www.cefin.unimore.it

CEFIN Working Papers

No 7

Differential Evolution
for Multiobjective Portfolio Optimization

by Thiemo Krink and Sandra Paterlini

January 2008

CEFIN – Centro Studi di Banca e Finanza
Dipartimento di Economia Aziendale – Università di Modena e Reggio Emilia
Viale Jacopo Berengario 51, 41100 MODENA (Italy)
tel. 39-059.2056711 (Centralino) fax 059 205 6927

Differential Evolution for Multiobjective Portfolio Optimization

Thiemo Krink[a, b], Sandra Paterlini[c]

[a] *Allianz Investment Management (AIM), Group Investments, Munich*
[b] *EVALife, Dept. of Computer Science, Univ. of Aarhus, Denmark*
[c] *Department of Economics, CEFIN – Center for Research in Banking and Finance & RECent-Center for Economic Research, Univ. of Modena and Reggio E., Italy*

Abstract: Financial portfolio optimization is a challenging problem. First, the problem is multiobjective (i.e.: minimize risk and maximize profit) and the objective functions are often multimodal and non smooth (e.g.: value at risk). Second, managers have often to face real-world constraints, which are typically non-linear. Hence, conventional optimization techniques, such as quadratic programming, cannot be used. Stochastic search heuristic can be an attractive alternative. In this paper, we propose a new multiobjective algorithm for portfolio optimization: DEMPO - Differential Evolution for Multiobjective Portfolio Optimization. The main advantage of this new algorithm is its generality, i.e., the ability to tackle a portfolio optimization task as it is, without simplifications. Our empirical results show the capability of our approach of obtaining highly accurate results in very reasonable runtime, in comparison with quadratic programming and another state-of-art search heuristic, the so-called NSGA II.

Key-words: Portfolio optimization, multiobjective, real-world constraints, value at risk, expected shortfall, differential evolution.

1 Introduction

Since the pioneer work of Markowitz (Markowitz 1952), modern portfolio optimization has become a core topic for academics and practitioners. In the classic Markowitz mean-variance approach, the target is to find the asset allocation which minimizes the risk and maximises the return. Since asset returns are assumed to be normally distributed, the variance-covariance matrix of the asset allocations is used to quantify the risk, while the expected returns quantify the profit. Such an optimization problem can be solved efficiently using a multiobjective technique related to the epsilon constraint method (Haimes et al. 1971, Chankong and Haimes 1983), which generates n points of the Pareto front by iteratively solving n single-objective problems with constraints using linear and quadratic programming.

However, the standard Markowitz mean-variance model is simplistic and cannot handle more realistic measures of risk and typical real world constraints. In particular, the Markowitz approach has been criticized mainly for two reasons. First, the assumption of normality of financial returns does not hold (Cont 2001), because the empirical distribution is typically leptokurtic and has fat tails. Instead one has to consider either higher moments of a probability distribution or use a non-parametric approach. Hence, a possible approach could be to model returns by elliptical and asymmetric stable distributions (see Lamantia et al. 2006, Ortobelli et al. 2004, Doganoglu et al. 2006). Another approach can be to use non-parametric approaches to generate returns scenarios: then, in such case, most alternative risk measures, such as Value-at-Risk, require a specification that cannot be stated by a quadratic equation (Gaivoronski and Pflug, 2004).

Second, there is often the need to introduce additional non-linear constraints. Typical constraints are so-called cardinality constraints (e.g.: an upper bound for the number of assets in the portfolio), buy-in thresholds (i.e.: an asset can be included in the portfolio only if its amount is bigger/smaller than a lower/upper bound), and roundlots (i.e.: the smallest volume of an asset that can be bought) (Chang et al. 2000, Streichert 2003). Such constraints are non-linear and cannot be solved by conventional optimization methods, such as quadratic programming.

Research has mainly focused on using mixed integer solvers or metaheuristics. Mixed-integers optimization problems are optimization problems with some discrete decision variables. Branch and bound and branch and cut in combination with LP/QP are examples of mixed integer solvers (Meyer 1975). An alternative are metaheuristics, which are heuristics that are often capable to solve a broad class of optimization problems for which there is no satisfactory-problem specific algorithm. They are usually inspired by some natural phenomena such as simulating annealing, evolutionary algorithms and ant colony optimization.

In this paper, we propose a new approach based on evolutionary algorithms, which are population based search heuristics. Such heuristics work by iteratively evolving a population of candidate solutions to the optimization problem towards better solutions until a stopping criterion is satisfied. Among many other applications, evolutionary algorithms have been widely applied to multiobjective optimization problems (Deb 2001, Coello Coello et al. 2002, Coello Coello 1999). Their main advantage is that they can tackle optimization problems as they are without requiring rigid properties, such as continuity, linearity or convexity of objective functions and constraints. Moreover their population of candidate solutions can be used to search for a set of representative solutions of the Pareto front simultaneously.

Our novel multiobjective search heuristic, which we will call henceforth DEMPO (Differential Evolution for Multiobjective Portfolio Optimization) is partly based on differential evolution (DE) (Storn and Price 1997, Price et al. 2004) and inspired by the NSGA-II algorithm by Deb et al. (Deb et al. 2002, Srinivas and Deb 1994). Moreover, Michalewicz's GENOCOP approach (Michalewicz and Fogel 2004) has been the starting point to develop our constraint handling techniques, and finally we implemented a custom made seed initialization approach.

The following sections are organized as follows. Section 2 reports the formal specification of portfolio optimization problems considered in this work; Sections 3 describes the DEMPO algorithm; Section 4 describes the experimental data and settings and reports the empirical results and Section 5 concludes our study. Appendix A shows the comparison of using DEMPO and NSGA II algorithms on six multiobjective benchmark problems.

2 The Portfolio Optimization Problem

2.1 Formal Specification

Let us consider the following variables

N	the number of available assets
n_s	the number of scenarios for the returns
G	the number for asset classes
r_{si}	the return of asset i in the sth scenario, s=1,...,n_s, i=1,...,N
μ_i	the expected monthly return of asset i $\mu_i = \sum_{s=1}^{n_s} r_{si} / n_s$
σ_{ij}	the covariance between the ith and the jth assets
Π_0	the portfolio value at time 0
Π_s	the portfolio value at time 1 for scenario s
K	the number of assets to invest $(K \le N)$
ε_i	the minimum investment ratio allowed in the ith asset ($i=1,...,N$)
δ_i	the maximum investment ratio allowed in the ith asset ($i=1,...,N$)
Δ_i	the minimum percentage change w.r.t. the previous allocation
ζ_g	the minimum investment ratio allowed in the gth class ($g=1,...,G$)
ϑ_g	the maximum investment ratio allowed in the gth class ($g=1,...,G$)
TR	the maximum turnover ratio

$$z_i = \begin{cases} 1 & \text{if the } i\text{th asset is chosen} \\ 0 & \text{otherwise} \end{cases}$$

$$y_g = \begin{cases} 1 & \text{if assets from the } g\text{th class are chosen} \\ 0 & \text{otherwise} \end{cases}$$

w_i	the portfolio weight $0 \le w_i \le 1, \sum_{i=1}^{N} w_i = 1$ of the ith asset

The constrained portfolio optimization problem a-la-Markowitz can be formulated in the following way:

MV- Mean-Variance

$$(1a) \qquad \min_{\mathbf{w}'} \quad f_1(\mathbf{w}') = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij}$$

$$(1b) \qquad \max_{\mathbf{w}'} \quad f_2(\mathbf{w}') = \sum_{i=1}^{N} w_i \mu_i$$

$$subject \quad to:$$

$$(1c) \ \sum_{i=1}^{N} w_i = 1 \quad 0 \le w_i \le 1$$

The optimization problem as stated in equation (1*a*) and (1*b*) is a multiobjective optimization problem with the two competing objectives of minimizing the variance of the portfolio returns (the risk) and maximizing the expected return of the portfolio (the profit). Considering only constraint (1*c*), we can solve this problem using quadratic and linear programming (QP/LP). However, such a formal specification is insufficient to solve realistic portfolio optimization problems in which managers and investors have often to consider different measures for risk and real-world constraints.

