

A Hybrid Solver for Constrained Portfolio Selection Problems

— preliminary report —

Luca Di Gaspero¹, Giacomo di Tollo², Andrea Roli³, Andrea Schaerf¹

1. DIEGM, Università di Udine, via delle Scienze 208,

I-33100, Udine, Italy, email: {`l.digaspero`, `schaerf`}@uniud.it

2. Dipartimento di Scienze, Università “G.D’Annunzio”, viale Pindaro 87,

I-65127, Pescara, Italy, email: `ditollo@unich.it`

3. DEIS, Università degli Studi di Bologna, Campus of Cesena, via Venezia 52

I-47023 Cesena, Italy, email: `andrea.roli@unibo.it`

Abstract

Portfolio selection is a relevant problem arising in finance and economics. While its basic formulation can be efficiently solved through linear programming, its more practical and realistic variants, that include various kinds of constraints and objectives, have to be tackled by approximate algorithms. In this work, we present a hybrid technique that combines a local search, as *master* solver, with a quadratic programming procedure, as *slave* solver. Preliminary results show that the approach is very promising and achieves results comparable or superior with the state of the art solvers.

1 Introduction

The *portfolio selection* problem consists in selecting a portfolio of *assets* that provides the investor a given expected return and minimizes the *risk*. One of the main contributions in this problem is the seminal work by Markowitz [1959], who introduced the so-called *mean-variance* model, which takes the variance of the portfolio as the measure of risk. According to Markowitz, the portfolio selection problem can be formulated as an optimization problem over real-valued variables with a quadratic objective function and linear constraints.

In this paper we consider the basic objective function introduced by Markowitz, and we take into account three additional constraints: the *cardinality* constraint which limits the number of assets, the *quantity* constraint which fixes minimal and maximal shares of each individual one in the portfolio, and the *preassignments* that force some specific assets to be included in the portfolio.

We devise a hybrid solution based on a local search metaheuristic that at each step calls a quadratic programming (QP) solver. The QP implements the Goldfarb-Idnani dual-active set algorithm [Goldfarb and Idnani, 1983] for strictly convex quadratic programs.

The use of local search techniques for the constrained portfolio selection problem has been proposed by several authors, including for example Rolland [1997]; Chang *et al.* [1999] and Schaerf [2002].

The cited works however use local search as overall solver, exploring a search space composed by both continuous and discrete variables. Conversely, our hybrid solver focuses on the discrete variables leaving the determination of the continuous ones to the QP solver.

In addition, we consider here a more general problem w.r.t. the cited three papers, including also the possibility to specify a minimum number of assets (and not only the maximum) and to declare some assets as preassigned.

The use of a hybrid solver has been (independently) proposed also by Moral-Escudero *et al.* [2006], who, however, make use of genetic algorithms instead of local search for the determination of the discrete variables.

In this paper, we describe our hybrid solver and we report some preliminary results. The hybrid solver compares favourably with the results in the literature. Nevertheless, further experiments are necessary to perform other comparisons, to investigate on the structure of the instances, and on the settings of the solver.

2 Problem definition

A set of n assets, $A = \{a_1, \dots, a_n\}$ is given. Each asset a_i has associated a real-valued *expected return* (per period) r_i , and each pair of assets $\langle a_i, a_j \rangle$ has a real-valued *covariance* σ_{ij} . The matrix $\sigma_{n \times n}$ is symmetric and the diagonal elements σ_{ii} represent the *variance* of assets a_i . A real value R represents the desired expected return.

