Computing and Informatics, Vol. 32, 2013, 777–796

MULTIOBJECTIVE ALGORITHMS WITH RESAMPLING FOR PORTFOLIO OPTIMIZATION

Sandra García, David Quintana, Inés M. Galván, Pedro Isasi

Computer Science Department
Carlos III University of Madrid
Avda. Universidad 30
28911 Leganes, Spain
e-mail: sgrodrig@inf.uc3m.es

Communicated by Jiří Pospíchal

Abstract. Constrained financial portfolio optimization is a challenging domain where the use of multiobjective evolutionary algorithms has been thriving over the last few years. One of the major issues related to this problem is the dependence of the results on a set of parameters. Given the nature of financial prediction, these figures are often inaccurate, which results in unreliable estimates for the efficient frontier. In this paper we introduce a resampling mechanism that deals with uncertainty in the parameters and results in efficient frontiers that are more robust. We test this idea on real data using four multiobjective optimization algorithms (NSGA-II, GDE3, SMPSO and SPEA2). The results show that resampling significantly increases the reliability of the resulting portfolios.

Keywords: Financial portfolio optimization, robust portfolio, multiobjective evolutionary algorithms

1 INTRODUCTION

The problem of choosing the right combination of financial assets has been the subject of research for a long time and it is one of the most active research lines in finance. This is often framed as a multiobjective optimization problem where the investor tries to find the right set of portfolios with the best risk/return profiles. The simultaneous minimization of risk and maximization of returns using two different objective functions defines a Pareto front that is referred to as the efficient frontier.

A large portion of the academic literature on this subject builds on the seminal work by Markowitz [13, 10]. The approach suggested by this author works under some assumptions that allow the problem to be tackled with quadratic programming. Unfortunately, these assumptions do not hold in the real world, which calls for alternatives. That is the reason why the framework of evolutionary computation is getting traction on this area. The solutions that have been suggested for this multiobjective problem using evolutionary computation range from weighting objectives into a single objective function. Chiranjeevi and Sastry [3] follow the first approach and break the above-mentioned objective into five components that are combined into a single fitness function that is optimized under cardinality constraints. Soleimani et al. [24] perform classic mean-variance optimization considering constraints such as transaction costs, round lots or cardinality constraints using a standard genetic algorithm. Among the authors that fall on the second category we could mention Skolpadungket et al. [3]. They test the performance of a set of multi-objective algorithms (VEGA, SPEA2, NSGA-II. . .) on a constrained version of the two objective problem. We also find more references [1, 7, 23]. For instance, the first one introduces a customized hybrid version of NSGA-II and the last two compare the performance of different multiobjective algorithms. Another example could be [18], where the authors compare FastPGA, MOCELL, AbYSS, and NSGA-II in both of the basic problems, and an extended version that considers the dividend yield as a third objective.

One of the most important factors that asset managers face when they have to assess the results provided by any of the above-mentioned methods is stability. Very often, the expected efficient frontier lies far from the actual one as the forecast risk/return profile of the portfolios is not accurate. This problem is one of the major reasons why some practitioners mistrust these kinds of approaches and the search for solutions has cleared the way for the field of robust portfolio optimization. The works mentioned before handle risk and return, but do not deal with the robustness of solutions. The main contribution of our work is the introduction of a resampling mechanism that reduces the risk mentioned before under a constrained portfolio problem.

When we forecast the risk and return of a specific portfolio, we rely on estimates for the expected returns of individual assets and the variance-covariance matrix. The forecasts for the expected returns might be inaccurate and the variance-covariance may suffer from the same problem. These forecasts are usually based on past data and this data may not be representative to picture the future due to, for instance, the presence of outliers.

In this context, there are several potential ways to approach the problem. The main two ones are either putting an emphasis on having robust estimates for the above-mentioned parameters [16] or implementing a system that deals with uncertainty in the estimation process [17, 25]. The approach suggested in this paper falls in the latter category. We will use multiobjective evolutionary algorithms (MOEAs) enhanced with a resampling mechanism that changes the parameters of the fitness function during the evolution process. We consider that using an evolutionary multiobjective algorithm that resamples past data to generate different scenarios and evolves the system using that information will improve the reliability of the resulting Pareto front.

The use of resampling in the context of portfolio optimization has been studied in [22, 20]. The most comparable traditional approach is described by Idzorek [11] who suggests combining traditional quadratic programming (QP) with a Monte Carlo simulation to derive a set of fronts that are subsequently merged into a single solution. The using of resampling within the context of multiobjective evolutionary algorithms is a better strategy because real-world constraints are intractable by QP and, at the same time, the approximation to the efficient frontier is made in a single run [15].

The rationale for the approach is that optimizing for a single scenario bears the risk of getting solutions that might be extremely sensitive to deviations in the parameters used in the fitness evaluation. Given that it is almost certain that we will not be able to accurately predict the behavior of all the assets, we could consider the alternative of targeting portfolios that offer appropriate risk/return tradeoffs under different scenarios.

The approach used in this work builds on [9] and extends the solution to the constrained version of the problem. We also test it on new algorithms. The nature of the resampling approach described in this paper is compatible with a wide array of evolutionary multiobjective algorithms. Given their different nature and behaviour, the study will compare the effect of adding the resampling strategy to the fitness functions on the representative set testing their performance. For this purpose, we have selected NSGA-II [6], one the most referenced algorithms in the field of multiobjective optimization. We have also chosen GDE3 [12], a differential evolution strategy, and SPEA2 [27], an algorithm that has been confirmed [23] to offer a good performance in portfolio optimization context. Finally, a multiobjective algorithm based on swarm, SMPSO [14], has been selected to explore the capability of swarm formulation in our context.

The rest of the paper is organized as follows. First, we make a formal introduction to the financial portfolio optimization problem. Then, we briefly present the evolutionary multiobjective algorithms used in this work. After that, we will describe our approach in detail. That will be followed by the experimental results and, finally, there will be a section devoted to summary and conclusions.