Nowadays, value at risk (VaR) and expected shortfall (ES) are among the most widely used measures of the portfolio risk. Value at risk ($VaR_{(1-\alpha)}$) is the α-quantile ($VaR_{1-\alpha} = Q_L(\alpha)$, e.g.: α=5%) of the distribution of the losses (L) of the portfolio, while the expected shortfall is the conditional mean value of the losses given that the losses have exceeded $VaR_{(1-\alpha)}$, that is $ES_{1-\alpha} = E(L \mid L < VaR_{1-\alpha})$.

We use a non parametric approach in order to compute $VaR_{(1-\alpha)}$ and $ES_{(1-\alpha)}$ (Gilli et al. 2006). The distribution of the losses has been computed by using the scenarios generated from time series data (see section 4.1 for the full description of the data). The loss L_s for scenario s=1,..., n_s has been defined as

$$L_s = \Pi_s - \Pi_0$$

where the portfolio value at time 1 is $\Pi_s = \Pi_0 e^{\sum_{i=1}^{N} w_i r_{si}}$ and Π_0 is the current portfolio value at time 0.

Then, we order the n_s simulated losses such that $L_1 \leq L_2 \leq \ldots \leq L_{ns}$ and we compute the value at risk and the expected shortfall according to

$$VaR_{1-\alpha} = L_{\lceil \alpha n_s \rceil} \quad \text{and} \quad ES_{1-\alpha} = \frac{\sum_{s=1}^{n_s} L_s \mathbf{1}_{\{L_s < VaR_{1-\alpha}\}}}{\sum_{s=1}^{n_s} \mathbf{1}_{\{L_s < VaR_{1-\alpha}\}}} \text{ where } \mathbf{1}_{\{L_s < VaR\}} = \begin{cases} 1 \text{ if } L_s < VaR_{1-\alpha} \\ 0 \quad \text{otherwise} \end{cases}$$

Expected shortfall was mainly introduced in order to promote diversification since value at risk is not a coherent risk measure and a concave function of allocation (Artzner et al. 1997). Figure 1.a shows the non-concave objective function for VaR_{95} for a portfolio of three Italian equities (AS Roma, Acea e Acotel Group), while figure 1.b shows the concave Expected Shortfall function ES_{95}. VaR and Expected Shortfall are computed on a grid of 50x50 points for w_1 and w_2, while w_3 is determined by the budget constraint when short selling is not allowed.

(a) **(b)**

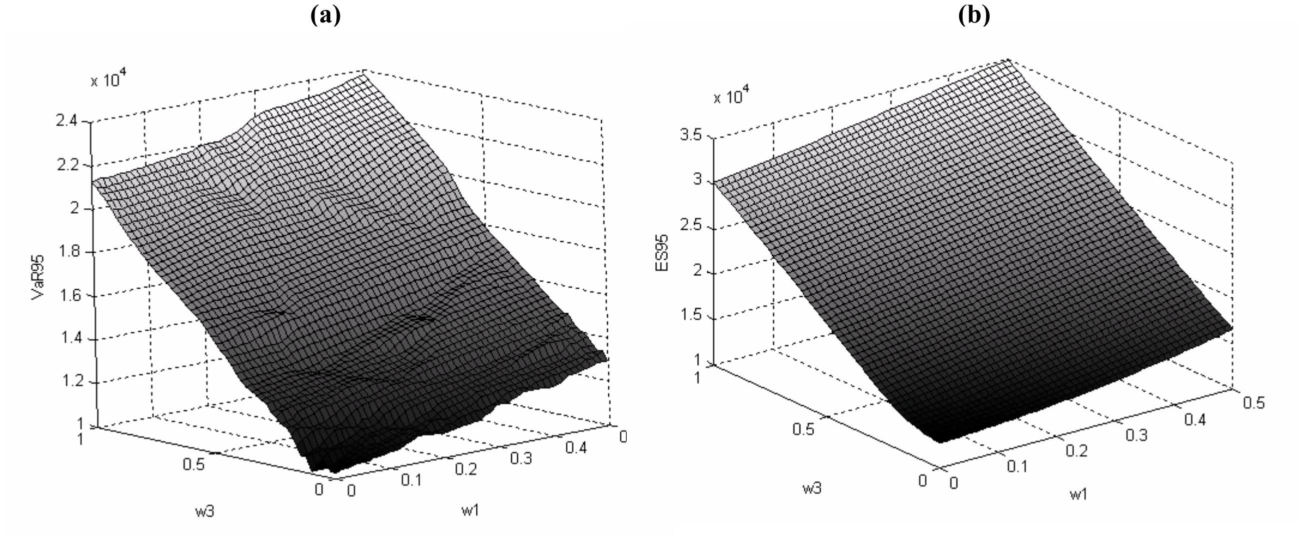

Figure 1: Objective function for VaR minimization (Figure 1.a) and for ES minimization (Figure 1.b) for a portfolio of three assets.

When such risk measures are introduced and we want to use a non-parametric approach to estimate them, it is not possible to use quadratic programming (Gaivoronski and Pflug, 2004). In contrast, the DEMPO algorithm can easily tackle the following multiobjective portfolio optimization problems:

MVaR- Mean-VaR$_{(1-\alpha)}$

$$(2a) \qquad \min_{\mathbf{w}'} \quad f_1(\mathbf{w}') = \overline{L} - VaR_{1-\alpha}$$

$$(2b) \qquad \max_{\mathbf{w}'} \quad f_2(\mathbf{w}') = \overline{L}$$

subject to:

$$(2c) \quad \sum_{i=1}^{N} w_i = 1 \quad 0 \le w_i \le 1$$

MES- Mean-ES$_{(1-\alpha)}$

$$(3a) \qquad \min_{\mathbf{w}'} \quad f_1(\mathbf{w}') = \overline{L} - ES_{1-\alpha}$$

$$(3b) \qquad \max_{\mathbf{w}'} \quad f_2(\mathbf{w}') = \overline{L}$$

subject to:

$$(3c) \quad \sum_{i=1}^{N} w_i = 1 \quad 0 \le w_i \le 1$$

where $\overline{L} = \sum_{i=1}^{n_s} L_s \Big/ n_s$ corresponds to the expected absolute return of the portfolio.

Finally, the DEMPO algorithm can tackle real-world constraints that managers have often to consider. Such constraints are: each asset can be included only if its weight is greater/smaller than its minimum/maximum investment ratio (4), assets from the same asset class cannot be considered if the class total investment is smaller/larger than the minimum/maximum investment ratio (6), the change in the asset weight from previous allocation must be greater than a certain threshold (5), the sum of the absolute change from the previous allocation must be smaller than the maximum turnover ratio (7).

Real-World Constraints

$$(4) \quad \forall \mathrm{i}: -1 \le \varepsilon_i z_i \le w_i \le \delta_i z_i \le 1$$

$$(5) \quad \forall \mathrm{i}: \left| w_i - w_{\mathrm{i}}^{'} \right| \ge \Delta_i \ \text{ or } \ \left| w_i - w_{\mathrm{i}}^{'} \right| = 0$$

$$(6) \quad \forall j \in G_g : \xi_g y_g \le \sum_{j \in G_g} w_j^{'} \le \vartheta_g y_g \ \text{ with } \ -1 \le \xi_g \le \vartheta_g \le 1$$

$$(7) \quad \sum_{i=1}^{N} \left| w_i - w_i^{'} \right| \le TR$$

$\mathbf{w}$: new asset allocation, $\mathbf{w}'$: previous asset allocation

3 The DEMPO Algorithm

3.1 Multiobjective Portfolio Optimization and Evolutionary Approaches

Evolutionary algorithms are population based search heuristics. They work by evolving a population of candidate solutions for the optimization problem. The evolution is driven by mathematical operators inspired by evolutionary biology. Among many other applications, they have been widely used in tackling multiobjective optimization problems, since, in contrast to standard mathematical programming techniques, they iteratively approximate the whole Pareto front simultaneously, they are not concern with the shape and continuity of the Pareto front, they can be easily applied even if the objective functions change and they can easily integrate special constraints. On the other hand, they are often criticized because of the extensive parameter tuning that is often required.

The most popular approaches in evolutionary multiobjective optimization are: aggregating functions, Schaffer's VEGA, Fonseca and Fleming's MOGA, Srinivas and Deb's NSGA, Horn and Nafpliotis'NPGA and Target vector approaches. The reader is referred to (Deb 2001, Coello Coello et al. 2002, Coello Coello 1999) for surveys about evolutionary multiobjective optimization techniques.