A portfolio is a set of real values $X = \{x_1, \dots, x_n\}$ such that each x_i represents the fraction invested in the asset a_i . The value $\sum_{i=1}^n \sum_{j=1}^n \sigma_{ij} x_i x_j$ represents the variance of the portfolio, and it is considered as the measure of the risk associated with the portfolio. Consequently, the problem is to minimize the overall variance, still ensuring the expected return R . The formulation of the basic (unconstrained) problem is thus the following.

$$\begin{aligned} \min \quad & \sum_{i=1}^n \sum_{j=1}^n \sigma_{ij} x_i x_j \\ \text{s.t.} \quad & \sum_{i=1}^n r_i x_i \geq R \end{aligned} \tag{1}$$

$$\sum_{i=1}^n x_i = 1 \tag{2}$$

$$0 \leq x_i \leq 1 \quad (i = 1, \dots, n) \tag{3}$$

This is a quadratic programming problem, and nowadays it can be solved optimally using

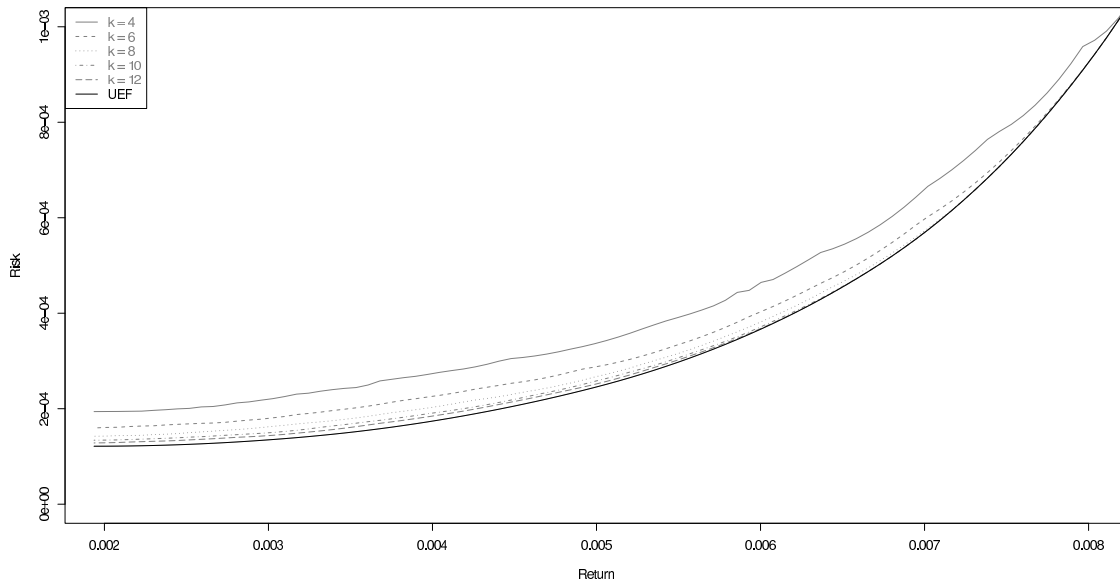


Figure 1: A comparison of the unconstrained efficiency frontier with the constrained efficiency frontier found by our best algorithm for different constraint levels

available tools despite the NP-completeness of the underlying decision problem [Mansini and Speranza, 1999].

By solving the problem as a function of R , we obtain the so-called *unconstrained efficiency frontier* (UEF), that gives for each expected return the minimum associated risk. The UEF for one of the benchmark instances is provided in Figure 1 (the lowest black solid line).

In our work we consider additional constraint types, and in order to model them correctly, we need to add to the formulation a set of binary decision variables Z such that $z_i = 1$ if and only if asset i is in the solution (i.e. $x_i > 0$).

Cardinality constraint: The number of assets that compose the portfolio is limited: we give two values k_{min} and k_{max} (with $1 \leq k_{min} \leq k_{max} \leq n$) such that:

$$k_{min} \leq \sum_{i=1}^n z_i \leq k_{max} \quad (4)$$

Quantity constraints: The quantity of each asset i that is included in the portfolio is limited within a given interval: we give a minimum ϵ_i and a maximum δ_i for each asset i , such that:

$$z_i = 0 \quad \vee \quad \epsilon_i \leq x_i \leq \delta_i \quad (i = 1, \dots, n) \quad (5)$$

Preassignments: Certain assets are preassigned in the portfolio: we give a binary vector P (if size n) such that $p_i = 1$ if and only if a_i is preassigned in the portfolio.

$$z_i \geq p_i \quad (i = 1, \dots, n) \quad (6)$$

Preassignment constraints have been discussed by Chang *et al.* [2000], and are not considered in previous formulations. For an overview of the formulations presented in the literature we forward the interested reader to [di Tollo and Roli, 2006].

