COMISEF

Computational Optimization Methods in Statistics, Econometrics and Finance

- Marie Curie Research and Training Network funded by the EU Commission through MRTN-CT-2006-034270 -

COMISEF WORKING PAPERS SERIES

WPS-045 21/09/2010

Heuristic Strategies in Finance – An Overview

M. Lyra

Heuristic Strategies in Finance – An Overview*

Marianna Lyra†

September 21, 2010

Abstract

This paper presents a survey on the application of heuristic optimization techniques in the broad field of finance. Heuristic algorithms have been extensively used to tackle complex financial problems, which traditional optimization techniques cannot efficiently solve. Heuristic optimization techniques are suitable for non-linear and non-convex multi-objective optimization problems. Due to their stochastic features and their ability to iteratively update candidate solutions, heuristics can explore the entire search space and reliably approximate the global optimum. This overview reviews the main heuristic strategies and their application to portfolio selection, model estimation, model selection and financial clustering.

Keywords: finance, heuristic optimization techniques, portfolio management, model selection, model estimation, clustering.

*Financial support from the EU Commission through MRTN-CT-2006-034270 COMISEF is gratefully acknowledged.

†Department of Economics, University of Giessen, Germany.

Contents

List of Tables

List of Figures

1 Introduction

The financial paradigm entails simple mathematical models but often imposes unrealistic assumptions. Basic financial models consist of linear and/or quadratic functions which ideally can be solved by traditional optimization techniques like quadratic programming. However, the finance community soon realizes the complexity of the economic environment and the market inefficiency. To cope with this, the financial community develops more realistic mathematical models which include more complex, often non-convex and non-differentiable functions with real world constraints. Under these more challenging conditions quadratic programming routines or other classical techniques will often not result in the optimal solution (or even in a near-optimum).

Heuristic optimization methods[1] are flexible enough to tackle many complex optimization problems (without a linear approximation of the objective function). Heuristics' flexibility stems either from the random initialization or from the intermediate stochastic selection and acceptance criterion of the candidate solutions. However, the optimization results should be carefully interpreted, due to the random effects introduced during the optimization process. Besides, repetitions of the algorithm might generate heterogenous solutions from an unknown distribution. To approximate the optimum solution without any prior knowledge of the distribution, Gilli and Winker (2009) suggested repeating an experiment with different parameter settings or applying extreme value theory. Also, Jacobson *et al.* (2006) developed a model to approach the unknown distribution of the results.

This paper examines the modern financial problems of portfolio selection, robust model estimation and selection and financial clustering. It analyzes research papers published during the last two decades, where alternative heuristic techniques were applied independently or complementarily to other computational techniques. Furthermore, this overview extends the contributions collected in Gilli *et al.* (2008), who presented the most popular heuristics and their usage in finance. Moreover, this work emphasizes and distinguishes the implementation characteristics of several applications.

The paper is organized as follows: Section 2 reports on heuristic and hybrid meta-heuristic methods and describes the basic algorithms. Also, it discusses some implementation issues that should be considered when applying heuristics. Section 3 introduces in more detail the technical issue of parameter setting in heuristic optimization algorithms. In due course, an

[1]The expressions 'heuristic optimization methods', 'heuristic strategies', 'heuristic techniques', 'meta-heuristics' and 'heuristics' will be used interchangeably throughout the paper.

established methodology, Response Surface Methodology (RSM) (Box and Draper (1987)), is used for parameter calibration. To the best of my knowledge, RSM has not been applied so far in the context of parameter tuning for heuristics. Section 4 addresses contemporary problems in finance and analyzes how heuristics can be applied to approach and overcome them. Finally, Section 5 concludes and suggests further applications of heuristics to open financial questions.

2 Heuristic Optimization Techniques

Heuristic optimization techniques form part of the broad category of computational 'intelligent' techniques which have, in principle, solved non-linear financial models. In comparison with other intelligent techniques, heuristics need little parameter tuning and they are less dependant on the choice of starting solutions. Heuristics provide an alternative or optimize the current intelligent techniques, like Neural Networks (NN) (Ahn *et al.* (2006)). The interaction of heuristics with other intelligent techniques is discussed in Section 4.

A diverse range of optimization heuristics have been developed to solve non-linear constrained financial problems. The first application of heuristics in finance can be found in the work of Dueck and Winker (1992), who used Threshold Accepting (TA) for portfolio optimization with different utility criteria. Their basic methodology is described in Section 4.1. This overview explains the basic optimization heuristic algorithms applied in the recent literature to financial problems.

The two main classes of heuristic algorithms are construction heuristics and improvement (or local search) heuristics. Construction heuristics start from an empty initial solution and construct, e.g. by the means of nearest neighborhood, the candidates that form an optimum solution (Hoos and Stützle (2005), ch.1). Local search heuristics iteratively update a set of initial candidates that improve the objective function. While constructive methods are easy and fast to implement and can be applied to problems like scheduling and network routing, they do not appear in practical financial problems. This might be due to their tendency to local optima and their problem-specific performance.

This chapter addresses only local search methods. Figure 1 decomposes the local search methods to trajectory methods and population based methods. The mechanisms of neighborhood structure, as well as the number of solutions that are updated simultaneously is what distinguish local search methods. Here we present the four most common local search heuristics

found in the financial literature. For an extensive description of heuristic optimization techniques see also Winker (2001), ch.5.

2.1 Trajectory Methods

2.1.1 Simulated Annealing

Simulated Annealing (SA) as well as TA (section 2.1.2) are threshold methods that accept a neighboring solution based on a threshold value. In that sense, they can accept not only improvements but also controlled impairments of the objective function value as long as a threshold is satisfied. SA is inspired by the principals of statistical thermodynamics, where the changes (transformations) in the energy are related to macroscopic variables, i.e. temperature. SA relates the change in the objective value to a probability threshold. The probability value is subject to a macroscopic variable, T. T is iteratively changed and the best objective function value is reported. Algorithm 1 describes the general outline of a SA implementation.

Initially, the algorithm randomly generates a solution (2:). Next, for a predefined number of iterations, I, the following steps are repeated. In every iteration a new neighboring solution is randomly generated (4:). The algorithm always accepts the new solution if it results in an improvement of the objective function value. However, an impairment is also accepted with a diminishing probability (6:), $e^{-\Delta/T} > u$. u is a uniformly distributed value lying in the interval $[0, 1]$. The new solution is carried onto the next iteration and is used as a benchmark.

The probability value is influenced by changes in the objective function value and the 'temperature' T. Ceteris paribus, the higher the (negative) change, Δ, the lower the probability that an impairment will be accepted. The 'temperature' T controls the tolerance of the algorithm in accepting downhill moves. The lower its value is, the less tolerant it is to negative changes. This is required, especially at the end of the search, where we believe we are closer to the optimum solution and impairments are unwelcome.

By allowing moves towards worse solutions, the algorithm has a chance to escape local minima. The number of iterations or the properties of the temperature define the stopping criterion. In the latter case the temperature reduces until an equilibrium state is found where no further change in the objective function value can be observed.