Differential Evolution (DE) is a rather new evolutionary algorithm for numerical optimization (Storn and Price 1997). It is simple to implement, requires little or no parameter tuning, and is known for remarkable performance and its superiority compared to other evolutionary algorithms, such as genetic algorithms or particle swarm optimization, in real-world and artificial problems. (see Price et al. 2004, Lampinen 2006, for reviews).

The Pareto-frontier Differential Evolution (PDE) by Abbass et al. (2001) has been, to our knowledge[1], the first paper on Multiobjective DE. Recently, other studies have proposed to use DE in multiobjective optimization problems (see for example, Sarker and Abbass 2002, 2004, Babu et al. 2005, Quintero and Coello Coello 2005). Comparisons on benchmark problems show the DE superiority with respect to other evolutionary algorithms, such as Strength Pareto

[1] We thank a referee for suggesting us this reference.

Evolutionary Algorithm (SPEA), Fonseca and Fleming's genetic algorithm (FFGA), Hajela's and Lin's genetic algorithm (HLGA), Niched Pareto Genetic Algorithm (NPGA), Non-dominated Sorting Genetic Algorithms (NSGA), Random Sampling Algorithm (RAND), Single Objective Evolutionary Algorithm (SOEA), Vector Evaluated Genetic Algorithm (VEGA) and Pareto Archived Evolution Strategy (PAES) (Sarker and Abbass 2002, 2004).

While evolutionary algorithms, in particular genetic algorithm, have found wide application in tackling the financial multiobjective portfolio optimization problem (Dueck and Winker 1992, Bertocchi and Giacometti, 1993, Vedarajan et al. 1997), even considering cardinality constraints (Chang et al. 2000, Streichert et al. 2003), there has been no work on using differential evolution for multiobjective portfolio optimization with real-world constraints to our knowledge.

3.2 The DEMPO Algorithm

The DEMPO algorithm is partly based on differential evolution (DE) (Storn and Price 1997, Price et al. 2004) and inspired by the NSGA-II algorithm by Deb et al. (Deb et al. 2002, Srinivas and Deb 1994). Moreover, Michalewicz's GENOCOP approach (Michalewicz and Fogel 2004) has been the starting point to develop our constraint handling techniques.

3.2.1 Diversity Preservation

One key aspect in using evolutionary algorithm is to have a mechanism that allow to have diversity in the population such that a good spread of solutions is maintained in the population. We use the crowding distance approach, proposed by Deb et al. (2002). After determining and sorting the non-dominated fronts (see Figure 2.a), for each solution we calculate the cuboid distance, i.e., the mean of the lower and upper distance for each objective f_i, (i=1,2) to the nearest two solutions within the same front. The distance is an estimate of the perimeter of the cuboid formed by considering the nearest neighbors as vertices (see Figure 2.b). Then, during selection between two individuals j and k, we select the one which is dominated by less other solutions and if they are dominated by the same number of solutions then we select the one with the larger distance value.

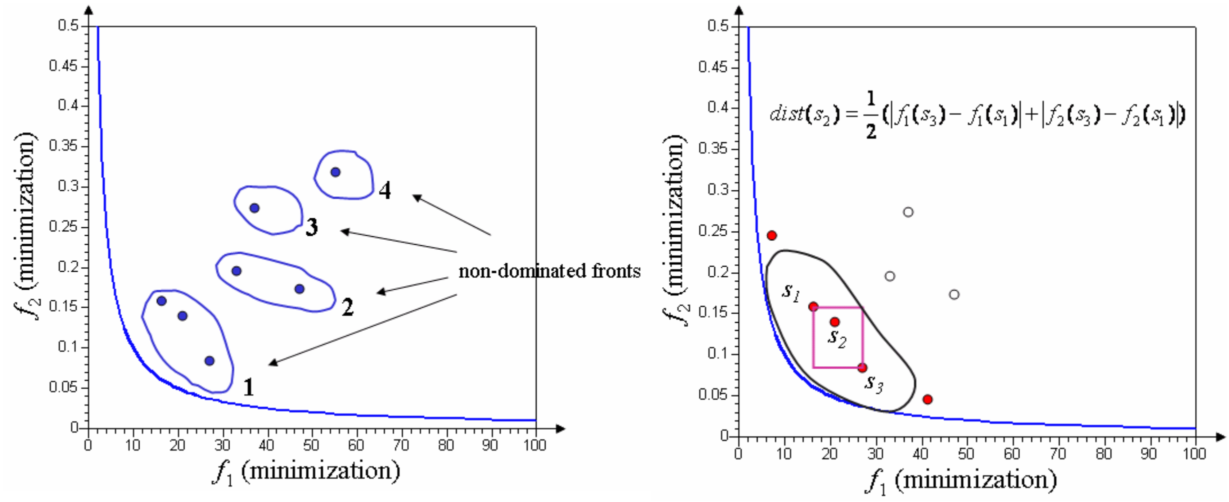

Figure 2: Determination of non-dominated fronts (a), and cuboid-distance calculation (b).

3.2.2 Constraints Handling

In order to handle inequality constraints, we rewrite the definition of *domination*, such that it includes the constraint handling (Deb et al. 2002). Then, a solution *i* is said to constrained-dominates a solution *j*, if any of the following conditions is true:

1. solution *i* is feasible and solution *j* is not
2. solutions *i* and *j* are both infeasible, but *i* has a smaller overall constraint violation
3. solutions *i* and *j* are both feasible and *i* dominates *j*

We deal with equality constraints by construction of feasible solutions or "repair" existing ones, i.e., a solution that violates constraints is modified until it is feasible. Such approach is inspired by Michalewicz's GENOCOP approach (Michalewicz and Fogel 2004) and is case to case dependent.

3.2.3 Main Loop

After loading and setting the problem specification and the search heuristic parameters, the population is initialized and evaluated with respect to the fitness function. Then, iteratively the population is evolved with Rand/1/Exp and crossover operators (Storn and Price 1997), and the non-domination ranking is computed. Cuboids (Deb et al. 2002) are computed in order to obtain diversification of the solutions and then sorted and ranked. The candidates with equal non-domination ranking are selected and the individual with the worst cuboid is replaced. The front is then determined. The algorithm stops after a fixed number of iterations.

Figure 3 shows the pseudo-code of DEMPO algorithm.

```
ExecDEMPO()
  Load and set problem specification
  Set DEMPO search heuristic parameters
  Calculate candidate solution seeds (optional)
  DEMPO(...)
    Initialize local variables including curIt = 1
    InitializePopulation(...)
    Initialize the candidate solutions randomly
    HandleEqualityConstraints(…) % "repair" solutions if necessary
    Insert seeds if generated previously
    for i=1:popSize
      FitnessFunc(...)  % evaluate the candidate solutions
    end
    while curIt<numIt
      for i=1:popSize
        Create new candidate with DE operators "Rand/1/Exp" and crossover
        HandleEqualityConstraints(…) % "repair" candidate if necessary
        FitnessFunc(...)  % evaluate the candidate
      end
      Calculate non-domination ranking to allow selection for multiple
      Objectives
      Calculate the cuboids to obtain diversification of the solutions
      Initialize the cuboids using normalized fitnesses
      Sort individuals by their fitness for each obj. and calculate cuboid
      distances
      Sort all cuboids back, such that their order corresponds to their id
      Rank the cuboids of the current population, i.e., smallest rank and
       smallest cuboid first
      Consider all candidates with equal non-domination ranking and determine
      the ids of individuals in the parent pop. with equal front rank
      Replace the individual that has the worst cuboid in the same front
      Determine the front
    end
```

Figure 3: Pseudo-code of DEMPO algorithm.

4 Experiments and Results

4.1 Benchmarks and financial dataset

In our experiments, we investigated the potential of DEMPO for multiobjective portfolio optimization by a comparison with the NSGA-II algorithm by Deb et al. (2002) and QP for a quadratic programming instance of the standard mean-variance portfolio optimization problem based on real data. The financial input dataset is a set of time series with daily observations for five years between 01/10/2001 and 02/10/2006 for 219 stocks in the Mibtel index traded on the Italian Stock Exchange. Data have been downloaded from Datastream.

Following the approach of Gilli, Kellezi and Hysi (2006), we did not assume that returns are normally distributed and we generate returns scenarios from asset prices. Our investment planning holding period is one month. The set of return scenarios has been created by bootstrapping 800 overlapping blocks of length 20 from the matrix of daily log-returns. The sum of the log-return, $r_{s.}$, for each block defines a monthly log-return scenario. The set of monthly scenarios is used to compute the expected returns $\mu_{.}$, the covariance matrix $\Sigma=[\sigma_{..}]$, the value at risk ($VaR_{(1-\alpha)}$) and the expected shortfall ($ES_{(1-\alpha)}$) in order to find the optimal portfolio.