Notice that if the number of preassigned assets $\sum_{i=1}^n p_i$ is greater than k_{min} , then k_{min} is implicitly set to it. Notice also that preassignments and minimum cardinality are especially meaningful in presence of constraints on the minimum quantity, otherwise they can be satisfied by infinitesimal quantities.

We call CEF the analogous of the UEF for the constrained problem. In Figure 1 we plot the CEF found by the best of our algorithms on a typical instance for the values $\epsilon_i = 0.01$, $\delta_i = 1$ (for $i = 1, \dots, n$), $k_{min} = 1$, and k_{max} varying from 4 to 12.

For higher values of k_{max} the curve is almost indistinguishable from the UEF, indeed the total distance among the CEF and the UEF becomes smaller than 10^{-3} .

3 A hybrid local search solver for portfolio selection

In order to apply local search techniques to portfolio selection we need to define the search space, the neighborhood structures, the cost function, and the selection rule for the initial solution.

Search space. Local search works on the space of the set Z only. For computing the actual quantities X , we use the QP solver, with input assets only those such that $z_i = 1$.

Constraints (4) and (6) are implicitly enforced by the local search solver by exploring only states that satisfy them. Constraints (1), (2), (3), and (5) are passed to the QP solver that handles them directly.

If the QP solver is unable to produce a feasible solution for the set of assets composing the current state and the target return R , we relax Constraint (1), and we build the configuration of X (based on Z) that gives the best return without violating the other constraints. This can be easily done by a greedy algorithm that sorts the assets by the expected return and assigns the maximum quantity to each asset in turn, as long as the sum is smaller than 1.

Neighborhood. The neighborhood relation is based on insertion, deletion, or replacement of an asset. That is, a move consists in either flipping the value of one z variable, or swapping the value of two of them.

In order to enforce Constraints (4) no deletion move is considered feasible if the number of assets in the state is equal to k_{min} , and no insertion if it is equal to k_{max} . To enforce Constraints (6), the preassigned assets are never removed from the solution.

Cost function. For each examined state we run the QP solver in order to assign the values to the X variables, and computing the cost of the state. If the QP solver returns a solution that satisfies all constraints, its cost is the total risk (that is computed by QP). Conversely, if no solution reaching the target return R is feasible, the cost is determined by the degree of violation of Constraint (1).

No.	Origin	assets	UEF
1	Hong Kong	31	$1.55936 \cdot 10^{-3}$
2	Germany	85	$0.412213 \cdot 10^{-3}$
3	UK	89	$0.454259 \cdot 10^{-3}$
4	USA	98	$0.502038 \cdot 10^{-3}$
5	Japan	225	$0.458285 \cdot 10^{-3}$

Table 1: The benchmark instances

Initial solution construction. For the initial solution, we first draw at random the number of assets between k_{min} and k_{max} , then we insert the preassigned assets (if any), and complete the portfolio by random assets.

Metaheuristics. We implemented three local search techniques, namely first descent (FD), steepest descent (SD), and tabu search (TS). FD is implemented searching for the first improving move at each iteration, whereas SD searches for the best one in the whole neighborhood. Both techniques stop when no further improvement is possible.

TS uses a short-term tabu mechanism (without any form of long-term prohibition). The length of the tabu list varies dynamically in the interval $[l_{min}, l_{max}]$: each move that enters the list is assigned a random tabu tenure, ranging from l_{min} to l_{max} (the values l_{min} and l_{max} are fixed parameters). A move is tabu if either tries to insert an asset removed by a move in the tabu list, or it tries to remove an asset inserted by a move in the list. TS stops when it does not improve the current best solution for a given number t of iterations (called *idle iterations*).

4 Benchmarks and experimental setting

We experimented our techniques on five instances obtained from real stock markets. The instances are taken from ORlib, and are available at the URL <http://mscmga.ms.ai.ac.uk/~jeb/orlib/portfolio.html>. We solve each instance for 100 equally distributed values for the expected return R .