2.1.2 Threshold Accepting

Dueck and Scheuer (1990) and Moscato and Fontanari (1990) conceived the

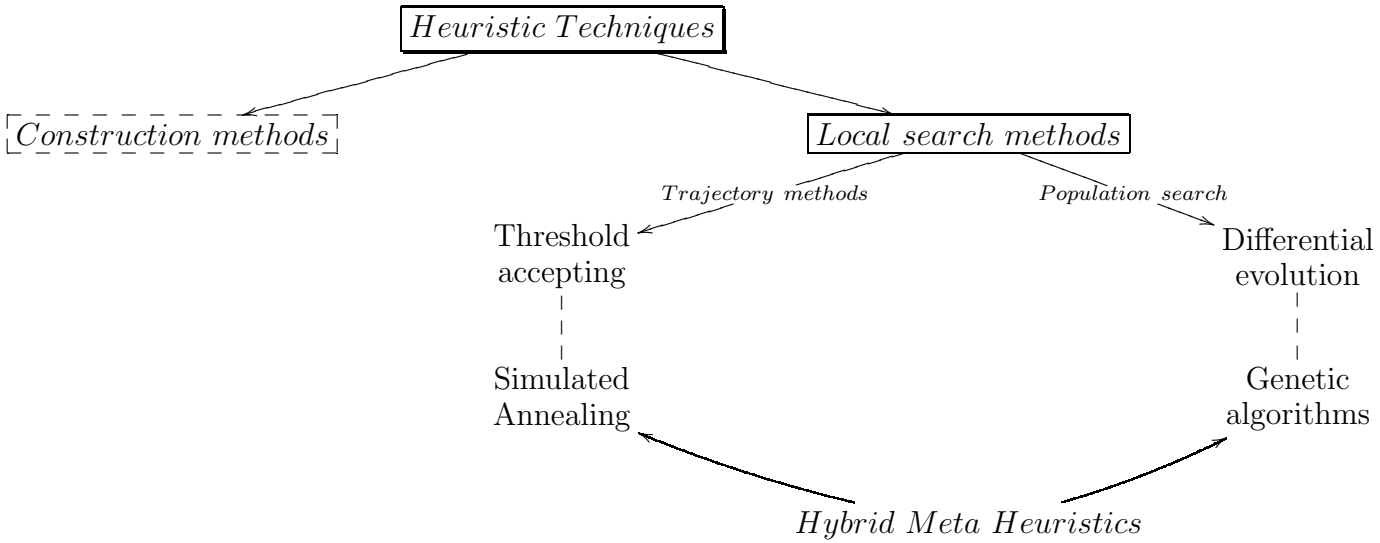


Figure 1: Decomposition of heuristic optimization techniques

Algorithm 1 General Simulated Annealing Algorithm.

```
Initialize I, T
Generate at random a solution χ^0
for r = 1 to I do
    Generate neighbor at random, χ^1 ∈ N(χ^0)
    Compute Δ = f(χ^0) − f(χ^1)
    if Δ < 0 or e^{−Δ/T} > u then
        χ^0 = χ^1
    end if
    Reduce T
end for
```

idea of building a simple search tool appropriate for many types of optimization problems named TA. Like SA, TA enables the search to escape local minima by accepting not only improvements, but also impairments of the objective function value. Nonetheless, the impairment acceptance is not a probabilistic decision but rather a given threshold sequence. More precisely, until the stopping criterion is met, the current candidate solution is compared with a neighboring solution ($\Delta = f(\chi^0) - f(\chi^1)$). A threshold value ($\tau$) determines to which extent not only local improvements, but also local impairments are accepted ($\Delta < \tau$). For a comprehensive overview of TA, its parameter settings and its convergence properties, see Winker (2001).

Winker and Fang (1997) and Winker and Maringer (2007) suggested a data driven approach for the construction of the threshold sequence. The threshold sequence is ex-ante constructed based on the search space. The local differences of the fitness function are sorted in descending orders. Thus, they represent a diminishing threshold sequence. The reduction of the neighborhood allows a wider search space at the beginning of the search and a greedy search towards the end. Moreover, Lyra *et al.* (2010*b*) proposed a threshold sequence based on the differences in the fitness of candidate solutions that are found in a specific place of the search process. Local differences actually calculated during the optimization run are considered. As a result, the threshold sequence adapts to the region where the current solution belongs and to the objective function used. By using a moving average, a smooth threshold sequence is obtained. In addition, the thresholds decrease linearly with the number of iterations.

2.2 Population based Methods

2.2.1 Genetic Algorithms

Genetic Algorithms (GA) are one of the oldest evolutionary optimization methods. They were developed by Holland (1975). They have been applied to numerous financial problems including trading systems, stock and portfolio selection, bankruptcy prediction, credit evaluation and budget allocation. Their wide applicability springs from generating new candidate solutions using logical operators and from evaluating the fittest solution. A thorough explanation of the algorithm follows.

Algorithm 2 depicts the implementation details of GA. First in (2:), a population n_p of initial solutions $P_{j,i}^{(0)}$ is randomly initialized. Usually, the solutions are encoded in binary stings but, other representations can be used. The objective function value (or the fitness value) is estimated for every initial solution. Second, the algorithm iteratively generates n_G new candidate solutions, 'chromosomes', as follows: The two fittest candidate solutions $P_{.,r_1}^{(0)}$ and $P_{.,r_2}^{(0)}$, 'parent chromosomes', are chosen from the initial set to generate a new candidate solution, 'reproduce'. The reproduction or update consists of two mechanisms, the recombination of existing elements, 'genes', crossover and random mutation.

Algorithm 2 Genetic Algorithms.

1: Initialize parameters n_p, n_G
2: Initialize and evaluate population $P_{j,i}^{(0)}$, $j = 1, \cdots$, d, i $= 1, \cdots, n_p$
3: **for** $k = 1$ to n_G **do**
4: **for** $i = 1$ to n_p **do**
5: Select r_1,$r_2 \in 1, \cdots, n_p$, $r_1 \neq r_2 \neq i$
6: Crossover and mutate $P_{.,r_1}^{(0)}$ and $P_{.,r_2}^{(0)}$ to produce $P_{.,i}^{(u)}$
7: Evaluate $f(P_{.,i}^{(u)})$
8: **if** $f(P_{.,i}^{(u)}) < f(P_{.,i}^{(0)})$ **then**
9: $P_{.,i}^{(1)} = P_{.,i}^{(u)}$
10: **else**
11: $P_{.,i}^{(1)} = P_{.,i}^{(0)}$
12: **end if**
13: **end for**
14: **end for**

There are alternative ways to crossover which are problem-specific and can improve the performance of GA. The simplest way is to choose randomly a crossover point and copy the elements before this point from the first parent and the elements after the crossover point from the second parent. This way, we construct the first child chromosome. The second child chromosome

$$\downarrow_{\text{crossover-point}}$$

$$parent_1 = \big(1 \quad 1 \quad 0 \quad 1 \quad | \quad 0 \quad 1 \quad 0 \quad ... \quad 1\big)_{1\times k}$$

$$parent_2 = \big(\mathit{1} \quad \mathit{0} \quad \mathit{1} \quad \mathit{0} \quad | \quad \mathit{1} \quad \mathit{1} \quad \mathit{0} \quad ... \quad \mathit{1}\big)_{1\times k}$$

$$child_1 = \big(1 \quad 1 \quad 0 \quad 1 \quad | \quad \mathit{1} \quad \mathit{1} \quad \mathit{0} \quad ... \quad \mathit{1}\big)_{1\times k}$$

$$child_2 = \big(\mathit{1} \quad \mathit{0} \quad \mathit{1} \quad \mathit{0} \quad | \quad 0 \quad 1 \quad 0 \quad ... \quad 1\big)_{1\times k}$$

Figure 2: The crossover mechanism in GA.

combines the second part of the first parent chromosome and the first part of the second parent chromosome. Figure 2 shows such a simple crossover procedure. An alternative crossover operators is to select more than one random crossover points (Back *et al.* (1996)) or to apply a uniform crossover (Savin and Winker (2010)).