Section 2.1 describes the portfolio optimization problem specification.

Moreover, we compared DEMPO with the NSGA-II for 6 classic, rather simplistic, multiobjective optimization benchmark problems. Appendix A reports the description and the empirical results of the multiobjective optimization benchmark problems.

4.2 Comparison Criteria

The comparison of results in multiobjective optimization is not a straight-forward task. In contrast to single objective optimization, the goal is to identify a set of candidate solutions that represents the Pareto front as good as possible. The main goals in multiobjective optimization are the convergence to the Pareto front and a good coverage of the Pareto optimal set. The true Pareto front is in fact a dense set. We try a discrete sampling of this set. These goals cannot be achieved with one single metric and many different metrics have been proposed. The reader is referred to Sarker and Coello (2002) for a review.

One possibility is to measure how close the evolved solutions are to the true Pareto front and how evenly they are scattered along the front. This of course requires that the true Pareto front is known, which is usually only the case for simplistically constructed benchmark problems. The point in real-world optimization is that the front is not known and shall be determined or at least approximated. However, another possibility is to use such measures to compare the fronts derived by two different algorithms in which one is used as a reference. In case of portfolio optimization, a QP solution to a quadratic programming problem instance can be considered as such a reference. For realistic non-QP applications, such a comparison is not possible. However, for performance comparison, i.e., to test the accuracy and robustness of the results, it can make sense to investigate a QP problem instance.

Deb et al. (2002) defined two criteria that can capture the notion of distance to a front and good coverage of the front to which they refer as gamma and delta.

Gamma is the average Euclidean distance of the minimum Euclidean distance between each solution obtained with the algorithm and the uniformly spaced solutions on the Pareto-optimal front. It measures the extent of convergence to a known set of Pareto-optimal solution and the smaller the value of this metric, the better the convergence toward the Pareto-optimal front.

Delta aims at quantifying the extent of spread achieved among the obtained solutions. Delta is defined as

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{M-1} | d_i - \bar{d} |}{d_f + d_l + (M-1)\bar{d}}$$

where d_i is the Euclidean distance between consecutive solutions in the nondominated set of M solutions, $\bar{d}$ is the average of these distances, d_f and d_l are the Euclidean distances between the extreme solutions and the boundary solutions of the nondominated set. If delta is close to zero, the set of nondominated solution is widely and uniformly spread out.

Another useful criterion, which does not require a known or reference set of Pareto optimal solutions, is to consider the area (or hyper-volume for more than two objectives) that the front of solutions is covering. If the Pareto front is known or defined as a reference, we calculate the percentage of area, i.e., the ratio of covered area of an algorithm and the covered area of the known Pareto front.

Evolutionary algorithms are usually run several times to end up with a set of alternative solutions for the problem at hand from which one has to be chosen. Hence, Fonseca and Fleming (1996) proposed to compare different algorithms statistically, instead of comparing only scalar values. They propose the so-called "attainment surfaces" method, which performs a statistical comparison of two algorithm on several runs. The method draws two attainment surfaces, one for each algorithm under investigation. Each surface divides then the objective space into two regions: one that contains the vectors which are dominated by the results of the algorithm, and another one that contains the vectors that dominate the results of the algorithm. Then, a number of sampling lines is drawn from the origin in order to intersect the attainment surfaces and for each sampling line (assuming minimization for both objectives) the intersection of an algorithm closer to the origin is the winner. The idea is then to consider a set of sampling lines which intersect the attainment surfaces across the full range of the Pareto frontier. Standard non-parametric statistical test can then be performed on the distribution of the intersections of the sampling line and the attainment surfaces of different runs. Knowles and Corne (2000) proposed to use the Mann-Whitney rank test to determine whether or not the intersection for one of the algorithms over different runs occur closer to the origin, providing the percentages of the surface in which each algorithm outperform the other at a chosen level of significance. Furthermore, Knowles and Corne (2000) also extended their method, henceforth MOSTATS[2], in order to compare more than two algorithms.

[2] We thank one of the referee for suggesting this comparison methodology. The code of MOSTATS can be downloaded at http://dbkgroup.org/knowles/multi/

In this study, we use all four comparison methods described above: delta, gamma, area covered and MOSTATS. The reader is referred to Sarker and Coello (2002) for a more detailed description of the assessment methodologies for multiobjective evolutionary algorithms.

4.3 Algorithmic settings and experimental set-up

Both NSGA-II and DEMPO were compared using a population size of 100 individuals and 500 iterations for all experiments, which were rather arbitrary settings that correspond to the experimentation of Deb et al. (2002). By keeping these values constant, we used equivalent number of fitness function evaluations for both algorithms. A summary of all algorithmic parameters is shown in Table 1.

Table 1: Parameters of Differential Evolution (DE) and NSGA II. *Pop.Size* refers to the population size, *Num.Gen.* to the number of generations, *M* to the number of points on the frontier, *N* to the number of decision variables. For the DE: *cr*: crossover rate; *f*: scaling factor. For NSGA II: *pc*: crossover probability; *pm*: mutation probability.

DE		**NSGA II**	
Parameter	Value	Parameter	Value
Pop.Size	100	*Pop.Size*	100
Num.Gen.	500	*Num.Gen.*	500
M	100	*M*	100
cr	(0.2-0.9)	*pc*	(0.3-0.9)
f	0.3	*pm*	*1/N*

We repeated each experiment for portfolio optimization 30 times to check the robustness of the results. Empirical results for portfolio optimization are reported in section 4.6, while the reader is refereed to Appendix A for results on multiobjective benchmark functions.

4.4 Preliminary experimentation and tuning of the algorithms

LP and QP do not require any parameter tuning other than to increase the maximum number of iterations, such that the algorithms can obtain a correct result. Our NSGA-II implementation is based on the code by A. Sheshadri[3]. We experiment with different values of the probability of crossover (from 0.3 to 0.9, with stepsize 0.1) and perform statistical comparison by MOSTATS over 30 runs for each parameter setting. Table 2.a shows that the percentage of space on which we can be 95% confident that each algorithm beats all of the other algorithms compared is always zero (row: "beats all"), and that the best result with respect to the percentage of the space for which we cannot be 95% confident that any other algorithm beats it, is obtained for a probability of crossover equal to 0.7 (i.e.: unbeaten: 89.4), while the worst result correspond to a probability of crossover of 0.4 (i.e.: unbeaten: 30.3). The reader should notice that there is not a monotonic relationship between the performance and the probability of crossover.

Regarding the DEMPO algorithm, we fixed the scaling factor *f* equal to 0.3 and we vary the crossover rates *cr* from 0.2 to 0.9 with stepsize 0.1. Table 2.b reports the statistical comparison over 30 runs for each parameter setting for DEMPO. The row "unbeaten" show the better performance is for lower crossover rates (from 0.2 to 0.5), while the row "beats all" clearly show that the percentage of space on which we can be 95% confident that each algorithm beats all of the other algorithm compared is always very close to zero, which suggest that all the crossover value could be chosen and DEMPO would anyway deliver comparable results. In fact, DE is known for requiring little parameter tuning.

Table 2.a: Statistical Comparison (MOSTATS) of NSGA II when the probability of crossover varies from 0.3 to 0.9 (stepsize=0.1) over 30 runs for each parameter setting. Entry *i* in the row "unbeaten" gives the percentage of the space on which the performance of the *i*th algorithm is unbeaten by any of the other algorithms compared (i.e.: the percentage of the fitness space for which we cannot be 95% confident, based on a non-parametric test - The Mann-Whitney Rank test, that any other algorithm beat it). Entry *i* in the row "beats all" gives the percentage of the space on which we can be 95% confident that the *i*th algorithm beats all of the other algorithms compared.

NSGA II - mv	crossover probability (pc)						
	0.3	0.4	0.5	0.6	0.7	0.8	0.9
unbeaten	40.1	30.3	86.3	38.4	89.4	43.3	40.9
beats all	0	0	0	0	0	0	0

[3]The code from A.Seshadri can be downloaded from MATLAB File Exchange.