2 FINANCIAL PORTFOLIO OPTIMIZATION

The portfolio can be defined as a collection of investments or assets held by an institution or a private individual. The Modern Portfolio Theory was originated in the article published by Harry M. Markowitz in 1952 [13]. It explains how to use diversification to optimize the portfolio. In general, the portfolio optimization problem is the choice of an optimum set of assets to include in the portfolio and the distribution of the investor's wealth among them. Markowitz [10] assumed that solving the problem requires simultaneous satisfaction of maximizing the expected portfolio return $E(R_p)$ and minimizing the portfolio risk (variance) σ_p^2, that is, solving a multiobjective optimization problem with two output objective functions [24, 3, 23, 18]. The portfolio optimization problem can be formally defined as follows:

- minimize the **risk** (variance) of the portfolio:

$$\sigma_p^2 = \Sigma_{i=1}^n \Sigma_{j=1}^n w_i w_j \sigma_{ij} \tag{1}$$

- maximize the **return** of the portfolio:

$$E(R_p) = \Sigma_{i=1}^n w_i \mu_i \tag{2}$$

- subject to:

$$\Sigma_{i=1}^n w_i = 1 \tag{3}$$

$$0 \leq w_i \leq 1; i = 1 \ldots n \tag{4}$$

where n is the number of available assets, μ_i the expected return of the asset i, σ_{ij} the covariance between asset i and j, and w_i are the decision variables giving the composition of the portfolio. The constraints referenced in Equations (3) and (4) require full investment of funds and prevent the investor from shortening any asset, respectively. In a quantitative way, the risk is represented with the standard deviation σ_p.

The solution to the problem should also consider some real world constraints [2] such as:

- Cardinality constraint: it is possible to define the maximum C_{max} and minimum C_{min} number of assets in which it is possible to invest ($w_i \neq 0$):

$$C_{min} \leq \Sigma(w_i \neq 0) \leq C_{max}. \tag{5}$$

- Values limit constraint: each weight w_i must have a value in the interval [lim_{inf}, lim_{sup}], where:

$$w_i \geq lim_{inf} \geq 0.0; w_i \leq lim_{sup} \leq 1.0; lim_{inf} \leq lim_{sup}. \tag{6}$$

All of these equations are solved by a set of points that constitute the efficient frontier of the problem. The points will define a curve similar to that in Figure 1, plotted in the *risk-return* space of all possible portfolios. The points of this curve represent portfolios which have the minimum amount of risk given a certain expected return (and vice versa).

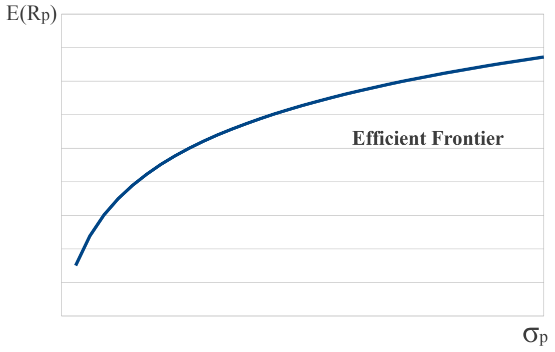

Figure 1. Efficient frontier

3 EVOLUTIONARY MULTIOBJECTIVE ALGORITHMS

As mentioned in the introduction, different evolutionary multiobjective algorithms have been tested in the context of our work.

The field of evolutionary mutiobjective optimization has been growing steadily over the last few decades since David Schaffer introduced VEGA in 1984 [21]. Unlike more traditional approaches, these algorithms do not target a single solution, but a whole set of alternatives that offer good compromises among the range of objectives. Each of these alternatives is non dominated. This means that there exists no other feasible solution which would provide an improvement in terms of an objective without causing a simultaneous deterioration in terms of another one. Not only that, they target the set of solutions in a single run.

Their structure has changed over time but second generation algorithms, such as the ones used in this paper, tend to have some traits in common. These algorithms are based on the idea of dominance, that is the algorithm favors those solutions that are non-dominated and makes a wide use of elitism. This usually means that there is an external population, or a secondary one, that stores the non-dominated individuals found along the evolutionary process. In addition to that, they usually include diversity preservation mechanisms that ensure that the solutions offered by the algorithm are spread along the whole Pareto front and not clustered together. Those who look for an introduction to the field or its applications might find [5] or [4] interesting.

As an illustration, we provide the description of SPEA2 [27], one of the algorithms most used to test resampling. SPEA2 was developed by Zitzler, Laumands and Thiele to solve some weaknesses of a previous version by the same authors called SPEA. Among the improvements, we could mention a fitness function that takes into account, for each solution, the number of individuals dominated by this one and the number of individuals which dominates it. This version also adds a density estimation of the population.

This algorithm uses a population and an archive. It assigns each individual a fitness value that is the sum of its strength raw fitness plus a density estimation. In each generation the non-dominated individuals of both the original population and the archive are used to update the archive; if the number of non-dominated individuals is greater than the population size, a truncation operator based on calculating the distances to the k^{th} nearest neighbor is used. All this procedure is known as Environmental Selection. Then, the algorithm applies the selection, crossover, and mutation operators to members of the archive in order to create a new population of offsprings which becomes the population of the next generation.