To complete the update, a random mutation of the elements takes place. During mutation the new elements change randomly. In the case of binary encoding a few randomly chosen bits change from 1 to 0 or from 0 to 1. Hence, for each member of the population a new candidate solution, 'children or offspring', is produced and evaluated (7:). In some implementations of GA a number of elite solutions, 'elite chromosomes', are transferred unmodified to the next step. These are solutions with distinguished characteristics like good fitness value. Finally, only the solutions with the best objective function value survive the generation evolution.

2.2.2 Differential Evolution

Similar to GA, Differential Evolution (DE) is a population based technique, originated by Storn and Price (1997), but more appropriate for continuous problems. The attractiveness of this technique is the little parameter tuning needed.

During the initialization phase, Algorithm 3 (1:), the population size n_p and the generation number n_G should be determined. Two technical parameters F and CR should also be initialized at this stage. In order to achieve convergence, n_p should be increased more than ten times the number

Algorithm 3 Differential Evolution.

1: Initialize parameters n_p, n_G, F *and* CR
2: Initialize population $P^{(1)}_{j,i}$, $j = 1, \cdots$, d, i $= 1, \cdots, n_p$
3: **for** $k = 1$ to n_G **do**
4: $\quad P^{(0)} = P^{(1)}$
5: $\quad$ **for** $i = 1$ to n_p **do**
6: $\quad\quad$ Generate r_1,r_2,r_3 $\in 1, \cdots, n_p$, $r_1 \neq r_2 \neq r_3 \neq i$
7: $\quad\quad$ Compute $P^{(v)}_{\cdot,i} = P^{(0)}_{\cdot,r_1} + F \times (P^{(0)}_{\cdot,r_2}$ - $P^{(0)}_{\cdot,r_3})$
8: $\quad\quad$ **for** $j = 1$ to d **do**
9: $\quad\quad\quad$ **if** $u < CR$ **then**
10: $\quad\quad\quad\quad P^{(u)}_{j,i} = P^{(v)}_{j,i}$
11: $\quad\quad\quad$ **else**
12: $\quad\quad\quad\quad P^{(u)}_{j,i} = P^{(0)}_{j,i}$
13: $\quad\quad\quad$ **end if**
14: $\quad\quad$ **end for**
15: $\quad\quad$ **if** $f(P^{(u)}_{\cdot,i}) < f(P^{(0)}_{\cdot,i})$ **then**
16: $\quad\quad\quad P^{(1)}_{\cdot,i} = P^{(u)}_{\cdot,i}$
17: $\quad\quad$ **else**
18: $\quad\quad\quad P^{(1)}_{\cdot,i} = P^{(0)}_{\cdot,i}$
19: $\quad\quad$ **end if**
20: $\quad$ **end for**
21: **end for**

of parameters to be estimated,[2] while the product of n_p and n_G defines the computational load. Price *et al.* (2005) in their book about DE state that, although the scale factor F has no upper limit and the crossover parameter CR is a fine tuning element, both are problem specific. Winker *et al.* (2010) in an application of DE propose values for $F = 0.8$ and $CR = 0.9$. Their calibration procedure as well as a more statistically oriented methodology are explained in Section 3.

As mentioned above, the initial population of real numbers is randomly chosen and evaluated (2:). Then, for a predefined number of generations the algorithm performs the following procedure. According to the characteristics of a population based approach a set of population solutions $P^{(0)}_{\cdot,i}$ is updated. The algorithm updates a set of parameters through differential mutation (7:) and crossover (9:) using arithmetic instead of logical operations.

While both in GA and DE the mutation serves as a mechanism to overcome local mimina their implementation differs. Differential mutation generates a new candidate solution by linearly combining some randomly chosen elements and not by randomly mutated elements, as GA does, whereas each element of the current solution crosses with a uniformly designed candidate

[2] A practical advice for optimizing objective functions with DE given on *www.icsi.berkeley.edu/ ∼ storn/*.

and not with an existing one.

Particularly, differential mutation constructs new parameter vectors $P_{.,i}^{(\upsilon)}$ by adding the scaled difference of two randomly selected vectors to a third one. (7:) demonstrates the 'metamorphosis' (modification), where F is the scale factor that determines the speed of shrinkage in exploring the search space.

Further, during crossover, (9:), DE combines the initial elements with the new candidates. With a probability CR, each component $P_{j,i}^{(0)}$ is replaced by a mutant one $P_{j,i}^{(\upsilon)}$ resulting in a new trial vector $P_{j,i}^{(u)}$ (Figure 3).

Finally, the value of the objective function $f(P_{.,i}^{(u)})$ of the trial vector is compared with that of the initial element $f(P_{.,i}^{(0)})$. Only if the trial vector results in a better value of the objective function, it replaces the initial element in the population. The above process repeats until all elements of the population are considered and for a predefined number of generations.

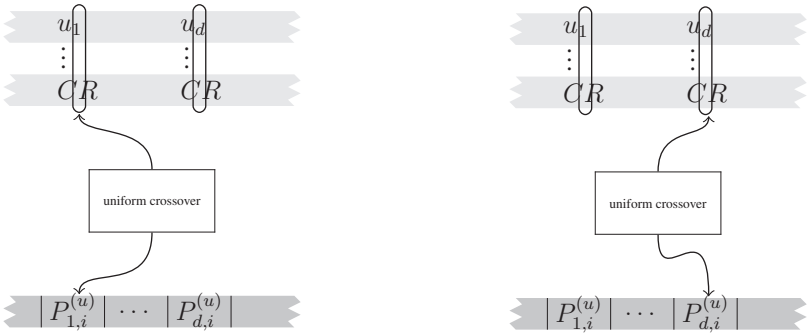


Figure 3: Crossover representation for the the first and the last parameter

2.3 Hybrid Methods

2.3.1 Hybrid Meta-Heuristics

Heuristic optimization algorithms, also known as meta-heuristics in their general setting, are suitable for optimizing many combinatorial problems. Though, as different meta-heuristics exhibit different (advantageous) characteristics, a combination of them can be beneficial for finding the optimal solution. When two or more meta-heuristics are combined, new hybrid computational intelligent techniques are constructed. The basic advantage of hybrid meta-heuristics is that they can combine the outstanding characteristics of specific heuristics.

Numerous hybrid meta-heuristics can be constructed when combining different heuristics and their characteristics. The taxonomy based on hybrids' design (algorithm architecture) merges a hierarchical and an independent scheme (see Talbi (2002)). Figure 4 presents this classification. The first two columns represent the hierarchical classification, while the last column the flat scheme. A low-level hybrid crosses components of different meta-heuristics. In a high-level method the meta-heuristics are autonomous (column 1). Both a high and a low level methods distinguish relay and cooperative hybridization (column 2). In the latter group the heuristics cooperate and in the former group they are combined in a sequence.