Table 2.b: Statistical Comparison (MOSTATS) of DEMPO when the crossover rate varies from 0.2 to 0.9 (stepsize=0.1) f over 30 runs for each parameter setting. Entry *i* in the row "unbeaten" gives the percentage of the space on which the performance of the *i*th algorithm is unbeaten by any of the other algorithms compared (i.e.: the percentage of the fitness space for which we cannot be 95% confident, based on a non-parametric test - The Mann-Whitney Rank test, that any other algorithm beat it). Entry *i* in the row "beats all" gives the percentage of the space on which we can be 95% confident that the *i*th algorithm beats all of the other algorithms compared.

DEMPO- mv	crossover rate (cr)							
	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
unbeaten	85.8	89.8	70.5	80.8	61.7	60.5	53.9	36.4
beats all	1.5	0	0	0.1	0	0	0	0

4.5 Runtime performance

In terms of CPU time, the DEMPO algorithm runs a lot faster than the NSGA-II for equivalent number of fitness evaluations. For instance, for mean-variance portfolio optimization with 219 assets without real world constraints, with population size equal to 100, number of iterations equal to 500 and number of points to be determined on the frontier equal to 100, QP needed ca 360 seconds DEMPO needed ca 360 seconds compared to ca 960 seconds for the NSGA-II. However, a more efficient implementation of the NSGA-II might compensate for some of the performance difference.

4.6 Results

Appendix A shows a comparison of NSGA II and DEMPO on six benchmark functions. The statistical comparison of the two algorithms by MOSTATS (i.e.: the Mann-Whitney Rank test - see Table A.2) shows that DEMPO outperforms NSGA II in the four most complex benchmark problems, while NSGA II turns out to be better only for the simplest benchmark function. The area and gamma criteria (see Table A.3) confirm the better performance of DEMPO for the highly dimensional problems, while NSGA II turns out to be better in four out of five cases with respect to the delta measure, suggesting that DEMPO could be further improved with respect to the extent of spread achieved among the obtained solution.

Things look very different for the simple MV-Mean Variance portfolio optimization problem. Here, as figure 4 shows, DEMPO obtained excellent results that closely resemble the

QP solutions, whereas the NSGA-II fails to obtain reasonable results. This is not just a matter of available runtime. Even in very long runs with 10.000 iterations, the NSGA-II could not obtain reasonable results in contrast to DEMPO, which can obtain a quality comparable to QP.

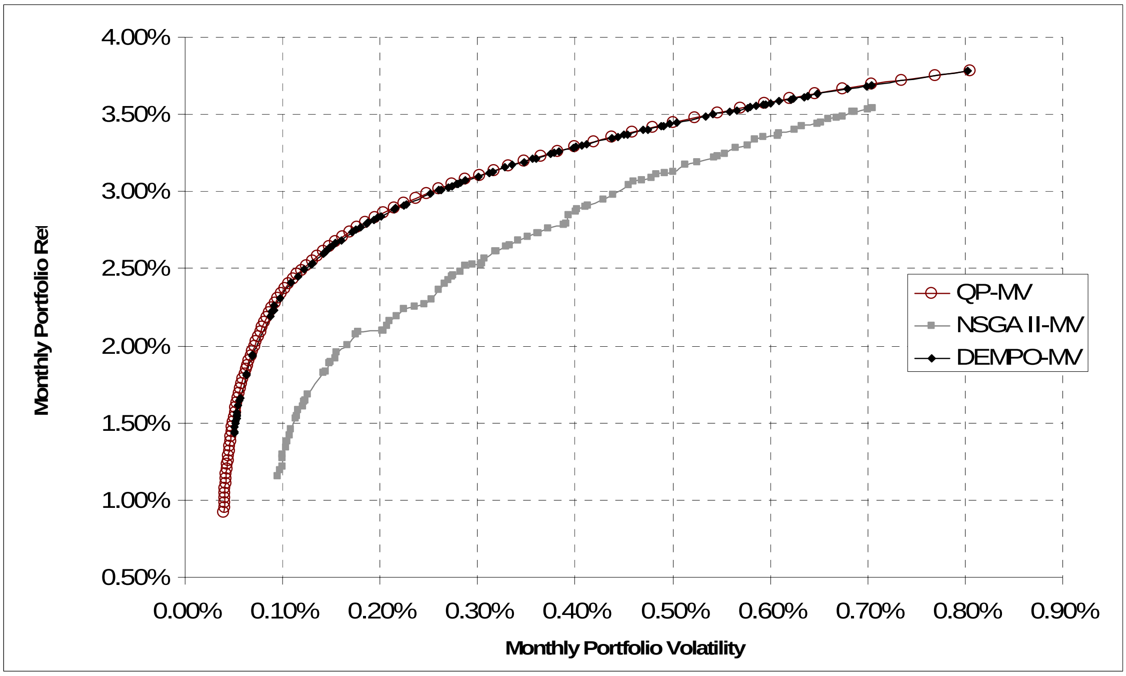

Figure 4: Mean-Variance Efficient frontier computed by QP, DEMPO and NSGAII.

Table 3 shows a statistical comparison by MOSTATS of DEMPO with different crossover rates (*cr*=0.2, 0.3, 0.4, 0.9) and NSGA II with different probability of crossover (*pc*=0.4, 0.5, 0.6, 0.7). The other parameter settings are reported in Table 1. We consider 30 runs for each parameter setting. DEMPO always outperform NSGA II, whichever crossover rate we consider. Furthermore, if we would compare 30 runs of DEMPO with a given crossover rate with all the results we have from NSGA II, we always have 100 and 100 in correspondence of DEMPO for both rows of MOSTATS output (unbeaten, beats all), which strongly supports the usage of DEMPO with respect to NSGA II.

Table 3: Statistical Comparison (MOSTATS) of DEMPO and NSGA II over 30 runs for each parameter setting. Entry *i* in the row "unbeaten" gives the percentage of the space on which the performance of the *i*-th algorithm is unbeaten by any of the other algorithms compared (i.e.: the percentage of the fitness space for which we cannot be 95% confident, based on a non-parametric test - The Mann-Whitney Rank test, that any other algorithm beat it). Entry *i* in the row "beats all" gives the percentage of the space on which we can be 95% confident that the *i*-th algorithm beats all of the other algorithms compared.

	DEMPO-mv crossover rate (*cr*)				**NSGA II - mv** probability of crossover (*pc*)			
	0.2	**0.3**	**0.4**	**0.9**	**0.4**	**0.5**	**0.6**	**0.7**
unbeaten	86	89.8	76.3	53.2	0	0	0	0
beats all	2.2	1.2	0.6	6.2	0	0	0	0

Tables 4 and 5 show the minimum, mean, maximum values, standard deviations and 90th percentiles of the ratio of covered area of DEMPO or NSGAII algorithm and the QP-MV frontier, the gamma and delta respectively over 210 runs (DEMPO: 30 runs for *cr*={0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8}, NSGA II: 30 runs for *pc*={0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}) and over 30 runs (DEMPO with *cr*=0.2, NSGA II with *pc*=0.7). The empirical results show that DEMPO clearly outperform NSGA II with respect to the ratio of the covered area and gamma, while the values for delta suggest that NSGA II can identify a frontier with more evenly scattered and spread out points than DEMPO (see also Figure 4). However, we notice that the standard deviation are always smaller for DEMPO algorithm, suggesting that this algorithm tends to converge more robustly towards the optimal front. Such result is confirmed also by the minimum, maximum and 90th percentiles values, which are closer to the mean values and less scattered for DEMPO rather than NSGA II. Furthermore, we notice that the maximum area/gamma for NSGA II is far below/above than the mean area/gamma for DEMPO. When we compare the algorithm for a given parameter setting (which we choose such that to consider the best/worst results achieved for NSGA II/DEMPO with respect to the area criterion), as reported in Table 5, we notice that it seems that also for NSGA II the increasing of the ratio of covered area leads to identify solutions that are less scattered along the Pareto front, which is confirmed by the larger mean delta value, that even turns out to be larger than the one of DEMPO.