Algorithm 1 SPEA2 Algorithm Pseudocode

```
Input: $N$ Population size
Input: $\overline{N}$ Archive size
Input: $T$ Maximum number of generations
Input: Crossover probability
Input: Mutation probability
1. Initialization: Create the initial population $P_0$ and the empty archive (external set) $\overline{P_0} = \varnothing$. Set $t = 0$.
2. Fitness assignment: Calculate fitness values of individuals in $P_t$ and $\overline{P_t}$
3. Environmental selection: Copy all non-dominated individuals in $P_t$ and $\overline{P}_t$ to $\overline{P}_{t+1}$. If size of $\overline{P}_{t+1}$ exceeds $\overline{N}$ then reduce $\overline{P}_{t+1}$ by means of the truncation operator, otherwise if size of $\overline{P}_{t+1}$ is less than $\overline{N}$ then fill $\overline{P}_{t+1}$ with dominated individuals in $P_t$ and $\overline{P}_t$
4. Termination: If $t \geq T$ or another stopping criterion is satisfied then set $A$ to the set of decision vectors represented by the non-dominated individuals in $\overline{P}_{t+1}$. Stop.
5. Selection: Perform binary tournament selection with replacement on $\overline{P}_{t+1}$ in order to fill the mating pool.
6. Variation: Apply recombination and mutation operators to the mating pool and set $P_{t+1}$ to the resulting population. Increment generation counter ($t = t + 1$) and go to Step 2.
Output: Non-dominated set.
```

The approach introduced in this paper was also tested with the following algorithms: NSGA-II, GDE3 and SMPSO.

NSGA-II, proposed by Deb et al. [6], is one of the most widely used multiobjective metaheuristics. It represents the new version of the NSGA algorithm proposed by the same author. It is a generational genetic algorithm based on obtaining an auxiliary population from the original one by applying the typical genetic operators (selection, crossover and mutation). Then, the two populations are merged and the individuals are sorted according to their rank. Inside each of these ranks, the crowding distance is used to sort the individuals from less to more crowded. A solution with a smaller value of this distance measure is, in some sense, more crowded by other solutions. Finally, the best solutions are selected to compose the new population that will be used to create the new offspring population.

GDE3 [12] is the third version of the Generalized Differential Evolution algorithm (GDE), extension of Differential Evolution strategy (DE) for global optimization with an arbitrary number of objectives and constraints. GDE3 starts with a population of random solutions. At each generation, an offspring population is created using the differential evolution operators; then, the current population for the next generation is updated using the solutions of both the offspring and the current population. Before creating the next generation, the size of the population is reduced using non-dominated sorting and a pruning technique aimed at diversity preservation, in a similar way as NSGA-II. However, GDE3 modifies the crowding distance of NSGA-II in order to solve some of its drawbacks when dealing with problems having more than two objectives.

SMPSO, proposed by Nebro et al. [14], is the new multiobjective particle swarm optimization algorithm (PSO) characterized by the use of a strategy to limit the velocity of the particles. It is based in OMOPSO [19] but including the velocity constriction procedure. This mechanism is useful when the velocity becomes too large because it can produce new effective particles positions. SMPSO also works with an external archive that stores the non-dominated solutions found during the searching process. Polynomial mutation [26] is used in the algorithm as a turbulence factor.

All the algorithms are implemented in jMetal [8], a Java framework aimed at multiobjective optimization with metaheuristics. By reusing the base classes of jMetal, all the techniques share the same basic core components (solution encodings, operators, etc.), which ensures a fair comparison of the considered techniques.

4 SUGGESTED MODELING APPROACH

The portfolio optimization problem is tackled in this work using different evolutionary multiobjective algorithms. All of them share the same chromosome structure and fitness evaluation procedure. This section introduces the details related to both issues and the metrics used in the experimental section.

4.1 Solution Encoding

The encoding chosen will represent each portfolio as a vector of real numbers; this means that the algorithms will work with real elements instead of binary ones. Each of these numbers represents the percentage of investment per asset (also called weight: w_i when i goes from $1 \ldots n$, where n is the number of investable assets). Here, each portfolio will be represented by a single element of the population.

Every individual must follow the constraints specified by Equations (3), (4) explained before. The sum of weights per individual must be 1.0, that is full investment is required. Also, the individuals must satisfy additional real-world constraints showed in Equations (5) and (6). Therefore, in order to satisfy all these constraints, the individuals are repaired after initializing the population (see Algorithm 2) and after applying the genetic operators (see Algorithm 3). Both repairing algorithms are based on the most intuitive procedure to transform chromosomes into portfolio satisfying constraints. Each individual of the population is checked and repaired. Whenever the number of invested assets does not belong to the interval $[C_{min}, C_{max}]$, its number is adjusted to ensure compliance with the cardinality constraint. This is achieved either by adding assets to the portfolio or dropping them until the requirement is met. In case the sum of weights per individual is not 1.0, the algorithm fine-tunes the holdings adding or subtracting random amounts up to the required adjustment. These changes are forced to comply with the investment limits $[lim_{inf}, lim_{sup}]$. The details of this process are described in Algorithms 2 and 3.

Algorithm 2 Reparation after initialization

```
Initialize population $P$ as a set of vectors with real numbers $\overline{x}_i = (x_{i1}, \ldots, x_{in})$; $x_{ij} \in [lim_{inf}, lim_{sup}]$
for each individual $\overline{x}_i$ of $P$ do
  A random number $\in [C_{min}, C_{max}]$ of values 0 is assigned to coordinates of vector $\overline{x}_i$
  while $\Sigma_{j=1}^{n} x_{ij} \neq 1$ do
    if $\Sigma_{j=1}^{n}(lim_{inf}/x_{ij} \neq 0) > 1$ then
      select randomly one coordinate $j$ of vector $\overline{x}_i$ such that $x_{ij} \neq 0$ and assign $x_{ij} = 0$
    end if
    if $\Sigma_{j=1}^{n}(lim_{inf}/x_{ij} \neq 0) < 1$ then
      select randomly one coordinate $j$ of vector $\overline{x}_i$ such that $x_{ij} = 0$ and assign $x_{ij} = lim_{sup}$
    end if
    if $\Sigma_{j=1}^{n} x_{ij} \neq 1$ then
      select randomly one coordinate $j$ of vector $\overline{x}_i$ such that $x_{ij} \neq 0$
      add/subtract the quantity left to make the $\Sigma_{j=1}^{n} x_{ij} = 1$ respecting the limits $[lim_{inf}, lim_{sup}]$
    end if
  end while
end for
Return($P$)
```

4.2 Fitness Functions: Resampling Parameters

As we mentioned, in this paper a resampling strategy is used to evaluate the fitness function with the purpose to obtain more robust and stable solutions for the portfolio optimization problem.

