Optimization of Cardinality Constrained Portfolios with an Hybrid Local Search Algorithm

*Hans Kellerer** *Dietmar Maringer*†

* Department of Statistics and Operations Research, Karl Franzens University of Graz
Universitätsstrasse 15/E3, A–8010 Graz, Austria
Email: hans.kellerer@kfunigraz.ac.at

† Department of Banking and Finance, University of Vienna
Brünner Strasse 72, A–1210 Vienna, Austria
Email: dietmar.maringer@univie.ac.at

1 Introduction

One of the main advantages of portfolios over single assets is that risk can be diversified without necessarily reducing the expected return — provided the "right" assets are selected and they are assigned the "right" weights. Since in practice investors tend to restrict themselves to a rather small number of different assets, the decision which securities to include or not is a crucial one which turns out to be NP–hard.

In this paper we suggest a hybrid local search algorithm which combines principles of Simulated Annealing and Evolutionary Strategies and which proved to highly efficiently approach this problem.

2 Approach

2.1 Model and Algorithm

We assume a market with N different securities with expected returns of r_i and covariances σ_{ij}. If the investor wants to include at most $k \in [1, ..., N]$ different assets and asset i is included in the portfolio, a binary variable b_i is set equal to 1, otherwise it is 0. The asset's weight in the portfolio, w_i, shall be nonnegative and the weights of all assets included in the portfolio are to add up to 1. Since rational risk averse investors will try to maximize their portfolio's expected return, r_P, while minimizing its risk, σ_P, the optimization model can be stated as follows:[1]

$$\max \quad \lambda \cdot r_P - (1 - \lambda) \cdot \sigma_P$$

[1] A survey on "traditional" portfolio selection according to Markowitz as well as subsequent models, constrained and unconstrained, can be found in [3].

subject to

$$r_P = \sum_{i=1}^{N} w_i \cdot r_i$$

$$\sigma_p = \sqrt{\sum_{i=1}^{N} \sum_{j=1}^{N} w_i \cdot w_j \cdot \sigma_{ij}}$$

$$0 \leq \lambda \leq 1$$

$$w_i \geq 0 \quad \forall i$$

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

$$\sum_{i=1}^{N} b_i \leq k \quad \text{where} \quad b_i = \begin{cases} 1 & w_i > 0 \\ 0 & w_i = 0 \end{cases}$$

To approach this NP–hard problem, we suggest an iterative algorithm that uses local search strategies and enhances *Simulated Annealing* (*SA*) with principles known from evolutionary processes.[2] The central idea of SA is to start with some arbitrary solution and have it modified by allowing random changes whenever they result in an improvement. Additionally, to overcome local optima changes that come with an impairment are accepted, yet with decreasing probability. The decision of whether to allow a modification or not is based on the *Metropolis function* known from the solidification in metals. However, unlike the standard version of SA where just one crystal is considered this algorithm uses a whole "population" of crystals which are permanently ranked. According to evolutionary principles, the fittest of the population's members are favored whereas the worst are eliminated and replaced either by a clone of one of the best crystals or some new individual which is endowed with properties of the best crystals.

The algorithm starts with a random initialization of the crystals each representing a portfolio. This includes selecting k of the N available assets and assigning them random positive weights such that their sum adds up to 1, i.e., $\sum_i w_{ict} = 1$ where w_{ict} is the weight of asset i in crystal c's portfolio in iteration t. The subsequent iterations consist of three stages: *modification* of the portfolio structure; *evaluation* and ranking of the crystals; *replacement* of the poorest crystals in the population.

Modification. For each crystal c, d_k assets of this crystal's portfolio P_c are selected. For these assets changes in the respective weights are changed according to $w'_{ict} = \max\{w_{ict} + \widetilde{z}_{ict}; 0\}$ where $\widetilde{z}_{ict} \in [-U_t; U_t]$, the other assets' weights are left unchanged, i.e., $w'_{ict} = w_{ict}$. U_t indicates the bandwidth for changes in iteration t which is steadily narrowed by $U_t = U_{t-1} \cdot \gamma_U$. If w'_{ict} becomes zero, with a probability p_r the respective asset is replaced by a new asset j which is not yet included (i.e., $w_{jct} = 0$) and is given some random weight $w'_{jct} \in [0; 2 \cdot U_t]$. When selecting j, preferences based on the "Averaged Idol" as introduced later are used. With probability $1 - p_r$ asset i is kept in the portfolio with zero weight which means that there are less than k assets represented in the portfolio.

Having changed the weights and standardized them such that $\sum_i w'_{ict} = 1$, the fitness of the modified portfolio P' is calculated. According to the principles of SA, the modifications are accepted with probability p which comes from the Metropolis function $p = \min\left\{1; \exp\left(\frac{\Delta F}{T_t}\right)\right\}$. $\Delta F = F_P - F_{P'}$ is the change in the fitness function (here: objective function) and $T_t = T_{t-1} \cdot \gamma_T$ is the temperature in iteration t. Due to this definition, impairments are the more unlikely the larger the decrease in the fitness function and the lower the temperature, i.e., the more iterations

[2] An application of Simulated Annealing on portfolio optimization as well as a presentation of the underlying principles of this method can be found in [2].

have already been passed. For each crystal, this procedure of generating a modified portfolio and deciding whether to accept it or not is repeated for a fixed number of times.

