extinction of some sub-regions, the algorithm finds a large number of Pareto-optimal
solutions in every segment. On the other hand, the procedure is in general
time consuming, since the quality of solutions depends on the population size,
but this shortcoming is balanced by the fact that the efficient solutions are
obtained after a small number of generations. Finally, a further difficulty is the
appropriate selection of $\sigma_{sh}$ as the algorithm performance is highly dependent
on this value.

Constraints in the size of portfolio and in lower and upper bounds on the
proportion of the portfolio in a given asset help the decision maker to facilitate
its portfolio management; and to avoid excessive transaction costs on one hand,
and to avoid holding very small/large amounts of any particular asset on the
other. It is empirically known that much of the portfolio risk can be diversified
by holding a rather small number of assets (Maringer 2005, ch. 4). We have
solved for the efficient frontier following the tradition of standard Markowitz
approach, however, focusing on the case where the investor wants to invest in
exactly $K$ out of $N$ number of assets. Furthermore, portfolios with positions in
assets with very small amounts have been excluded through the use of threshold
constraints. The resulting efficient frontier gives the best possible trade-off risk
against return for a particular number of assets ($K$). The investor then examines
the trade-off points in the possibilities curve and selects the one particular point
of interest. This may be the point with the lowest variance but having the low-
est return, located in the lower left part of the frontier; or it may be the point
with the maximum expected return but with the maximum risk, located in the
right upper part of the frontier; or it may be any intermediate point. The proper
selection of the particular point depends on the investor’s willingness to assume
risk. In the next step, the investor implements the one particular portfolio
whose image is the point in the nondominated frontier. Furthermore, solving
for different values of $K$, the trade-off between risk, return and the number of
assets of the portfolio could be examined.

Currently our research focus on a generalization of the cardinality con-
strained mean-variance problem, by including class constraints that limit the
proportion of the portfolio that can be invested in assets in each class, such as
bank stocks, telecommunication stocks etc. For its solution, procedures of the
so called second generation multiobjective genetic algorithms are tested.
References