As apparent from Equations (1) and (2), these functions are very dependent on the values of the expected asset returns and the variance-covariance matrix. One of the most important challenges that portfolio managers face when they operate within Markowitz's framework is the dependence of the solutions on the estimates for these parameters. Given the difficulty inherent to financial forecasting, it is normal that these parameters are not accurate. This lack of accuracy is likely to result in

Algorithm 3 Reparation after genetic operators

```
for each individual $\overline{x}_i$ of $P$ do
  if there are less $x_{ij} = 0$ than $C_{min}$ then
    set one random $x_{ij} = 0$ to $x_{ij} = lim_{inf}$
  end if
  if there are more $x_{ij} = 0$ than $C_{max}$ then
    set the $x_{ij} \neq 0$ with less value to $x_{ij} = 0$
  end if
  while $\Sigma_{j=1}^{n} x_{ij} \neq 1$ do
    if $\Sigma_{j=1}^{n}(lim_{inf}/x_{ij} \neq 0) > 1$ then
      select randomly one coordinate $j$ of vector $\overline{x}_i$ such that $x_{ij} \neq 0$ and assign $x_{ij} = 0$
    end if
    if $\Sigma_{j=1}^{n}(lim_{inf}/x_{ij} \neq 0) < 1$ then
      select randomly one coordinate $j$ of vector $\overline{x}_i$ such that $x_{ij} = 0$ and assign
      $x_{ij} = lim_{sup}$
    end if
    if $\Sigma_{j=1}^{n} x_{ij} \neq 1$ then
      select one random $x_{ij} \neq 0 \in [lim_{inf}, lim_{sup}]$
      add/subtract one random quantity left to make the $\Sigma_{j=1}^{n} x_{ij} = 1$ respecting the
      limits $[lim_{inf}, lim_{sup}]$
    end if
  end while
end for
Return($P$)
```

a set of portfolios that are expected to behave in one way and do so in a completely different one. Figure 2 shows a real example of one Pareto front where the solutions are evaluated using the forecast parameters (represented with "+" points at the top) and the real parameters (represented with "x" points at the bottom). The portfolios that were optimised for the most likely scenario (mean return for each asset over a period of time, and variance-covariance matrix computed using the same data) define the Pareto front on top. However, it is clear that, once we calculate the risk and returns for the very same portfolios using the real observed parameters (actual assets returns instead of the mentioned averages), their profiles can be very different. The difference between the two fronts, that represent the same solution, can be potentially very important. For this reason, we suggest altering the fitness function to manage this risk.

The basic idea behind the solution that we suggest is to keep changing, for every generation, the parameters of the fitness functions during the evolution process. This prevents the efficient frontier from being specialized in a single scenario. Furthermore, the algorithm may find solutions with good performance evaluated under resampled scenarios whose values are likely to be close to the real ones. These solutions would have better performance for the real value of the parameters (only known a posteriori).

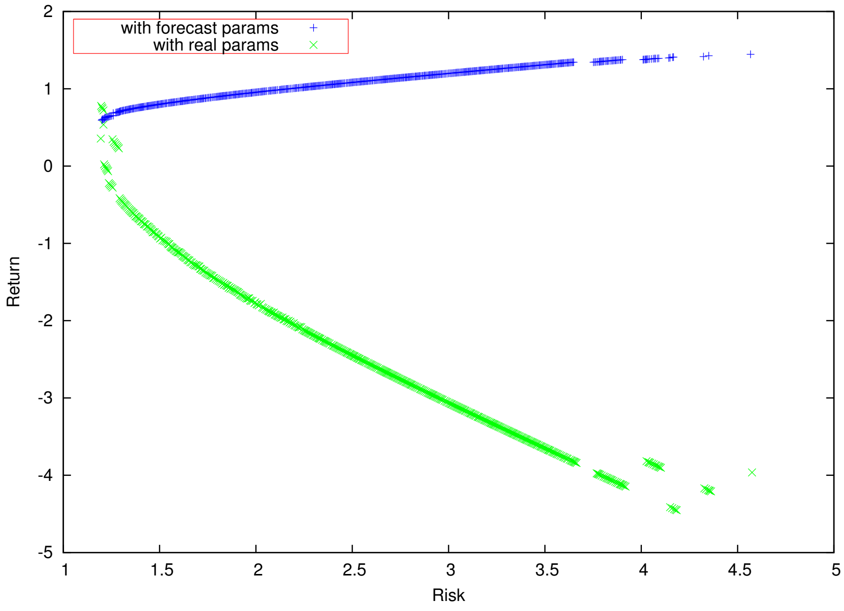

Figure 2. Solution evaluated with forecast and real parameters

The resampling mechanism has very little computational cost (almost not perceived) to the algorithm. The starting point for the evaluation of portfolios is the framework introduced in Section 2, where the fitness of each individual is determined by evaluating the two objective functions: return $E(R_p)$ to be maximized, and the risk σ_p to be minimized. In our approach, once the individual is evaluated in one resampled scenario, it is not evaluated any more until the end of the execution when its objectives are calculated under the forecast non-resampled scenario. Moreover, different individuals are evaluated in different scenarios along the evolution process, which can avoid getting all solutions specialized in a single scenario.

The approach that we use to generate these scenarios is a nonparametric bootstrap. The usual way to forecast the value of parameters for the model is averaging the returns and computing the associated variance covariance matrix over a number of periods. Our algorithm starts from the same point, the definition of a time window. Instead of using all the data to derive a single estimate for the parameters, data are resampled. The resampling process selects a random set of time periods that has the same size as the original window (each period might be selected more than once). Then, we average the returns for those time periods and compute the variance-covariance matrix. Every time we do this, we generate new estimates for the parameters that are based on past data. These estimates can subsequently be used to calculate the risk and return of the portfolios. The process is described in Algorithm 4.