Evaluation. The crystals are ranked according to their fitness. This evaluation and ranking procedure is crucial for the decision which solutions to reinforce and which to eliminate. Similar to the rank based system introduced in [1] we allow only the best π of all crystals to be role models for others (we shall call them *prodigies*). According on their ranks, the prodigies' portfolios are assigned linearly decreasing *amplifying factors* ranging from $\pi + 1$ down to 1. In addition we enlarge the group of role models by ε *elitists* all of which represent the best overall solution found so far. This is done by assigning the best portfolio found so far an amplifying factor of ε and treating it as an additional prodigy. Crystals of the current population that are not prodigies, have an amplifying factor of 0.

Replacement. To reinforce promising tendencies on the one hand and eliminate rather disappointing ones on the other, the ω worst crystals of the current population are replaced with crystals which are considered to have high potential. We distinguish two alternatives of high potential replacements:

Clone: Based on the amplifying factors, probabilities for selecting an existing portfolio are calculated such that the better a prodigy's fitness the higher the probability it is chosen. An unchanged replicate of this portfolio replaces the poor portfolio. The effect of cloning is that a new crystal starts off with a supposedly good structure but will evolve a different structure than its twin.

Averaged Idol: An average weight for each asset is calculated based on the elitists' and prodigies' portfolios. The weights w_{ict} of asset i in crystal c's portfolio are multiplied by the amplifying factors, added up, and normalized so that the overall sum is 1. These "averaged weights" are used for probabilities to select k assets and assign them weights that reflect these averaged weights (yet with a random component). The effect of this averaging is that an assets is preferred when it is found in many a prodigies'/elitists' portfolio and has a high weight in those portfolios. Unlike in usual genetic programming systems, there are not just two parents but a whole group of successful ancestors that pass on their endowment.

With a probability of p_c a "Clone" will be chosen, with a probability of $1 - p_c$ a newly generated crystal based on the "Averaged Idol" will be used for a replacement.

The algorithm stops after a fixed number of iterations and reports the best solution found, i.e., the last elitists' portfolio. The algorithm's runtime is merely influence by the population size, C, and the maximum number of different assets included in the portfolio, k, its complexity being $\mathcal{O}\left(C \cdot \left(\ln(C) + k^2\right)\right)$.

2.2 Variants

In the course of developing the reported version of the algorithm, we experimented with several modifications and extensions which were eventually turned down for different reasons.

Variants for the *modification* of portfolios included different ways of changing the weights and exchanging assets. Amongst these were alternative versions for calculating $\widetilde{z}_{ict}$ which was either distributed within a constant bandwidth (i.e., with constant $U_t = U$) or the distribution of which was distorted according to the Averaged Idol. Additional versions concentrated on the selection of a new asset j which was either perfectly random or orientated on the covariances σ_{ij}. All of these variants led to either lower reliability of the results or increased the runtime significantly without noticeable effect on the quality of results.

Variants for the *evaluation* stage included a nonlinear system for generating the amplifying factors, exclusion of the elitists and/or prodigies, and allowing all members of the current population to contribute to public knowledge and not just the prodigies. Both elitists and prodigies turned out to significantly improve the effectiveness of the algorithm in terms of convergence speed. A high number of agents that contribute their experience to the "Averaged Idol" merely increases runtime without apparent positive effect. Amongst the ranking system, the linear version of the amplifying factors turned out to be both simple and effective.

The *replacement* was found to be most effective when both clones of existing good portfolios and newly generated portfolios based on successful role models were allowed. To avoid the danger of getting stuck in a local optimum by "inbreeding" with extremely similar prodigies, we also introduced a third alternative were a random portfolio was generated independently of the other portfolios. Since the other portfolios have already passed a number of steps within the optimization process, this new portfolio was given an extra number of iterations. As turned out, however, this alternative mainly increased the runtime (because of the extra number of iterations) but had hardly any effect on the result since these new portfolios almost never made it into the group of the best portfolios with a portfolio structure that differed significantly from some already existing prodigies' portfolios.

3 Computational Study

3.1 Data and Parameters

In order to test the algorithm we use two data sets. The first contains the 30 stocks represented in the German Index DAX30, the second contains 96 stocks of the British FTSE100. In both cases we used daily quotes over the period July 1998 – December 1999. Based on the corresponding historic returns we calculated the covariances σ_{ij} which are used for estimators of future risk. The expected returns, r_i, were generated with a standard *Capital Asset Pricing Model* (*CAPM*)–approach[3] according to $r_i = 0.05 + 0.06 \cdot \beta_i$, i.e., with an expected safe return of 5 % and an expected market risk premium of 6 % and with beta coefficients, β_i, coming from the historic returns. The maximal number of assets to be included in the portfolio was in both cases $k \in \{5; 10; 15; 20\}$.