Furthermore, hybrid meta-heuristics can be categorized based on other characteristics, independent of the hierarchy. These are homogenous or heterogenous, global or partial, and general or special. When identical in structure meta-heuristics are used, a homogenous hybrid is constructed. Two different meta-heuristics construct a heterogenous hybrid. When the whole search space is explored by heuristics we have a global hybridization. In a partial hybrid, different heuristics explore only a part of the solution space. General hybrid algorithms are suitable for solving the same type of problems, whereas special hybrids combine meta-heuristics that solve different types of problems. Besides their advantages, hybrids should be applied with caution since algorithm performance will be well affected by parameter tuning, which becomes more intense as compared to single heuristics.

Two characteristics distinguish optimization heuristics; the number of solutions they iteratively update and the mechanism for this update. The first characteristic refers to the distinction between trajectory search and population search heuristics and the second to the mechanism of neighborhood solution construction. Let us take a threshold heuristic as an example, i.e. TA, and a population based heuristic, i.e. GA. The new hybrid resulted from a composition of a threshold and a population based method is a high-level or

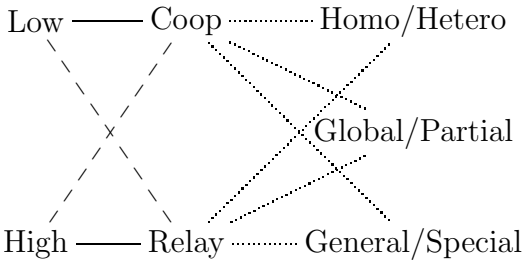


Figure 4: Classification of Hybrid Meta-Heuristics

heterogenous, global, general hybrid also known as Memetic Algorithm (MA) (Moscato (1989)). On the one hand, a GA can explore efficiently a wider search space using the cross over and mutation mechanisms. On the other hand, TA can well escape local minima by its acceptance criterion. It accepts neighborhood solutions even if they result in a deterioration, as long as they do not exceed the threshold value. Thus, combining these meta-heuristics can be beneficial for the algorithm convergence. In the next sections some applications of hybrid meta-heuristics are described, e.g. Maringer (2005) used a MA in the context of risk factor and portfolio selection.

2.3.2 Other Hybrids

Often in the literature, additional hybridizations of meta-heuristics with either constructive methods or other intelligent techniques can be found. Some selected applications are presented in the following.

Gilli and Winker (2003) constructed a hybrid optimization technique using the Nelder-Mead simplex search and the TA algorithm. They model the behavior of foreign exchange (DM/US-$) market agents while optimizing the Agent Based Model (ABM) parameters. The Nelder-Mead simplex search (Lagarias *et al.* (1999)) has the advantage of finding an optimum solution with few computational steps.[3] Starting from $d + 1$ (in this application 3) initial solutions, the search moves to a better solution that is a reflection of this initial triangle or makes bigger steps if the reflection results in an improvement of the objective function value. However, if the simplex method cannot find a reflection point that outperforms the initial parameter settings, the search space shrinks and the algorithm gives a solution which is nothing more than a local optimum. To overcome the problem, the authors applied the acceptance criterion of TA (Section 2.1.2). Using that technique they showed that interactions between market agents could realistically represent foreign exchange market expectations.

[3]The simplex search chooses an efficient step size.

In a different application, Trabelsi and Esseghir (2005), Ahn *et al.* (2006) and Min *et al.* (2006) used GA together with other intelligent technics to select an optimal set of bankruptcy predictors. Besides model selection, these papers applied GA to tune the parameters of bankruptcy prediction models. Trabelsi and Esseghir (2005) constructed a hybrid algorithm which evolves an appropriate (optimal) artificial neural network structure and its weights. Ahn *et al.* (2006) and Min *et al.* (2006) forecast bankruptcy with Support Vector Machines (SVM). GA selected the best set of predictors and estimated the parameters of a kernel function to improve SVM performance using data from Korean companies. The latter paper stressed the potentials of incorporating GA to SVM in future research.

Varetto (1998) applied GA to train NN. The meta-heuristic was used not only for model selection, but also for weight estimation.

Despite of the potentials of meta-heuristics in tuning or training other techniques, e.g. NN and SVM, they should be applied with caution. First, the more unknown parameters a heuristic algorithm has, the more complex an algorithm is. Hence, it will be difficult to evaluate the consistency of the results. So, keeping the implementation simple is more advisable. Second, different heuristics are suitable for different types of problems. Selecting the appropriate heuristic technique will improve the performance of the hybrid, e.g. the application of DE, instead of GA, in weight estimation might improve both the efficiency and the computational time of the hybrid since DE appears to be more suitable for continuous problems.

2.4 Technical Details

2.4.1 Constraint Handling Techniques

Real world financial problems face simultaneously several constraints. Heuristics are able to consider almost any kind of constraints, resulting generally in good approximations of the optimal solution. However, the presence of constraints makes the search spaces discontinuous and finding a solution that satisfies all the constraints might be a demanding task. Several methods to handle constraints are suggested in the literature so as to ensure at the end a feasible solution.

When running the optimization heuristics, the problem constraints have to be taken into account. To this end, alternative methods can be considered: rewriting the definition of domination, such that it includes the constraint handling; imposing a penalty on infeasible solutions; applying a repair function to find the closest feasible solution to an infeasible one.

The first possibility has been described for the application of DE in credit

risk assignment in Krink *et al.* (2007). The intuitive idea of this constraint handling technique is to leave the infeasible area of the search space as quickly as possible and never return. For minimization problems, the procedure can be described as follows within Algorithm 3, Section 2.2.2:

1. If the new candidate solution $P_{j,i}^{(u)}$ *and* the current candidate solution $P_{j,i}^{(0)}$ satisfy the side constraints, $P_{j,i}^{(u)}$ replaces $P_{j,i}^{(0)}$ if its fitness $f(P_{j,i}^{(u)})$ satisfies the condition $f(P_{j,i}^{(u)}) \leq f(P_{j,i}^{(0)})$.

2. If only one candidate solution is feasible, select the feasible one

3. If both solutions violate constraints, ...

   (a) ... select the one that violates fewer constraints.

   (b) ... if both solutions violate the same number of constraints, $P_{j,i}^{(u)}$ replaces $P_{j,i}^{(0)}$ if its fitness $f(P_{j,i}^{(u)})$ satisfies the condition $f(P_{j,i}^{(u)}) \leq f(P_{j,i}^{(0)})$.

In the second possibility, a penalty technique allows infeasible candidate solutions while running the algorithm as a stepping stone to get closer to promising regions of the search space. In this case, a penalty term is added to the objective function. Solutions should be penalized the more they violate the constraints.

Dueck and Winker (1992) (Section 4.1) provide an implementation of TA where infeasible solutions are accepted to pass on to the next step by imposing a punishment function. The penalty function is added to the objective function (utility function), which ensures at the end a feasible solution. Lyra *et al.* (2010*b*) (Section 4.4) implemented TA for credit risk assignment with both the constraint-dominated handling technique and a penalty technique. To guarantee a feasible solution at the end and avoid early convergence to a local minimum, the penalty was increased over the runtime of the algorithm. An increasing penalty function was also applied by Gimeno and Nave (2009) (Section 4.2) in an GA implementation to parameter estimation.

In contrast, Krink *et al.* (2009) in an application of DE to index tracking (Section 4.1.4), applied a repair function to find the closest feasible solution to an infeasible one.