Algorithm 4 Resampling method

```
$S$ is the original sample set with a size $N_s$.
$S'$ is the new sample set with a size $N'_s$. At the beginning, $S' = \varnothing$ and $N'_s = 0$.
while $N'_s \neq N_s$ do
    Select randomly $X_i \in S$.
    $S' = S' \cup \{X_i\}$.
end while
Return($S'$)
```

Hence, when we use resampling we create a set of "likely scenarios" that produces less specialized solutions in just one scenario which can be close, or not, to the reality, making the solution more robust.

4.3 Evaluation Metrics

Solutions resulting from different runs of the multiobjective algorithms must be compared using quantitative metrics that measure the success of the algorithms. It is generally admitted that there is no single metric that can be used to evaluate those objectives simultaneously; this is especially true when the best Pareto-optimal front is not known.

The metrics most widely used in multiobjective context are Hypervolume and Spread [28]. These metrics could be useful when we look for well distributed and strong dominant fronts. However, they are not appropriate to capture the effect showed in Figure 2. For this reason, we define a specific metric based on the Euclidean Distance to model the reliability of the solutions.

The metric used in this work is calculated separately for both return and risk objectives. The aim is to evaluate the distance between the forecast risk/return for every portfolio in the efficient frontier $(\sigma_p^2, E(Rp))$ and the actual risk/return for the same portfolio computed with the real parameters $((\sigma_p^2)', E'(Rp))$, that is, the distance between the estimates for t_n based on data from t_1 to t_{n-1}, and the actual values at t_n.

The expression in terms of return is formally defined for each efficient frontier in Equation (7).

$$ED_{E(R)} = \frac{\sqrt{\sum_{p=1}^{N}(E(R_p) - E'(R_p))^2}}{N} \tag{7}$$

where N is the number of portfolios in the Pareto front. The equivalent in terms of risk is Equation (8).

$$ED_{\sigma^2} = \frac{\sqrt{\sum_{p=1}^{N}(\sigma_p^2 - (\sigma_p^2)')^2}}{N} \tag{8}$$

These metrics are specific for each efficient frontier and the lower the values are, the shorter the distance between the forecasted risk and return and the actual ones in terms of each objective.

5 EXPERIMENTAL VALIDATION

In this section we test the aforementioned approach on a specific asset allocation problem. We try to identify the mixes of broad investment categories that provide the optimal balance between expected risk and return, while increasing robustness.

As we described in the the introduction, we consider different multiobjective meta-heuristics on the constrained version of the problem. For every algorithm, NSGA-II, SPEAII, SMPSO and GDE3, we compare the performance of the standard version versus the resampled equivalent. For experimental purposes, the cardinality constraints $[C_{min}, C_{max}]$ and the value limit constraints $[lim_{inf}, lim_{sup}]$ are set to $[2, 6]$ and $[0.1, 0.8]$, respectively. The choice of these limits is arbitrary but in line with the values found in the literature. These limits were set from the beginning and no experimentation was made for any alternative value.

Section 5.1 describes the data sets used in this work. Section 5.2 gives details about the parameter settings for multiobjective algorithms and some general parameters of the problem. Finally, Section 5.3 presents and analyses the experimental results.

5.1 Data Set

The sample used for the experimental analysis consists of 240 monthly returns for eight broad financial indexes representing that many asset classes. The series of monthly returns cover January, 1990 through December, 2009 and the source is the data vendor *Datastream*. Given the nature of the source, the sample cannot be made available by the authors. However, the list of indexes and codes is provided in Table 1, making the experimental work replicable.

Name	Code
Frank Russell 2000 Value	FRUS2VA
Frank Russell 2000 Growth	FRUS2GR
Frank Russell 1000 Value	FRUS1VA
Frank Russell 1000 Growth	FRUS1GR
S & P GSCI Commodity Total Return	GSCITOT
MSCI EAFE	MSEAFEL
BOFA ML CORP MSTR ($)	MLCORPM
BOFA ML US TRSY/AGCY MSTRAAA ($)	MLUSALM

Table 1. Data Sets

5.2 Experimental Settings

We implement the sliding window technique. Each window has a size n of 120 series of returns. The series t_1 to t_{n-1} are used in the algorithm to get the final solutions and the period t_n is used to evaluate them. Each time the window moves one month. In our experimentation the window is moved 120 times taking the interval from 31/01/1990 to 31/12/2009. Previous experiments modifying the size of the windows have been carried out. In order to work with full years we studied the window sizes which were multiples of 12 and we saw that the best one for our case was 120.The reason is that, using less than 120 periods per window, the Pareto fronts are less accurate, but with more than 120 the time of computation grows significantly while the quality of the fronts remains similar.

Under the parameters described in Table 2, the algorithms are run 20 times per window getting for each window a set of 20. The results are reported in Section 5.3.

SPEA2	
Population size	50, 200 and 500 individuals
Archive size	50, 200 and 500 individuals
Crossover	SBX, $p_c = 0.9$
Mutation	Polynomial, $p_m = 1/L$
Selection of Parents	Binary tournament
NSGA-II	
Population size	50, 200 and 500 individuals
Crossover	SBX, $p_c = 0.9$
Mutation	Polynomial, $p_m = 1/L$
Selection of Parents	Binary tournament
SMPSO	
Archive size	50, 200 and 500 individuals
Swarm size	50, 200 and 500 particles
Mutation	Polynomial, $p_m = 1/L$
GDE3	
Population size	50, 200 and 500 individuals
Crossover	DE crossover, $CR = 0.9$
Mutation	DE mutation, $F = 0.5$
Selection of Parents	DE selection

Table 2. Parameters. $L = 8$ (individual length). The termination condition is to compute 100 iterations.