For the sake of comparability, we set the constraint parameters exactly as in previous work [Chang *et al.*, 1999; Schaerf, 2002; Moral-Escudero *et al.*, 2006]: $\epsilon_i = 0.01$ and $\delta_i = 1$ for $i = 1, \dots, n$, and $k_{max} = 10$ for all instances. The minimum cardinality is not considered in the cited work, and therefore we set it to $k_{min} = 1$ (no limitation).

Even though we did not perform a thorough parameter setting for TS, we observe on preliminary experiments that the TS metaheuristic was quite robust. Therefore we choose to set the TS parameters to reasonable values for the instances at hand, that is: $l_{min} = 3$, $l_{max} = 5$ and $t = 100$.

As in previous work, we measure the quality of our solutions in average percentage loss w.r.t. the UEF. Table 1 illustrates for all instances the original market, and the average variance of UEF.

Inst. No.	FD + QP		SD + QP		TS + QP		EA + QP [Moral-Escudero <i>et al.</i> , 2006]		TS [Schaerf, 2002]	
	% diff.	time (s)	% diff.	time (s)	% diff.	time (s)	% diff.	time (s)	% diff.	time (s)
1	0.003665	1.26	0.003212	4.3	0.003212	63.8	0.003212	415.1	0.004093	251
2	2.6610	5.335	2.5314	20.2	2.5315	336.7	2.5318	552.7	2.5362	531
3	2.0015	5.41	1.9215	23.6	1.9212	408.9	1.9215	886.3	1.9260	583
4	4.7726	7.585	4.6937	27.6	4.6937	467.9	4.6951	1163.7	4.6982	713
5	0.2418	15.675	0.2022	69.5	0.2020	1248.2	0.2020	1465.8	0.2026	1603

Table 2: Comparison of results

5 Preliminary experimental results

In order to compare our solver to the results in the literature we ran some preliminary experiments on the benchmark instances.

Experiments were performed on an Apple iMac computer equipped with an Intel Core 2 Duo (2.16 GHz) processor and running Mac OS X 10.4; the SD, FD and TS metaheuristics have been coded in C++ exploiting the framework EASYLOCAL++ [Di Gaspero and Schaerf, 2003]. The executables were obtained using the GNU C/C++ compiler (v. 4.0.1).

For every point of the frontier we ran 3 trials of our metaheuristics, therefore altogether we ran 300 trials (3 trials \times 100 points) for each instance. Except for the first point of the UEF, one out of the three trial started from the solution found in the previous solved point of the UEF (instead of a random initial solution). Moreover, because of the stochastic nature of the metaheuristics employed, we repeat 30 times this procedure on each instance, in order to obtain statistically sound results.

Table 2 shows the best results and the running times obtained by our hybrid metaheuristics in comparison with previous work.

All results are the best average percentage difference w.r.t. the UEF (rounded at the fourth decimal digit); the running times of our solvers are reported as the median of the running times (out of the 30 repetitions mentioned above), whereas for the Moral-Escudero *et al.* [2006] hybrid evolutionary algorithms the figures are the minimum running times needed for finding their best solution (taken verbatim from their paper) on a PC having comparable performances (a Pentium IV processor, 3.2GHz). Running times of Schaerf [2002] were obtained re-running Schaerf’s software on our machine.

Table 2 clearly shows that we obtain results superior to Schaerf [2002] both in terms of risk, although minimal, and running times. This indicates that the hybrid solvers outperforms a monolithic local search one.

Regarding the comparison with Moral-Escudero *et al.* [2006], although the best values are not the most significant statistic, we can conclude that we obtain with our SD solver the same results of their best solver, but in a much shorter time (on a comparable machine). Anyway, before drawing definitive conclusions, we need other (more difficult, probably) instances to test the possible difference in performance of the algorithms.

Also the TS solver achieves very good results that on many instances are slightly better than those of SD. However this is obtained at the price of longer running times due to a

deeper exploration of the search space. This suggests that, for these instances, the strategy of restarting the algorithm from a random solution is more effective than a more systematic exploration around the local minima, but a deeper analysis of this behavior is needed. Concerning FD, in general this algorithm is inferior (in terms of solutions quality) than the other two.