Generally, Lyra *et al.* (2010*b*) found that the constraint-dominated handling technique performed well while taking comparatively little computation time. However, depending on the kind of objective function used the penalty technique may improve the reliability of heuristics, i.e. reduce the variance of the results obtained. Depending on the type of the objective function,

different penalty weight functions can be applied, e.g. linear, exponential, increasing or decreasing with the iteration number. Since calculation of the penalty weight function might increase the CPU time, a rather simple penalty function is recommended.

2.4.2 Calibration Issues

The selection of parameter values for a heuristic optimization algorithm is essential for its efficiency. These parameters are the iteration number, the neighborhood structure and the new solution (candidate) acceptance criterion. A simple, but sometimes time consuming way to select the parameter values is to iteratively run the algorithm for different parameter values and then extract some descriptive statistics (Maringer (2005)). Nonetheless, a regression analysis or a Response Surface Analysis (Box and Draper (1987))) can also be tested. For a small number of parameters, the above methods are efficient and relatively fast in finding the optimal parameter settings for a heuristic algorithm.[4] Alternatively, in higher dimensions (higher number of parameters) a heuristic can optimize the parameters of another heuristic.

Winker *et al.* (2010) implemented DE for LMS estimation of the CAPM and the multifactor model of Fama-French. A detailed description of this optimization problem is given in Section 4.2.2. While DE, as any optimization heuristic technique, should not depend on the specific choice of starting values, the optimal combination of the technical parameters, F and CR, determines the speed of convergence to the global solution and the variability of the results, respectively. To find the optimal combination of F and CR, the authors run the algorithm for different combinations of F and CR. The procedure is illustrated in Algorithm 4 for parameter values ranging from 0.5 to 0.9.

Algorithm 4 Calibration Issues.

1: Initialize parameters n_p, n_G
2: Initialize population $P_{j,i}^{(1)}$, $j = 1, \cdots$, d, i $= 1, \cdots, n_p$
3: **for** $F = 0.5, \cdots, 0.9$ **do**
4: **for** $CR = 0.5, \cdots, 0.9$ **do**
5: Repeat Algorithm 3 from line 3-21
6: **end for**
7: **end for**

Figure 5 exhibits the dependence of the best objective value obtained for

[4]See Section 3 for an application of Response Surface Analysis to the estimation of DE parameters in an application to Least Median of Squares (LMS) estimation of the CAPM and a multifactor model.

different combinations of F and CR for the LMS objective function (Section 4.2.2) using the first 200 daily stock returns of the IBM stock starting on January, 2nd, 1970. The population size n_p and the number of generations n_G are set to 50 and 100, respectively. The left side of Figure 5 presents the results for a single run of the algorithm, while the right side shows the mean over 30 restarts. Although the surface is full of local minima for CR below 0.7, it becomes smoother as CR reaches 0.8 independent of the choice of F. The results clearly indicate that for higher values of CR, results improve, while the dependency on F appears to be less pronounced. Based on these results, values of $F = 0.8$ and $CR = 0.9$ are proposed for estimating the parameters of the CAPM and the Fama-French factor model.

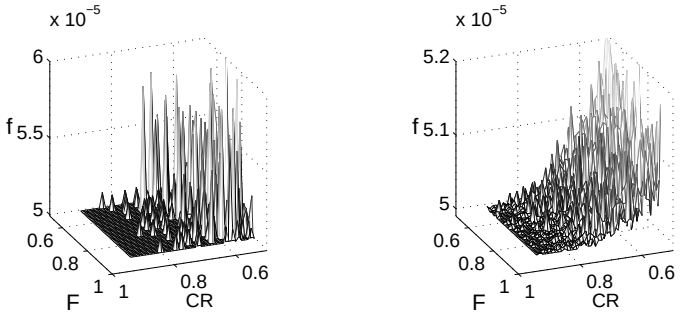


Figure 5: Calibration of technical parameters.

Instead of repeating a core algorithm numerous times,[5] a more efficient experimental design methodology can be applied, namely RSM.

3 Response Surface Analysis for Differential Evolution

3.1 Response Surface Analysis

RSM helps explore the relationship between the objective function value and the level of parameters using only few level combinations (Box and Draper (1987)). The aim of RSM is to find the parameters value with little tuning and without precise prior knowledge of their optimum value.

RSM consists of three main stages. In the first stage, the input variables are initialized. The initialization is based on a prior knowledge about the best

[5] In this case 41×41 for all combinations of F and CR ranging in 0.5:0.01:0.9.

combination of parameters that will optimize the objective function. These values constitute the reference or the center points. Next, the experimental domain is specified. It is based on orthogonal and/or rotatable experimental designs.[6] Inside this domain the factors are assigned specific values (levels), at which the response variable is estimated.

In the final stage, a second order polynomial is fitted using the designed experiments. After differentiation, this results in the optimal set of parameters. The reason for choosing a polynomial relationship is that no prior information about the existing relationship between the input and the response variable is needed.

$$y_i = \beta_0 + \sum \beta_i X_i + \beta_{ii} X_i^2 + \beta_{ij} X_i X_j + \varepsilon \,, \tag{1}$$

where

y_i	minimum median of squared residuals for $i-th$ setup of input variables
$\beta' s$	parameters of equation
$X' s$	input variables for setup i
ε	residual .

To find the optimal combination of factors' value that optimize the objective function of interest one should first understand how the objective function value changes with possible changes in the factors' level. To discover the relationship, the most common process is trial and error. By randomly changing the factor values (either simultaneously or one at a time) one can investigate the reaction of the objective function value. If the lower and upper bounds of the factor levels are known, one can evaluate the objective function for these factor combinations and then make some inference about the possible relationship between the parameters and the objective function. Alternatively, one can model the relationship by considering either a linear or a non-linear shape. When the shape of the relationship is not known a priori, for three factors, a second order polynomial relationship can be tested (see Equation 1).[7] This process approximates either linear or quadratic relationship.

3.2 Implementation Details for Differential Evolution

RSM explores the relationship between several input variables (factors) and a response output. In this exercise, the performance of RSM is tested in finding

[6] Orthogonal and rotatable designs are described in detail in what follows.

[7] A higher order polynomial can be tested when there are enough data points.

the optimal combination of DE parameters value in the specific application of obtaining the CAPM parameters to minimize the median of squares error.

The input variables are the (scaling) factor F, the crossover probability CR and the population size n_p. Note that for DE the product of the population size n_p and number of generations n_G determines the computational complexity of the algorithm. Hence, the aim is to find the optimal combination of the parameters F, CR and n_p for a given computational load $n_p \times n_G$. For that, the computational load ($n_p \times n_G$) is kept constant. Three different levels of computational resources are considered, with $n_G \times n_p$= 500, 2500 and 5000 meaning, low, medium and high complexity, respectively. Since the product of $n_p \times n_G$ should be constant, an adjustment is made on n_G in each alteration.

The response variable is the minimum value obtained for

$$\min_{\alpha,\beta}(\text{med}(\varepsilon_{i,t}^2)), \tag{2}$$

where $\varepsilon_{i,t} = y_{i,t} - \alpha - \beta x_{i,t}$ are the residuals of a factor model.