5.3 Experimental Results

This section shows the results of the experimental process described before. For every multiobjective algorithm tested, we compare the performance of the standard and the resampled version of the algorithm. The version of the algorithm is labeled either as "NR", for the standard non-resampled version, or "R", the resampled one. The population size is also used to label the experiment. For instance, NR-200 corresponds to the standard version of the algorithm using a population size of 200.

Results are provided as descriptive statistics (Average, Medium, Variance) for the average prediction error across the 120 time periods for both risk and return objectives. Here, errors are defined in terms of the metrics described in Section 4.3. In order to simplify the comprehension, tables show descriptive statistics, across 20 experiments, for the average results for the metrics $ED_{E(R)}$ (Equation (7)) and ED_{σ^2} (Equation (8)). These are denoted as "Return" and "Risk", respectively. To compare the performance of the non-resampled and resampled versions of the algorithms, the difference between the errors obtained by both versions, labeled as *% Av.*, is also reported.

The first set of results is reported in Table 3. This table shows the results for NSGA-II. The advantage in terms of error distance of using resampling ranges from 17 % to 19 % and 16 % to 18 % for return and risk, respectively. The lowest prediction errors are achieved using the largest population size and also the difference between using resampling and the standard algorithm is most evident.

	Return				**Risk**			
	Average	Median	Variance	% Av.	Average	Median	Variance	% Av.
NR-50	2.4055	1.8437	5.0467		0.0215	0.0172	0.0012	
R-50	1.9848	1.4206	3.8320	−0.1749	0.0180	0.0134	0.0009	−0.1601
NR-200	2.4066	1.7901	5.2318		0.0216	0.0172	0.0013	
R-200	1.9901	1.4232	3.7390	−0.1731	0.0180	0.0133	0.0009	−0.1673
NR-500	2.4044	1.7462	5.3553		0.0217	0.0171	0.0014	
R-500	1.9574	1.3864	3.7524	−0.1859	0.0177	0.0130	0.0009	−0.1842

Table 3. Metrics values for Euclidean Distance – NSGA-II

Like NSGA-II, GDE3 also shows the inverse relationship between prediction error and population size. In global terms, GDE3 systematically beats NSGA-II in the standard configuration. Once we add resampling, the differences are less apparent as the algorithm reduces the prediction error to a lower extent. As we can see in Table 4, NSGA-II shows marginally better results in terms of risk for population sizes of 50 and 500.

	Return				**Risk**			
	Average	Median	Variance	% Av.	Average	Median	Variance	% Av.
NR-50	2.1653	1.5232	4.5713		0.0203	0.0161	0.0012	
R-50	1.9450	1.4249	3.8725	−0.1017	0.0181	0.0142	0.0012	−0.1067
NR-200	2.1703	1.5124	4.8341		0.0203	0.0162	0.0013	
R-200	1.9127	1.3676	3.7686	−0.1187	0.0178	0.0139	0.0011	−0.1243
NR-500	2.1607	1.5084	4.9065		0.0202	0.0160	0.0014	
R-500	1.8804	1.3360	3.7995	−0.1297	0.0178	0.0137	0.0011	−0.1208

Table 4. Metrics values for Euclidean Distance – GDE3

The third algorithm, SMPSO, offers intermediate results that are more in line with GDE3. The difference between the expected behaviour of the portfolios in the solution and the observed ones drops with resampling, but the improvement is slightly lower than that achieved by the differential evolution. Given that the starting errors (those offered by the basic algorithms) are very similar, these results are the second best alternative of the three mentioned. The results reported in Table 5 show improvements that range from 10 % to 13 %

	Return				Risk			
	Average	Median	Variance	% Av.	Average	Median	Variance	% Av.
NR-50	2.1890	1.5396	4.6426		0.0203	0.0161	0.0013	
R-50	1.9711	1.4408	3.9476	−0.0995	0.0182	0.0142	0.0012	−0.1039
NR-200	2.1808	1.5013	4.8605		0.0204	0.0163	0.0013	
R-200	1.9342	1.3855	3.7287	−0.1131	0.0179	0.0140	0.0011	−0.1201
NR-500	2.1841	1.5062	5.0983		0.0204	0.0162	0.0014	
R-500	1.9049	1.3750	3.8518	−0.1278	0.0179	0.0137	0.0011	−0.1209

Table 5. Metrics values for Euclidean Distance – SMPSO

Finally, the resampled version of SPEA2 seems to be the best option in this domain among the alternatives considered. Table 6 shows that the set of portfolios selected by the basic algorithm is the most unreliable. However, adding resampling reduces the average distance between the expected return and the real one up to 31 %. If we consider the differences in terms of risk, they drop to 29 %.

	Return				Risk			
	Average	Median	Variance	% Av.	Average	Median	Variance	% Av.
NR-50	2.5337	1.8637	5.8800		0.0225	0.0183	0.0014	
R-50	1.8286	1.3373	3.1472	−0.2783	0.0168	0.0127	0.0008	−0.2556
NR-200	2.5593	1.8341	6.0887		0.0229	0.0183	0.0015	
R-200	1.8317	1.3093	3.2972	−0.2843	0.0168	0.0123	0.0008	−0.2668
NR-500	2.5755	1.8190	6.2906		0.0232	0.0183	0.0016	
R-500	1.7800	1.2503	3.2099	−0.3089	0.0165	0.0120	0.0008	−0.2882

Table 6. Metrics values for Euclidean Distance – SPEA2

The differences between the errors shown by the basic and resampled versions of the algorithm, *% Av.*, were tested for statistical significance. We used the protocol described in Algorithm 5 and all the differences were significant at 1 %.