As already pointed out by Schaerf [2002], even though Chang *et al.* [1999] solve the same problem instances, a fair comparison between their and our solutions is not possible because they compute the CEF differently. Specifically, they do not solve a different instance for each value of R , but (following [Perold, 1984]), they reformulate the problem without Constraint (1) and with the following objective function: $f(X) = \lambda f_1(X) + (1 - \lambda)f_2(X)$.

The problem is then solved for different values of λ . What they obtain is the solution for a set of values for R which are not homogeneously distributed.

6 Conclusions and future work

Experiments show that our solver is comparable with (or superior than) the state of the art for the less constrained problem formulation (no minimum, no preassigned). Experiments for the general problem are still to be done.

In the future, we plan to test this approach against more realistic formulations of the problem, such as the one with preassignments, that is useful for representing investor behaviour. Further experiments are also planned aiming at introducing the cardinality constraint expressed as an equality (i.e., $k_{min} = k_{max}$); indeed, it has been shown that when imposing this constraint, the problem seems to be more difficult and the CEF may collapse toward a single point [Armananzas and Lozano, 2005]. Comparisons with Armananzas and Lozano and Crama and Schyns [2003] will also be pursued.

We also aim at identifying difficult instances and verify whether more sophisticated local search metaheuristics, such as TS, could improve on the results of the simple SD strategy.

We are furthermore aimed in exploring and verify economic phenomena that arise from different settings of the parameters (diversification in small portfolios, investor preferences, ...) and comparing several risk measures on the same instances, to show analogies and differences in portfolios obtained optimizing different objective functions.

Acknowledgments

We thank Michael Schyns and Ruben Ruiz-Torrubiano for helpful clarification about their work.

References

R. Armananzas and J.A. Lozano. A multiobjective approach to the portfolio optimization problem. In *The 2005 IEEE Congress on Evolutionary Computation, CEC 2005*. Springer

Verlag, 2005.

- T.-J. Chang, N. Meade, J. E. Beasley, and Y. M. Sharaiha. Heuristics for cardinality constrained portfolio optimisation, 1999. Working paper, available from <http://mscmga.ms.ic.ac.uk/jeb/jeb.html>.
- T.-J. Chang, N. Meade, J. E. Beasley, and Y. M. Sharaiha. Heuristics for cardinality constrained portfolio optimisation. *Computers and Operations Research*, 27:1271–1302, 2000.
- Y. Crama and M. Schyns. Simulated annealing for complex portfolio selection problems. *European Journal of Operational Research*, 150:546–571, 2003.
- Luca Di Gaspero and Andrea Schaerf. EASYLOCAL++: An object-oriented framework for flexible design of local search algorithms. *Software—Practice and Experience*, 33(8):733–765, 2003.
- Giacomo di Tollo and Andrea Roli. Metaheuristics for the portfolio selection problem. Technical Report R-2006-005, Dipartimento di Scienze, Università “G. D’Annunzio” Chieti–Pescara, 2006.
- D. Goldfarb and A. Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27:1–33, 1983.
- Renata Mansini and Maria Grazia Speranza. Heuristic algorithms for the portfolio selection problem with minimum transaction lots. *European Journal of Operational Research*, 114:219–233, 1999.
- Harry Markowitz. *Portfolio selection, efficient diversification of investments*. John Wiley & Sons, 1959.
- R. Moral-Escudero, R. Ruiz-Torrubiano, and A. Suárez. Selection of optimal investment with cardinality constraints. In *IEEE World Congress on Computational Intelligence*, 2006.
- Andre F. Perold. Large-scale portfolio optimization. *Management Science*, 30(10):1143–1160, 1984.
- Erik Rolland. A tabu search method for constrained real-number search: applications to portfolio selection. Technical report, Dept. of accounting & management information systems. Ohio State University, Columbus. U.S.A., 1997.
- Andrea Schaerf. Local search techniques for constrained portfolio selection problems. *Computational Economics*, 20(3):177–190, 2002.