To specify the factor levels that help estimate the parameters of Equation 1, a Central Composite Design (CCD) is applied. CCD is an experimental method that efficiently builds a second order model for the response variable (Barker (1985)). Efficiency means to obtain a representative relationship between the parameters and the objective function of interest with the least parameter tuning.

CCD contains five level-combinations. First, a two level full factorial design assigns two levels, a low and a high level, to the three factors (input variables). Thus, eight experiments result from the two level-combinations (2^3) in our problem. A two level factorial combination alone can capture a possible linear relationship between the input variable and the response output. The two levels range symmetrically around a center point (e.g. $F_{center}\pm 0.1$). In addition to the two level factorial design, three additional levels are assigned to the factors. That is, first, a level that represents the best known value for the factors (F_{center}). Also, in order to expand the experimental domain, two additional levels are considered, both above and below the high and low values considered above.

By using five levels for each factor a curvilinear relationship, even up to a fourth level, or a quartic relationship can be investigated. Besides, we are able to search the optimal factor levels over a wider experimental domain. The efficiency of the CCD stems from the fact that it uses only 18 level combinations to calibrate the input variables. Alternatively, a five level full factorial design uses $5^3 = 125$ level-combinations. The Appendix gives a detailed explanation and a numerical representation of the CCD experiments.

For each level-combination of input variables a response value is obtained, the minimum median of squared residuals, Equation 2. To control for the randomness in the heuristic's output, the experiments are repeated 10 times. The mean response value (mean out of 10 repetitions of Equation 2) for each combination is reported. Using the response values from all setups (experiments), a second order polynomial is fitted.

Factors' optimal values are determined by differentiating the fitted second order polynomial with respect to each input variable, Equation 3.[8]

$$\frac{\partial y}{\partial X_i} = \beta_i + 2\beta_{ii}X_i + \beta_{ij}X_j = 0 \tag{3}$$

Then, the optimal combination of input variables is used to estimate Equation 2. The median of squares residuals for CAPM estimation is compared with the best known optimum reported by Winker *et al.* (2010).

3.3 Output

Tables 1 and 2 report the optimal combination of DE parameters, as suggested by RSM using orthogonal and rotatable designs, respectively. We report results for different computational complexities, low, medium and high (columns 1-4). The optimal values of n_p are rounded up to the nearest integer. Since the values of n_p are large enough, the rounding does not affect the mathematical results of the design (does not influence dramatically the objective function value). The reported optimal parameter combinations are used to evaluate the performance of DE. For that, Equation 2 is evaluated and the median of squares residuals for CAPM estimation is reported. The evaluation is repeated 30 times to control for the stochastic effects of heuristics. Columns 5 to 11 report for each set of parameters the best value, the median, the worst value, the variance, the 5th percentile, the 90th percentile, and the frequency of the best value occurs over 30 repetitions of the algorithm.

Table 1 presents the results of RSM with orthogonal design. The output suggests that the level of the scale factor F is irrelevant for the convergence of DE especially when CR is above 0.7. With n_p at least ten times the number

[8]For this step the build-in Matlab 7.6 function, quadprog is used. This tool uses a reflective Newton method for minimizing a quadratic function. It is effective for quadratic functions when only bound constraints are considered. In our problem the lower bound of F, CR, n_p reflect the lower level CCD design, while the upper bounds are 2, 1 and inf, respectively. The upper bound of F is based on Price *et al.* (2005), CR is a probability, whereas n_p has no real upper limit. The choice of lower bounds prevents the Newton method from stacking in a local optimum.

of parameters of the objective function (parameters of CAPM) and *CR* above 0.7, the algorithm can converge to the best results reported in Winker *et al.* (2010). Although, for medium computational load the algorithm finds the global optimum just once, in 13 other cases it approaches the optimum with accuracy of 4 decimals. For high computational complexity the algorithm is more stable and results in the best reported optimum (Winker *et al.* (2010)) in all 30 repetitions.

Table 2 reports the results of RSM for rotatable design. RSM results in slightly different optimal parameter settings. Yet, n_p must be at least ten times the number of parameters in order for the algorithm to reach the global optimum. For medium computational complexity the optimum is identical with that of orthogonal design with lower variance in 30 repetitions. Similarly, for high computational complexity the global optimum can be found, but not repetitively. Generally, n_p is an important parameter in the convergence of DE and *CR* stabilizes the algorithm.

RSM is proven to be a useful starting tool for calibrating the parameters of DE that best minimize the median of squares residual (Equation 2) without using high computational resources. Of course, one should bear in mind that the analysis is sensitive to the starting values (centers) as well as the dispersion of the high and low design points around the center. Shrinking or widening the distance between the design points (distance of factorial and orthogonal designs from center) might result in highly uncertain parameter estimates.

Possible extensions of the approach might include different formulations (or transformations), other than a polynomial one, to represent the relationship between the parameters and the objective function. Also, in higher dimensions (higher number of parameters) a heuristic tool can be used to optimize the parameters' values.

4 Heuristics in Finance

4.1 Portfolio Selection

In the simple case of the Markowitz mean-variance (MV) framework with two assets, investors choose their optimal portfolio by maximizing the expected portfolio return ($E = w_1\mu_1 + w_2\mu_2$) given their risk profile, or by minimizing the portfolio risk ($V = w_1^2\sigma_1^2 + w_2^2\sigma_2^2 + 2w_1w_2\sigma_{1,2}$) for a given desirable level of return. The variance of a stock σ_1^2 is the covariance $\sigma_{1,1}$. These models consist of a linear function and a quadratic function of the weighted asset covariances which can be solved by traditional optimization techniques like

Table 1: Optimal Orthogonal Design to minimize Median of Squares Residuals.

CPU	F	CR	n_p	**Best**	**Med**	**Worst**	**Var**	**q5%**	**q90%**	**Freq**
H	0.6597	0.9871	25	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	$4.7501 \cdot 10^{-41}$	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	30
M	0.6616	0.7226	26	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	$5.6897 \cdot 10^{-5}$	$1.7661 \cdot 10^{-12}$	$4.9935 \cdot 10^{-5}$	$5.1175 \cdot 10^{-5}$	1
L	1.3457	0.6587	13	$5.0069 \cdot 10^{-5}$	$5.2804 \cdot 10^{-5}$	$9.9768 \cdot 10^{-5}$	$1.5042 \cdot 10^{-10}$	$5.0075 \cdot 10^{-5}$	$6.9232 \cdot 10^{-5}$	1

Table 2: Optimal Rotatable Design to minimize Median of Squared Residuals.

CPU	F	CR	n_p	**Best**	**Med**	**Worst**	**Var**	**q5%**	**q90%**	**Freq**
H	0.6318	0.6318	81	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	$4.7501 \cdot 10^{-41}$	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	1
M	0.6322	0.9910	26	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	$2.8666 \cdot 10^{-29}$	$4.9935 \cdot 10^{-5}$	$4.9935 \cdot 10^{-5}$	1
L	1.3041	1.0000	11	$5.0079 \cdot 10^{-5}$	$5.1599 \cdot 10^{-5}$	$5.4474 \cdot 10^{-5}$	$9.0143 \cdot 10^{-13}$	$5.0089 \cdot 10^{-5}$	$5.2711 \cdot 10^{-5}$	1

quadratic programming.