Algorithm 5 Statical testing protocol

```
if the values follow a Normal distribution of Gauss (applying the Kolmogorov-Smirnov test) then
    if the variances are homogeneity (the Levene test is used to check it) then
        A t-test is performed.
    else
        A Welch test is executed.
    end if
else
    A Wilcoxon test is applied to compare the medians of the solutions
end if
```

As we could see in the previous results, the suggested resampling mechanism resulted in a systematic and significant increase in the reliability of the solutions across the algorithms. The minimum decrease in average error was 10 % for SMPSO and the maximum gain was 31 % for SPEA2. This improvement comes at a low computational cost and no additional fitness evaluations. Furthermore, the differences in average errors were insignificant at 1 % regardless of the population size.

There is a potential explanation for relative gain of robustness of the basic versions of SPEA2 and NSGA-II vs. the resampled alternatives. The inspection of a sample of efficient frontiers for different experiments suggests that both SPEA2 and NSGA-II tend to provide fronts that are slightly wider on the high risk/high return side of the efficient frontier than the ones offered by the rest of the standard algorithms. Given that the portfolios located in that section are more prone to large deviations, this tends to result on a worse baseline. Furthermore, the differences in average errors were insignificant at 1 % regardless of the size. This is likely to make the improvement on a relative basis higher than average.

6 CONCLUSIONS

Portfolio optimization is one of the most active research lines in finance. The choice of the right combination of financial assets can be framed as a multiobjective optimization problem where the investor tries to find set of portfolios with the best risk/return profiles. These problems can be approached in different ways, but evolutionary multiobjective algorithms (MOEAs) are increasingly used as they provide the flexibility necessary to deal with real-world scenarios.

Portfolio managers often face the problem that the expected efficient frontier derived from their forecasts for future returns is subject to uncertainty. This means that if the real parameters differ from the forecasted ones, the risk and returns of the portfolios included in the estimated efficient frontier might deviate substantially from the predictions. This could result in extreme underperformance on some portfolios. This uncertainty is one of the major reasons why some practitioners mistrust quantitative methods based on modern portfolio theory, and the search for solutions has cleared the way for the development of robust portfolio optimization.

Robust optimization can not be addressed by traditional techniques, such as QP, because considering real-world constraints converts the basic portfolio into a more complex problem impossible to solve by QP. We presented a resampling mechanism based on a nonparametric bootstrap that, used in combination with MOEAs, should lower the likelihood of getting solutions whose behaviour, under normal circumstances, is much worse than anticipated. The mechanism resamples past data to generate different scenarios that are used during the evolutionary process to evaluate the portfolios that are added to the population. This prevents the efficient frontier from being specialized in a single scenario.

The approach was tested on real data on a sample of monthly returns for eight indexes representing different broad investment categories including stock, bonds, etc. In this study the performance of several multiobjective meta-heuristics has been compared in both basic and resampled versions. We have experimented with four multi-objective metaheuristics (NSGA-II, GDE3, SMPSO and SPEA2) over 120 time periods using a sliding window. The results show that resampling significantly enhances the reliability of the solutions as measured by average error committed predicting the performance of the final set of portfolios both in terms of risk and return. This is very important as it indicates that decision makers could rely more in the robust front to pick the right portfolio according to their preferences. The best results were obtained using SPEA2 and resampling, which resulted in improvements on the average reliability of the portfolios selected of up to 31 %.

Acknowledgements

The authors acknowledge financial support granted by the Spanish Ministry of Science under contracts TIN2008-06491-C04-03 (MSTAR) and TIN2011-28336 (MOVES), and Comunidad de Madrid (CCG10-UC3M/TIC-5029).

REFERENCES

[1] Anagnostopoulos, K. P.—Mamanis, G.: The Mean-Variance Cardinality Constrained Portfolio Optimization Problem: an Experimental Evaluation of Five Multiobjective Evolutionary Algorithms. Expert Systems with Applications (to appear – Corrected Proof).

[2] Barbosa, H. J. C.—Lemonge, A. C. C.: A Genetic Algorithm Encoding for a Class of Cardinality Constraints. Proceedings of the 2005 Conference on Genetic and Evolutionary Computation (GECCO '05), pp. 1193–200, New York, NY, USA 2005.

[3] Chiranjeevi, Ch.—Sastry, V. N.: Multi Objective Portfolio Optimization Models and its Solution Using Genetic Algorithms. International Conference on Computational Intelligence and Multimedia Applications, Vol. 1, 2007, pp. 453–457.

[4] Cai, X.—Wei, O.—Huang, Z.: Evolutionary Approaches for Multi-Objective Next Release Problem. Computing and Informatics, Vol. 31, 2012, No. 4, pp. 847–875.

[5] Evolutionary multi-Objective Optimization: A Historical View of the Field. IEEE Computational Intelligence Magazine, Vol. 1, 2006, No. 1, pp. 28–36.

[6] Deb, K.—Pratap, A.—Agarwal, S.—Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, Vol. 6, 2002, No. 2, pp. 182–197.

[7] Deb, K.—Steuer, R. E.—Tewari, R.—Tewari, R.: Bi-Objective Portfolio Optimization Using a Customized Hybrid NSGA-II Procedure. Evolutionary Multi-Criterion Optimization (EMO), LNCS Vol. 6576, pp. 358–373, Springer 2011.

[8] Durillo, J. J.—Nebro, A. J.—Alba, E.: The JMetal Framework for Multi-Objective Optimization: Design and Architecture. IEEE Congress on Evolutionary Computation (CEC 2010), LNCS, Vol. 5467, pp. 4138–4325, Springer 2010.

[9] García, S.—Quintana, D.—Galván, I. M.—Isasi, P.: Portfolio Optimization Using SPEA2 with Resampling. Proceedings of the 12th International Conference on Intelligent Data engineering and Automated Learning (IDEAL '11), Springer-Verlag 2011, pp. 127–134.

[10] Markowitz, H. M.: Portfolio Selection: Efficient Diversification of Investments. John Wiley & Sons 1959.

[11] Idzorek, T. M.: Developing Robust Asset Allocations. Working Paper, Ibbotson Associates, Chicago 2006.