We ran some 25,000 experiments with random values for the parameters used in the algorithm. For each problem, i.e., for each k and data set, the parameters for the runs with the best results and the worst results were analysed in order to determine favorable values for the parameters. When regarding efficiency as a combination of high convergence speed and reliable results, we found the highest efficiency was achieved with a population size of 100 where the $\pi = 12$ best portfolios represent the prodigies and where the number of elitists is $\varepsilon = 100$. Each population had 250 generations to find the optimum, and in each iteration any of the 100 agents of the population built 4 modified portfolios the best of which was accepted according to the metropolis function. This implies, that in each population $250 \cdot 100 \cdot 4 = 100,000$ test portfolios were generated. For the modification of the weights the bandwidth for $\widetilde{z}_{ict}$ was $[-U_t; +U_t]$ with $U_0 = 0.3$ and $\gamma_U = 0.985$. The probability that an asset i with modified weight $w'_{ict} = 0$ was replaced with some new asset was $p_r = 0.4$. In the replacement stage the $\omega = 10$ worst agents were replaced with a clone (with probability $p_c = 0.3$) or a new agent based on the "Averaged Idol" (with probability $1 - p_c = 0.7$). For the SA part, the initial temperature was $T_0 = 750°$, and the cooling parameter was $\gamma_T = 0.9$. Implemented in *Delphi 5* with a 900 MHz Pentium III, the runtime was approx. 3 seconds for the smallest problems ($k = 5$) and approx. 15 seconds for the largest ($k = 20$).

3.2 Results

The lefthand graph in Figure 1 depicts the results of 944 runs for the case where a portfolio consisting of $k \leq 10$ out of the $N = 30$ securities represented in the DAX30 is to be selected. The graph on the

[3]For a detailed presentation of the *Capital Asset Pricing Model* (*CAPM*) see, e.g., [3].

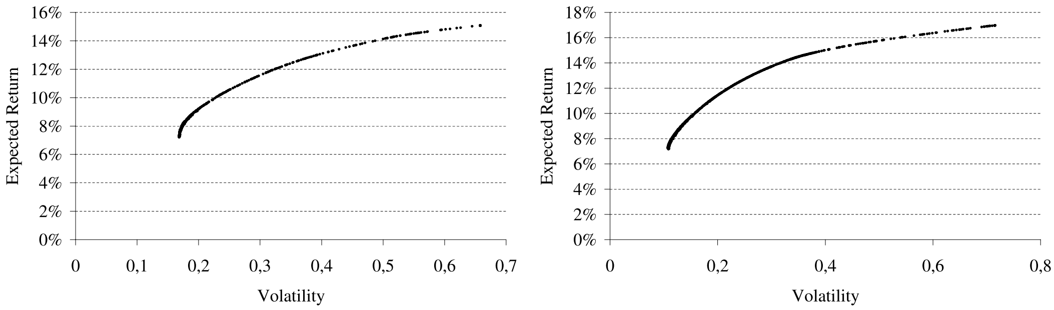

Figure 1: Results for the DAX30 data set with $k = 10$ and $N = 30$ (left) and for the FTSE10 data set with $k = 20$ and $N = 96$ (right)

right depicts the corresponding results of 2,410 runs for selecting $k \leq 20$ out of the $N = 96$ available FTSE100–stocks. As mentioned earlier, each run comprises 100,000 test portfolios. Note that in the first case there are $\sum_{k=1}^{10} \binom{30}{k} = 5.3 \times 10^7$ different valid combinations of stocks, in the second case there are $\sum_{k=1}^{20} \binom{96}{k} = 2.9 \times 10^{20}$ alternatives. For each of these alternatives, of course, the number of different valid combinations of weights is infinite. We want to emphasize that *all* the runs with the final set of parameters are reported and that no results were excluded for poor performance. As can be seen from the two figures, the algorithm finds highly reliable results. The portfolios form a very smooth line and there are no significant outliers in the sense that a portfolio has a lower expected return but a higher volatility than any other portfolio. The portfolios found by the algorithm can therefore be regarded as members of the so-called *Efficient Set*, i.e., the theoretical set of optimal portfolios. Outliers, i.e., *inefficient portfolios*, would lie below this line.

For both the DAX data set and FTSE data set, we ran the algorithm for $k = 5$, 10, 15, and 20. The results are analogous to those reported. We conclude from that the algorithm's efficiency and reliability are therefore virtually not influenced by the problem size which depends on the size of the market (as can be seen from the different sizes of the DAX and the FTSE data sets) and the number of assets to be selected.

4 Conclusion

In this paper we presented a metaheuristic which combines principles from Simulated Annealing with Evolutionary Strategies and which uses additional modifications. Having applied this algorithm we find that it yields highly efficient and reliable results for portfolio selection.

References

[1] B. Bullnheimer, R.F. Hartl, and C. Strauss. A New Rank Based Version of the Ant System — A Computational Study. *Central European Journal of Operations Research*, 7:25–38, 1999.

[2] 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.

[3] E.J. Elton and M.J. Gruber. *Modern Portfolio Theory And Investment Analysis*. John Wiley & Sons, Inc., 5th ed., New York et al. 1995.