Table 4: Minimum, mean, maximum values, standard deviations and 90th percentiles of the ratio of covered area of DEMPO or NSGAII algorithm and the QP-MV frontier, the gamma and delta over 210 runs (DEMPO: 30 runs for *cr*={0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8}, NSGA II: 30 runs for *pc*={0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9} for Mean-Variance (Problem: 1*a*-1*c*). Best values are reported in bold.

		min	mean	max	std	90th percentile
Ratio of Covered Area	DEMPO	75.6480	**97.5224**	100.0000	3.4091	99.6356
	NSGA II	4.4722	46.0764	78.8925	22.4183	72.9006
Gamma	DEMPO	0.00009	**0.00011**	0.00025	0.00002	0.00013
	NSGA II	0.00041	0.00156	0.00582	0.00077	0.00249
Delta	DEMPO	0.66279	0.77962	1.05432	0.05300	0.83056
	NSGA II	0.46885	**0.67903**	0.92748	0.11417	0.84404

Table 5: Minimum, mean, maximum values, standard deviations and 90th percentiles of the ratio of covered area of DEMPO or NSGAII algorithm and the QP-MV frontier, the gamma and delta over 30 runs (DEMPO with *cr*=0.4, NSGA II with *pc*=0.7 for Mean-Variance (Problem: 1*a*-1*c*). Best values are reported in bold.

		min	mean	max	std	90th percentile
Ratio of Covered Area	DEMPO	75.6480	**95.2284**	99.9046	7.3021	99.6948
	NSGA II	26.7118	56.1974	82.0917	20.8756	78.0827
Gamma	DEMPO	0.00009	**0.00011**	0.00016	0.00002	0.00014
	NSGA II	0.00091	0.00147	0.00250	0.00042	0.00208
Delta	DEMPO	0.73282	**0.79818**	0.84165	0.03217	0.83309
	NSGA II	0.73503	0.81274	1.05432	0.09006	0.94397

Summing up, DEMPO has always smaller standard deviation than NSGA II, converging in a more stable and robust way towards the optimal frontier. However, NSGA II seems, even it is not always the case, to identify frontiers with more evenly scattered and spread out points than DEMPO. Hence, we are currently working on improving DEMPO capability of identifying points more scattered along the frontier. One option is to use an ad hoc initialization that calculates candidate solution seeds more scattered using the QP-MV algorithm.

As Figure 4 shows, DEMPO can obtain results with the same fidelity and about the same performance of QP for realistic portfolio optimization problems while allowing using any kind of objectives and constraints without any requirements for linearity or convexity. Interestingly the runtime for multiobjective QP solutions (ca 360 seconds for 100 solutions) is comparable to DEMPO (ca 360 seconds for 100 solutions)

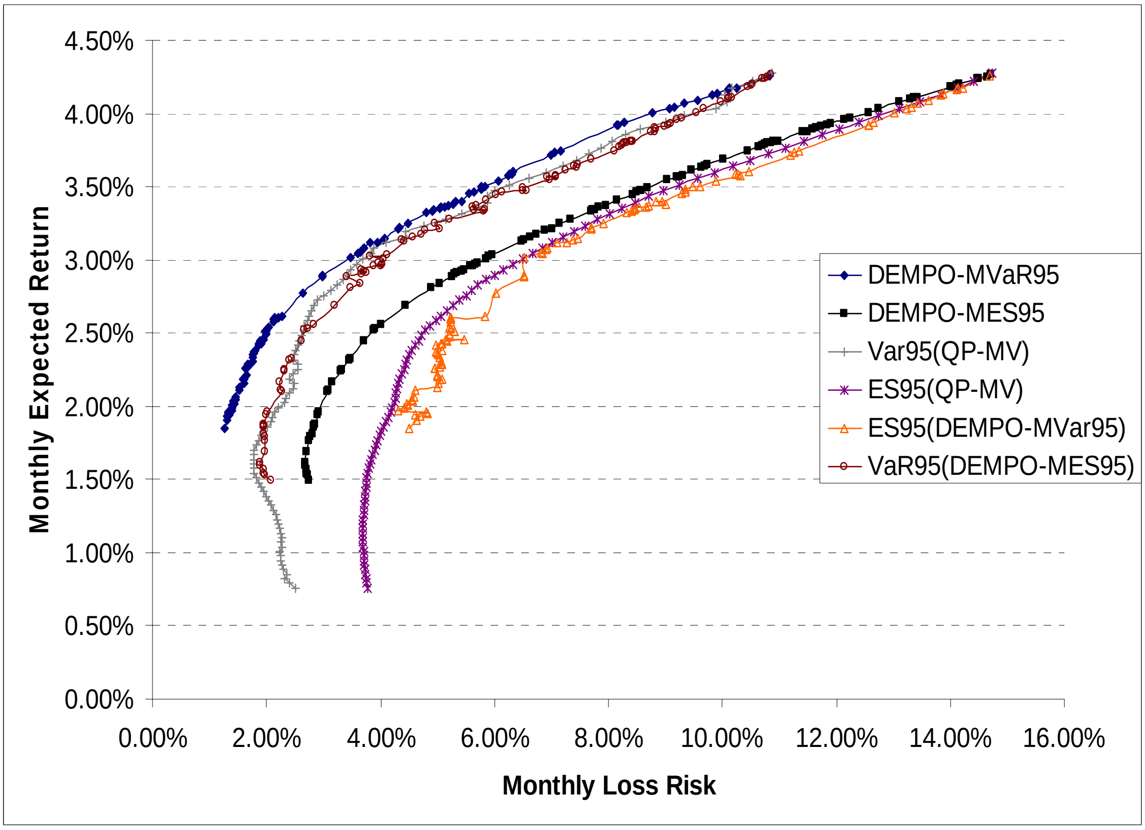

Figure 5: Comparison of Efficient frontiers for mean-Var_{95} and mean-ES_{95} computed by DEMPO and QP-MV. DEMPO-MVaR95 is the efficient frontier in the space (-VaR95, $\overline{L}$) computed by DEMPO when solving problem (2a, 2b, 2c), DEMPO-MES95 is the efficient frontier in the space (-ES95, $\overline{L}$) computed by DEMPO when solving problem 3a-3c, Var95(QP-MV) is the frontier in the space (-Var95, $\overline{L}$) computer by QP when solving problem (1a, 1b,1c) ES95(QP-MV) is the frontier in the space (-ES95, $\overline{L}$) computer by QP when solving problem (1a, 1b, 1c), ES95(DEMPO-MVaR95) is the frontier in the space (-ES95, $\overline{L}$) computed by DEMPO when solving problem (2a, 2b, 2c), VaR95(DEMPO-MES95) is the frontier in the space (-VaR95, $\overline{L}$) computed by DEMPO when solving problem (3a, 3b, 3c).

Figure 5 shows in the space of monthly loss risk (-VaR_{95} or –ES_{95}) and monthly expected return ($\overline{L}$) the frontiers identified by DEMPO when considering value at risk (DEMPO-MVar95, (problem specification: equations 2a, 2b, 2c) and expected shortfall (DEMPO-MES95) as measures of risk (problem specification: equations 3a, 3b, 3c). Such frontiers lie clearly above the frontiers that can be plotted in the same space when considering the MV-QP asset allocations (Var95-QP-MV and ES95-QP-MV). Defining such different risk measures does not allow tackling the optimization problem by using standard quadratic programming techniques. Moreover, we show that the frontiers (ES95(DEMPO-MVar95)) and (VaR95(DEMPO-

MES95)) determined respectively in correspondence of the optimal asset allocation of DEMPO-MVar95 and DEMPO-MES95 and plotted respectively in the space (-ES95, $\overline{L}$) and (-VaR95, $\overline{L}$) are clearly dominated.

Finally, DEMPO algorithm allows also determining the frontier when the real world constraints 4-7 are imposed. Figure 6 shows a comparison of the frontiers identified by DEMPO for equations (2a, 2b, 2c) and (3a, 3b, 3c) and the MV-QP solution when each asset weight cannot be bigger than 0.3.