[12] Kukkonen, S.—Lampinen, J.: GDE3: The Third Evolution Step of Generalized Differential Evolution. In IEEE Congress on Evolutionary Computation (CEC 2005), pp. 443–450.

[13] Markowitz, H.: Portfolio selection. The Journal of Finance, Vol. 7, 1952, No. 1, pp. 77–91.

[14] Nebro, A. J.—Durillo, J. J.—García-Nieto, J.—Coello Coello, C. A.—Luna, F.—Alba, E.: SMPSO: A New PSO-Based Metaheuristic for Multi-Objective Optimization. In 2009 IEEE Symposium on Computational Intelligence in Multicriteria Decision-Making (MCDM 2009), IEEE Press, 2009, pp. 66–73.

[15] Wei, J.—Wang, Y.—Wang, H.: A Hybrid Particle Swarm Evolutionary Algorithm for Constrained Multi-Objective Optimization Computing and Informatics, Vol. 29, 2010, No. 5, pp. 701–718.

[16] Perret-Gentil, C.—Victoria-Feser, M.: Robust Mean-Variance Portfolio Selection. Fame research paper series, International Center for Financial Asset Management and Engineering, April 2005.

[17] Pug, G.—Wozabal, D.: Ambiguity in Portfolio Selection. Quantitative Finance, Vol. 7, 2007, No. 4, pp. 435–442.

[18] Radziukyniene, I.—Xilinskas, A.: Evolutionary Methods for Multi-Objective Portfolio Optimization. In S. I. Ao, Len Gelman, David WL Hukins, Andrew Hunter, and A. M. Korsunsky (Eds.): Proceedings of the World Congress on Engineering 2008, Vol. II, WCE '08, July 2–4, 2008, London, U.K., Lecture Notes in Engineering and Computer Science, International Association of Engineers, Newswood Limited 2008, pp. 1155–1159.

[19] Reyes, M.—Coello Coello, C. A.: Improving PSO-Based Multi-Objective Optimization Using Crowding, Mutation And α-dominance. In C. A. Coello, A. Hernández, E. Zitler (Eds.): Third International Conference on Evolutionary MultiCriterion Optimization, EMO 2005, LNCS, Vol. 3410, Springer 2005, pp. 509–519.

[20] Ruppert, D.: Statistics and Finance: An Introduction. Journal of the American Statistical Association, Vol. 101, pp. 849–850, June 2006.

[21] Schaffer, J. D.: Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. Proceedings of the First International Conference on Genetic Algorithms, L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 93100, 1985.

[22] Shiraishi, H.: Resampling Procedure to Construct Estimation Error Efficient Portfolios for Stationary Returns of Assets. J. Japan Statist. Soc., Vol. 40, 2010, No. 2, pp. 189–206.

[23] Skolpadungket, P.—Dahal, K.—Harnpornchai, N.: Portfolio Optimization Using Multiobjective Genetic Algorithms. In Proceeding of 2007 IEEE Congress on Evolutionary Computation 2007, pp. 516–523.

[24] Soleimani, H.—Golmakani, H. R.—Salimi, M. H.: Markowitz-Based Portfolio Selection with Minimum Transaction Lots, Cardinality Constraints and Regarding Sector Capitalization Using Genetic Algorithm. Expert Syst. Appl., Vol. 36, pp. 5058–5063, April 2009.

[25] Tütüncü, R. H.—Koenig, M.: Robust Asset Allocation. Annals OR, Vol. 132, 2004, No. 1-4, pp. 157–187.

[26] Hamdan, M.: On the Disruption-Level of Polynomial Mutation for Evolutionary Multiobjective Optimisation Algorithms. Computing and Informatics, Vol. 29, 2010, No. 5, pp. 783–800.

[27] Zitzler, E.—Laumanns, M.—Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland 2001.

[28] Zitzler, E.—Thiele, L.: An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland 1998.

Sandra Garcia Rodriguez graduated in computer science engineering from Carlos III University of Madrid. After the five-year engineering she studied a specialization master on artificial intelligence and then the Ph. D. in the same area. From September 2007 to April 2013 she worked at the Department of Computer Science, University Carlos III of Madrid, as a Professor and researcher participating in several international projects. Since May 2013 she has been working in the Department of Computer Science at Complutense University of Madrid. Her current research focuses in artificial neural networks, evolutionary computation techniques (such as genetic algorithms, evolutionary strategies, and particle swarm), multi-objective optimization and robust portfolio optimization.

David Quintana is currently an Interim Associate Professor in the Department of Computer Science at Universidad Carlos III (UC3M) de Madrid, where he is part of the EVANNAI research group. He received his Ph. D. in finance from Universidad Pontificia Comillas (ICADE) and his Master in computer science, with a specialization in artificial intelligence, from UC3M. His research is mainly focused in applications of soft computing, mostly evolutionary computation and artificial neural networks, in finance and economics.

Inés M. Galván received her Ph. D. degree in computer science from Universidad Politécnica de Madrid, Spain, in 1998. From 1992 to 1995, she received a Doctorate-Fellowship, as a Research Scientist, in the European Commission, Joint Research Center Ispra, Italy. Since 1995, she has been with the Department of Computer Science, University Carlos III of Madrid, where she has been an Associate Professor since 2000. Her current research focuses in artificial neural networks, evolutionary computation techniques (such as genetic algorithms, evolutionary strategies, and particle swarm) and multi-objective optimization. Her research interests also cover applications fields, such as classification, times series prediction, brain computer interfaces and finances.

Pedro Isasi received his Ph. D. degree in computer science from Universidad Politécnica de Madrid (UPM), Madrid, Spain, in 1990. He is currently a Professor in computer science with University Carlos III of Madrid, Madrid, Spain, where he is also the Head of the Department of Computer Science and Founder and Director of the Neural Network and Evolutionary Computation Laboratory. His principal research is in machine learning, metaheuristics, and evolutionary optimization methods, mainly applied to the field of finance and economics, forecasting, and classification.