However, investors may well seek to optimize utility criteria other than MV (like a Cobb-Douglas objective function for portfolio choice) or other risk criteria (like Value at Risk (VaR), expected shortfall (ES)). Alternative risk criteria are mainly chosen to relax the normality assumption of returns and to accommodate the asymmetry of asset returns. A non-convex, complex, non-differentiable utility function (with linear or non-linear side conditions) will not result in the optimum (efficient) portfolio selection (or in any optimum near) by quadratic programming routines. The optimum might be even difficult to approach for a linear utility function (an absolute or semi-absolute deviation risk function) when constraints are considered (Mansini and Speranza (1999)). Yet, optimization heuristics are flexible enough to optimize portfolios based on complex objective functions without the need for a linear approximation of the objective function.

The seminal work of Dueck and Winker (1992) opened new horizons to portfolio optimization using heuristics, more precisely using TA. In that application, two more complex risk functions other than variance were optimized, being a semi variance function and a weighted mean geometric return function. Considering a slightly more complex objective function, even with linear side conditions on the weights, the problem was not tractable with standard optimization algorithms. TA, with fairly limited computational resources available, could select among 71 bonds of the Federal Republic of Germany profitable portfolios (minimize risk functions given expected return) in reasonable time. The portfolios were 5% to 30% more profitable than the usual practice until that point, meaning human portfolio selection. No transaction costs were considered though.

In the last two decades, heuristic optimization techniques have been applied to higher-dimension problems, due to extensive availability of computational resources. They have been applied to portfolio optimization problems that also allow for side conditions like transaction cost, taxes, cardinality constraints, etc. The following subsections describe the most recent research in this direction.

4.1.1 Transaction Costs and integer Constraints

Apart from the alternative utility criteria that can be pursued by an investor, several other implications and constraints appear in real life in portfolio selection problems. The constraint considered here is transaction costs. In the early literature transaction costs were omitted, mainly for simplicity. However, the selection process with transaction costs can often result in totally different optimal portfolios (Dueck and Winker (1992)). Thus, mod-

ern portfolio selection includes transaction costs in the optimization process (Maringer (2005)).

Transaction costs can appear in the form of minimum amount, proportional amount on trading volume and fixed fees per trade. For an investment in asset a_m $m = 1, \ldots, M$, transaction costs (c) can vary between

c_f	fixed costs
$c_v = c_p \cdot a_m \cdot S_m^{(0)}$	variable costs where $S_m^{(0)}$ is the current stock price
$\max\{c_f, c_v\}$	variable costs with lower (minimum) limit
$c_f + c_v$	fixed plus variable costs .

The optimization algorithm selects the assets a_m and their weights by using (ideally) the total budget amount. Then, transaction costs are part of the budget. Often, a budget amount is uninvested since only integer numbers of assets can be bought. In such a case the excess amount can be deposited.

If one of the above conditions is considered the search space is far from being smooth and convex. Hence, it might be difficult to approach the global optimum or even something near. In that respect, heuristic techniques can be applied. SA, TA and GA are often used in portfolio selection problems. SA and TA are flexible enough to adjust approximately to any restrictions. Their construction (Section 2) allows for faster and more efficient application to discrete financial problems. GA has the additional advantage of simultaneously maintaining a population of candidate portfolios.[9]

Maringer (2005), ch. 3, applied SA to portfolio selection with transaction costs. He used 30 German assets from the DAX index. For the optimal portfolio composition, budget constraints with transaction costs and integer constraints are included and compared. Integer constraints and several types of transaction schemes were used and their relationship to the investment level was tested. The study concluded that the presence of transaction costs, when also non-negativity and integer constraints are included, affected the optimal portfolio structure. Different cost schemes could affect the diversity of the portfolio which could lead to different utilities. The number of different assets invested are reduced with increasing transaction schemes. One exception are investors facing only a fixed cost scheme with an initial investment over 1 million. Nevertheless, SA, Section 2.1.1, could help find a portfolio composition with better diversification (improvement of the utility function) and higher Sharpe Ratio (SR) than the simplified model with no transaction costs.

[9]In high dimensions, the simultaneous maintenance of many candidate solutions can be a computational burden.

In reality not only transaction costs appear, but also other constraints like transaction limits and cardinality constraints. Together these constraints can add to the complexity of the problem and harden the construction of an optimal portfolio. This is especially prominent when we consider small portfolios. The induction of additional constraints into the portfolio selection problem is presented in the following sections.

4.1.2 Minimum Transaction Lots

Minimum transaction lots is a type of integer constraint. It constraints the number of units from a specific stock or bond that should be bought simultaneously. That is, a minimum bundle of units, e.g. 5000 or multiples, from a specific asset should be bought. The constraint is formulated in terms of money as,

$$l_m = N_m \cdot S_m^{(0)}\,, \tag{4}$$

where

l_m	minimum transaction lots in money
N_m	minimum required units of asset m .

In the late 1990s, Mansini and Speranza (1999) showed that for a linear utility function (an absolute or semi-absolute deviation risk function) when linear constraints (minimum transaction lots and proportional transaction costs) are considered the problem is NP hard. They applied variants of evolutionary heuristic algorithms to solve the portfolio selection problem with transaction lots. The algorithm would select the k cheapest assets a_k to construct the portfolio. Different crossover points indicated the securities in and out of the portfolio. The half non-selected assets were replaced by the k low cost assets in the portfolio. Instead of a typical stopping criterion, e.g. reach fixed iteration number, the algorithm was repeated until a number of assets was tested in the portfolio construction. This basic algorithm (and some variations) allowed for a good[10] solution to the portfolio optimization problem in reasonable time. The distinctive advantage of that heuristic was the independence of the computational time from the number of assets.

In another application, Gilli and Këllezi (2002) optimized their portfolio choice using as risk measures VaR and ES (4.1.5). In that application the problem structure allowed for transaction lots constraints. Also, transaction

[10] A good solution is the one that has low distance from the theoretical MV framework)

costs and cardinality constraints were considered. For the optimization, a trajectory heuristic, TA, was applied instead of a population based one. They confirmed the robustness of TA when compared with quadratic programming under MV framework and without any integer constraints. Both approaches resulted in the same optimum MV portfolio. When other non-convex objective functions, such as VaR and ES, together with integer constraints, like transaction lots, were introduced, TA could easily deal with those in reasonable computational time.

By nature, asset weights are continuous variables. So, Krink and Paterlini (2009) propose a Differential Evolution algorithm for Multiobjective Portfolio Optimization (DEMPO) for portfolio selection under the MV framework with risk measures other than variance, i.e. VaR and ES. Transaction lots, weight constraints and limit on the proportion of assets in a particular sector or industrie are considered. The algorithm adopts DE for weight selection. DE could approximate closer and faster the MV frontier compared to quadratic programming (QP) and another GA. Still, the convergence of DE to the exact front shape can be further improved. To overcome local minima, the acceptance criterion of a local search method can be used. It would also be of interest to test its performance in an even more realistic framework with transaction costs and cardinality constraints (some additional constraints are included in Fastrich and Winker (2010)).

4.1.3 Cardinality versus Diversity

In reality, private investors choose a rather small-sized portfolio which is not necessarily diversified internationally (Cuthbertson and Nitzsche (2005), Ch. 18). Small portfolios, when optimally chosen, can diversify the risk away and thus increase investor's utility. Besides, portfolios with fewer stocks are easier to manage and less expensive in terms of total transaction costs than larger more diversified portfolios.