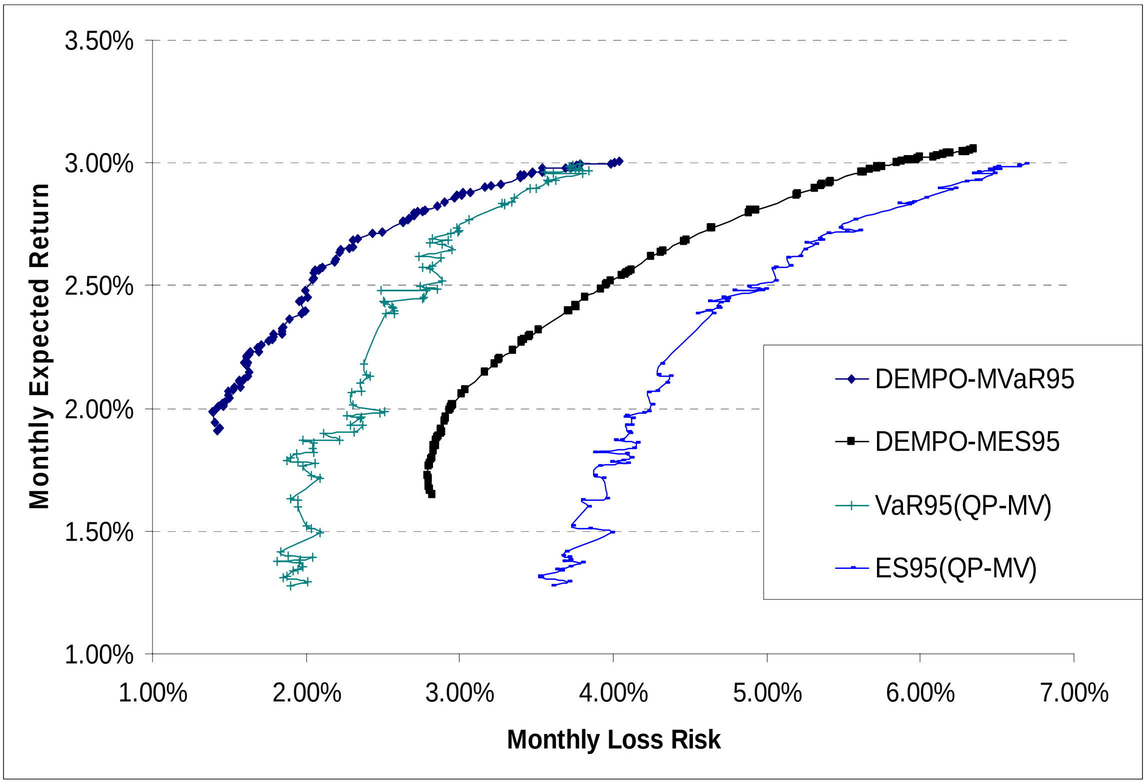

Figure 6: Comparison of Efficient frontiers for mean-Var$_{95}$ and mean-ES$_{95}$ computed by DEMPO and QP-MV with an upper bound of 0.3 for each asset weight. DEMPO-MVaR95 is the efficient frontier in the space (-VaR95, $\overline{L}$) computed by DEMPO when solving problem (2a, 2b, 2c), DEMPO-MES95 is the efficient frontier in the space (-ES95, $\overline{L}$) computed by DEMPO when solving problem (3a, 3b, 3c), Var95(QP-MV) is the frontier in the space (-Var95, $\overline{L}$) computer by QP when solving problem (1a, 1b, 1c), ES95(QP-MV) is the frontier in the space (-ES95, $\overline{L}$) computer by QP when solving problem (1a, 1b, 1c).

5 Discussion and Conclusions

In this paper, we have introduced a new multiobjective algorithm for portfolio optimization: DEMPO - Differential Evolution for Multiobjective Portfolio Optimization. Perhaps the most important result is that the new algorithm has the great advantage of full generality, i.e.: the ability to tackle a problem as it is without requiring rigid assumptions about convexity and linearity, while obtaining highly accurate results in very reasonable runtime. The algorithm allows considering different objective functions, such as value at risk and expected shortfall, and typical real world constraints that managers have often to satisfy. The comparison with quadratic programming for the standard mean-variance portfolio optimization problem shows that DEMPO can reach comparable results with the same runtime for high dimensional problem. The main drawback of DEMPO with respect to QP seems to be the inability of identifying solutions over the frontier as spread out as the QP solutions. We are currently working on this problem and preliminary results suggest that by using an ad-hoc initialization scheme this drawback does not exist any longer. Moreover, to our knowledge there has not been a comparison with a QP approach to portfolio optimization yet that has demonstrated that the quality of results obtained with a DE based approach and the required runtime is comparable for high dimensional problems.

We have also shown that one of the most popular multiobjective search heuristics the NSGA-II cannot nearly obtain the same quality of results. Why is this the case? One of the main reasons is the great performance of differential evolution for continuous numerical problems compared to genetic algorithms. This insight is not new and has been reported in a variety of recent studies. NSGA II clearly underperforms DEMPO and QP even when we consider the simple mean-variance portfolio optimization problem.

Moreover, proper constraint handling is essential in real-world optimization. The original constraint handling of the NSGA-II is indeed very elegant for problems with inequality constraints (as long as these are not too tightly defined), but clearly insufficient for equality constraints. For the latter it makes a lot more sense to apply other techniques, such as "repair" operators as the one we use in this investigation.

References

Abbass, H.A., and Sarker, R., Pareto Differential Evolution Algorithm, International Journal on Artificial Intelligence Tools, 11, 4, 531-552, 2002.

Abbass, H.A., Sarker, R., and Newton, C., PDE: A Pareto-frontier Differential Evolution Approach for Multi-objective Optimization Problems, Congress on Evolutionary Computation (CEC'2001), Seoul, Korea, 971-978, 2001.

Artzner, P., Delbaen, F., Eber J.M., and Heath, D., Thinking coherently, Risk Magazine, 10, 68-71, 1997.

Babu, B.V., Chakole, P.G., and Mubeen, J.H.S, Multiobjective differential evolution (MODE) for optimization of adiabatic styrene reactor, Chemical Engineering Science, 60, 17, 4822-4837, 2005.

Bertocchi, M., Giacometti, R., The Parallel Genetic Algorithm as Global Optimizer. Mathematical Optimization: Theory, Methods and Applications, K.H. Elster and C. Sutti eds, Verona, 29-43, 1993.

Bertocchi, M., Giacometti, R.,Global Continuous Optimization: a Parallel Genetic Approach, Neural Network World, Int. J. on Neural and Mass-Parallel Computing and Information Systems, M. Novak ed, 3, 6, 665-679, 1993.

Chankong, V., and Haimes, Y., Multiobjective Decision Making Theory and Methodology, New York : North-Holland, 1983.

Chang, T.-J., Meade, N., Beasley, J.E., Sharaiha, Y.M., Heuristics for Cardinality Constrained Portfolio Optimization, Computers and Operations Research, 27, 1271-1302, 2000.

Coello Coello, C.A., A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques, Knowledge and Information Systems. An International Journal, 1, 3, 269-308, 1999.

Coello Coello, C.A., David, A.V.V., and Gary, B.L., Evolutionary Algorithms for Solving Multi-Objective Problems, Kluwer Academic/Plenum, New York, 2002.

Cont, R., Empirical Properties of Asset Returns: Stylised Facts and Statistical Issues, Quantitative Finance, 1, 223-36, 2001.

Deb, K., Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, New York, 2001.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., A Fast and Elitist Multiobjective Genetic Algorithm, IEEE Transactions on Evolutionary Computation, 6, 2, 182-197, 2002.

Doganoglu, T. Hartz, C., Mittnik, S., Portfolio Optimization when Risk Factors are conditionally varying and heavy tails, Center for Financial Studies, Frankfurt, 24, 2006.

Dueck, G., and Winker, P., New concepts and algorithms for portfolio choice, Applied Stochastic Models and Data Analysis, 8, 159-178, 1992.

Fonseca, C.M. and Fleming, P.J., On the performance assessment and comparison of stochastic multiobjective optimizers. In: Parallel Problem Solving from Nature IV, Voigt, H.-M., Ebeling, W., Rechenberg, I., and Schwefel H.-P., Springer-Verlag, Berlin, Germany, 584-593, 1996.

Gaivoronski, A.A., Pflug, G., Value at Risk in Portfolio Optimization: Properties and Computational Approach. Journal of Risk, 7, 2, 1-31, Winter 2004-2005.

Gilli, M., Këllezi, E. and Hysi H., A Data-Driven Optimization Heuristic for Downside Risk Minimization, The Journal of Risk 8, 3, 1-18, 2006.

Haimes, Y., Lasdon, L., and Wismer, D., On a bicriterion formulation of the problems of integrated system identication and system optimization, IEEE Transactions on Systems, Man, and Cybernetics, 1, 296-297, 1971.

Knowles, J., and Corne, D., The Pareto archived evolution strategy: a new baseline algorithm for multiobjective optimization. In: 1999 Congress on Evolutionary Computation, Washington D.C., IEEE Service Centre, 98-105, 1999.

Knowles, J., and Corne, D., Approximating the nondominated front using the Pareto archived evolution strategy, Evolutionary Computation, 8, 2, 149-172, 2000.

Lamantia, F., Ortobelli, S., and Rachev, S., VaR, CVaR and Time Rules with Elliptical and Asymmetric Stable Distributed Returns. Investment Management and Financial Innovations 4, 19-39, 2006.

Lamantia, F., Ortobelli, S., and Rachev, S., An Empirical Comparison among VaR Models and Time Rules with Elliptical and Stable Distributed Returns. Investment Management and Financial Innovations 3, 8-29, 2006.

Lampinen, J., A Bibliography of Differential Evolution Algorithm, http://www2.lut.fi/~jlampine/debiblio.htm

Markowitz, H.M., Portfolio Selection, Journal of Finance, 7, 77-91, 1952.

Michalewicz, Z., and Fogel, D.B., How to solve it: modern heuristics, Springer, 2004.

Meyer, R.R., Integer and mixed-integer programming models: general properties, Journal of Optimization Theory and Applications, 16, 191-206, 1975.

Ortobelli, S., Rachev, S., Huber, I., Biglova, A., Optimal portfolio selection and Risk management: A comparison between the Stable Paretian Approach and the Gaussian one.

Handbook of Computational and Numerical Methods in Finance, (Rachev S. edt.) Birkhauser, Boston 197-252, 2004.

Price, K., Storn, R., and Lampinen, J., Differential Evolution - A Practical Approach to Global Optimization, Springer, 2004.

Quintero, L.V.S., and Coello Coello, C.A., An Algorithm Based on Differential Evolution for Multi-Objective Problems, International Journal of Computational Intelligence Research, 1, 2, 151-169, 2005.

Sarker, R., and Abbass, H.A., Differential evolution for solving multiobjective optimization problems, Asia-Pacific Journal of Operational Research, 21, 2, 225-240, 2004.

Sarker, R. and Coello, C., Assessment Methodologies for Multiobjective Evolutionary Algorithms. In: Evolutionary Optimization, Sarker, R., Mohammadian, M., and Yao, X., (edited), Kluwer Academic Publishers, Boston, 177-195, 2002.

Srinivas, N., and Deb, K., Multiobjective optimization using nondominated sorting in genetic algorithms, Evolutionary Computation, 2, 3, 221–248, 1994.

Sheshadri, A., Matlab File Exchange, Optimization > NSGA - II: A multi-objective optimization algorithm, http://www.mathworks.com/matlabcentral/fileexchange .

Storn, R., and Price, K., Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces, Journal of Global Optimization, 11, 341-359, 1997.

Streichert, F., Ulmer, H., and Zell., A., Evolutionary algorithms and the cardinality constrained portfolio selection problem. In Ahr, D., Fahrion, R., Oswald, M., and Reinelt, G., editors, Operations Research Proceedings 2003, Selected Papers of the International Conference on Operations Research (OR 2003), Springer, 2003.

Vedarajan, G., Chan, L.C., and Goldberg, D.E., Investment Portfolio Optimization using Genetic Algorithm. In: Koza, J.R., Late Breaking Papers at the Genetic Programming 1997 Conference, Stanford University California: Stanford University Bookstore, 255-263, 1997.

APPENDIX A

Appendix A shows the results for six multiobjective benchmark problems regarding MOSTATS, the distance (gamma), the dispersion (delta) and the proportion of area covered with respect to a known Pareto front with 100 points. Table A.1 below reports the description of the six benchmark problems we have considered.

Table A.1: Multiobjective benchmark problems (Deb et al. 2002)

Problem	*n*	Variable bounds	Objective functions	Optimal solutions	Comments
SCH	1	[-10^3,10^3]	$f_1(x) = x^2$ $f_2(x) = (x-2)^2$	$x \in [0,2]$	convex
FON	3	[-4, 4]	$f_1(x) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right)$ $f_2(x) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i + \frac{1}{\sqrt{3}}\right)^2\right)$	$x_1=x_2=x_3$	nonconvex
KUR	3	[-5, 5]	$f_1(x) = \sum_{i=1}^{n-1}(-10\exp(-0.2\sqrt{x_i^2 + x_{i+1}^2}$ $f_2(x) = \sum_{i=1}^{n}\left(\lvert x_i\rvert^{0.8} + 5\sin x_i^3\right)$	refer Deb 2001	nonconvex
ZTD1	30	[0,1]	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{x_1/g(x)}\right]$ $g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$	$x_1 \in [0,1]$ $x_i = 0, i = 2,..,n$	convex
ZTD2	30	[0,1]	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - (x_1/g(x))^2\right]$ $g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$	$x_1 \in [0,1]$ $x_i = 0, i = 2,..,n$	nonconvex

Problem SCH has 1 dimension, FON and KUR have 3 dimensions, whereas problems ZTD1 and ZRD2 have 30 dimensions. SCH and ZTD1 are convex functions, while FON, KUR and ZTD2 are non convex.

Table A.2 reports the results from the MOSTATS comparison, while Table A.3 reports the mean values and the standard deviation over 30 runs for delta, gamma and area (see section 4.2 for a full description). The reported results are obtained with *cr*=0.4 for DEMPO and with *pc*=0.7 for NSGA II, while the other parameter values are reported in Table 1. Moreover, we run the experiments with different parameter settings for *pc* and *cr*, as reported in Table 1, obtaining comparable results. Results are available upon request from the authors.

The smaller gamma and delta are, the closer to 100 the area is (we consider the ratio of the covered area of DEMPO or NSGA II and the area of the known Pareto optimal solution) the better the performance of the algorithms. Best mean values in 30 runs are reported in bold. The statistical comparison by MOSTATS suggests that DEMPO should clearly be preferred to NSGA II for the more complex problems. For the very simple benchmark problems with low dimensionality and without constraints, the results for the DEMPO algorithm are not better than the NSGA-II with respect to the gamma and delta criteria. However, there are clear differences for higher dimensional problems as KUR, ZTD1 and ZTD2: in such problems DEMPO is better than NSGA II with respect to the area and gamma criteria, but not always with respect to delta. NSGA II seems to have better spread out solution than DEMPO, but DEMPO tend to converge to better front.

Table A.2: Statistical Comparison (MOSTATS) of DEMPO and NSGA II over 30 runs for each benchmark problem. The rows "unbeaten" report the percentage of the space on which the performance of DEMPO (column 3) or of NSGA II (column 4) is unbeaten respectively by NSGA II or DEMPO (i.e.: the percentage of the fitness space for which we cannot be 95% confident, based on a non-parametric test - The Mann-Whitney Rank test, that any other algorithm beat it). The rows "beats" report the percentage of the space on which we can be 95% confident that the DEMPO (column 3) and NSGA II (column 4) beats respectively NSGA II and DEMPO.

		DEMPO	**NSGA2**
SCH	unbeaten	39.2	100
	beats all	0	60.8
FON	unbeaten	96.1	91.9
	beats all	8.1	3.9
KUR	unbeaten	94.9	14
	beats all	86	5.1
ZDT1	unbeaten	100	2.7
	beats all	97.3	0
ZDT2	unbeaten	100	1.5
	beats all	98.5	0

Table A.3: Mean values and standard deviations (in brackets) over 30 runs for DEMPO and NSGA II for 6 multiobjective benchmark problems. Best results are reported in bold.

		SCH		FON		KUR		ZDT1		ZDT2	
	Algorithm	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Area	DEMPO	**98.11**	3.14	98.43	1.43	**98.78**	1.55	**100.00**	0.03	**100.00**	0.02
	NSGA2	97.24	8.07	**98.46**	1.02	71.38	18.35	82.19	3.27	75.39	15.41
Gamma	DEMPO	0.0166	0.0006	**0.0044**	0.0002	**0.0235**	0.0014	**0.0050**	0.0007	**0.0040**	0.0002
	NSGA2	**0.0162**	0.0012	0.0048	0.0002	0.5546	0.4016	0.0854	0.0160	0.1422	0.0228
Delta	DEMPO	0.8178	0.1139	0.8787	0.0160	0.9102	0.0148	0.6806	0.0507	**0.6689**	0.0377
	NSGA2	**0.4401**	0.0893	**0.7942**	0.0120	**0.8410**	0.0261	**0.4728**	0.0281	0.8809	0.0984