In order to select the optimal size of a portfolio, an additional integer constrain is needed, namely a cardinality constraint. Cardinality constrains the number of different assets (A) in a portfolio, $\#\{A\} = K$. When K varies from $1, \ldots, K$ the constraint is formulated as

$$\sum_{M} a_m \leq k_{max} \,, \tag{5}$$

where

k_{max}	maximum number of different assets in portfolio
A	integer quantities of active positions in portfolio

a_m active position in asset, m = 1,. . . ,M .

The optimization algorithm is slightly altered to find not only which asset weights are optimal but also which assets should be ideally selected from the available ones by optimizing a given objective function. So, different k from M assets $\binom{M}{k}$ and their combinations are selected based on the investors risk profile. This additional integer constraint (especially, together with transaction costs and transaction lots) results in a discrete combinatorial optimization. While traditional selection algorithms result in suboptimal portfolios as the number of assets and constraints increases, heuristics serve as a panacea. Heuristics can capture the discrete selection nature of the problem and consider at the same time the additional constraints. The following literature imposes a cardinality constraint in the portfolio selection problem and improves or optimizes the risk-return SR or the MV portfolio, respectively.

First Chang *et al.* (2000) and later on Maringer (2005) used heuristics to replicate the efficient frontier with integer constraints. Chang *et al.* (2000) used three different heuristics, GA, Tabu Search (TS)[11] and SA to replicate all the Pareto-optimal (non dominated) portfolios of the efficient frontier under cardinality and transaction lots constraints. While a multiple agent heuristic, GA, resulted in fairly better approximations of the theoretical efficient frontier, it needed comparatively more time. In contrary, Maringer (2005) reported the superiority of a single agent heuristic, SA, in selecting a portfolio that maximizes SR.[12]

Alternatively, Maringer (2005) combined a local and a population search method to improve the selection procedure for Markowitz efficient frontier.[13] The performance of the hybrid meta-heuristic was tested against a pure meta-heuristics, like SA and a variant of it. While all three methods share local search characteristics, which help to escape fast inferior solutions, the hybrid meta-heuristic was more reliable – closer to the best-known solution.

[11]TS belongs to local search heuristics. For a comprehensive representation of the algorithm see Glover and Laguna (1997).

[12]Its distinctive selecting performance lies on the random initial selection of assets and the update procedure of the candidate solution. Initially, assets are randomly selected and included in the portfolio. It is flexible to select not only between asset with high risk premium per unit of risk (SR), but also good asset combinations. In the update procedure an asset is added not only if it improves the SR but also if it contributes to the diversification of risk. Ideally, these are assets with negative (or low) correlation.

[13]In the Sharpe framework the heuristic encoded only the asset selection, whereas in the Markowitz efficient frontier identification, the heuristic encoded both weight and model selection. The last approach improved the performance of heuristics.

The results suggest a rather small-sized portfolio, which when optimized by heuristics can replicate relatively accurately the market index. For the few k from M assets and their combinations, the weights are quickly identified by the algorithm and more computational effort is then devoted to update (improving) the candidate selection. Thus, the algorithm results at the end in a portfolio that can be as diversified as the market index.

Further research can be devoted in identifying the factors affecting the optimal K. More precisely, how different risk aversion, utility functions or stock index affect the choice of K. One alternative is to endogenize in the optimization process the choice of K.

4.1.4 Index Tracking

A straight-forward extension of an active portfolio optimization with cardinality considerations is the passive tracking of an index with a limited number of assets. In the former case, investors choose the portfolio that best optimizes their utility function (risk-return equilibrium) under constraints. This decision is called active portfolio management. In the other case, investors are convinced that in the long run the best thing to do is to mimic the market index. In passive selection, investors look for the optimal combination of assets to track the index as close as possible. Typically, the objective function of interest, hereafter called the tracking error, is to minimize the distance between the selected portfolio's daily returns and index returns.

This is a twofold optimization where the optimal asset positions and their weights should be selected. In addition, investors face all the constraints discussed in the previous sections which result in a complex optimization problem. Selecting the optimal asset positions is a discrete combinatorial problem whereas selecting the optimal asset weights is a continuous problem. In that respect, alternative heuristics (suitable for discrete or continuous problems) or a combination of them are applied.

Maringer and Oyewumi (2007) applied DE to track a portfolio with constrained number of assets that mimics best the DJIA64 index. The objective function of interest is to minimize the root-mean-squared deviation of the selected portfolio's daily returns from those of the index. The resulted tracking portfolio outperformed both in- and out-of-sample, in terms of daily asset returns, in many cases the index. For smaller portfolios (especially for cardinality below 30), the positive difference increases for the in-sample training set. Yet, for cardinality below 40 the suggested portfolio is more volatile than the index. However, the higher volatility is outweighed with a higher return-to-risk ratio. Whereas heuristics perform reliably also in this complex problem, there was no clear-cut indication of the optimal number of assets

that best minimize the tracking error.

More insides to the trade-off between the optimal number of asset in a portfolio and the tracking error is given by Krink *et al.* (2009). They also applied DE but with a combinatorial search operator (DECS-IT)[14] to track the perfectly diversified market portfolio by minimizing the volatility between the log-returns of the benchmark (DJIA65 and Nikkei 225 price indices) portfolio and the tracking portfolio in every time period.[15] DECS-IT could potentially solve that selection problem with constraints and provide quality results in reasonable time. The results suggested that by increasing the number of assets in a portfolio the in-sample annualized tracking error volatility reduces up to a certain limit. The same is true for the relationship between cardinality and out-of-sample performance.

There are still however some open questions for constrained index tracking, namely the performance of the tracking portfolio in different window set-ups[16] and the factors affecting the volatility in the results. Finally, one can further investigate the effect of index composition in selecting the maximum number of assets of the tracking portfolio.

4.1.5 Downside Risk Measures

As mentioned in Section 4.1.2, alternative risk measures can be used in place of variance, e.g. VaR and ES. These measures relax the normality assumption of returns imposed by the MV framework. So, instead of minimizing the mean (positive and negative) squared deviations from the expected return, the risk of having negative deviations from the expected portfolio value is minimized. Specifically, the above risk measures minimize either the probability that the expected value of the portfolio falls below a given value (VaR) or the expectancy of such a fall (ES).

Like semi-absolute risk function, VaR and ES cannot always be optimized by traditional linear or quadratic solvers. Especially, in the presence

[14]DE is by nature more suitable for continuous problems. Thus, a combinatorial operator is needed to improve the performance of DE in the discrete combinatorial asset selection. Using this tool the quality of the results and the runtime of the algorithm improved.

[15]Additionally, by building an artificial index they evaluated the performance of DE using the asset weights instead of the returns. They evaluated the asset weights' distance of the benchmark portfolio (Nikkei 225 price index was considered) from those found in the tracking portfolio. The benchmark asset weight are calculated from market values. For this experiment they neglected the cardinality constraint. DECS-IT could very precisely replicate the 225 asset weights. In a comparison with QP (with no cardinality or weight constraints), the two methods could obtain similar results.

[16]Maringer and Oyewumi (2007) experimented with one and two year windows and suggested that shorter in-sample windows together with higher cardinalities are more important for the out-of-sample performance.

of integer constraints, i.e., transaction lots and cardinality constraints, the search space is highly non-smooth and discontinuous. Thus, it is difficult to choose an optimal portfolio that satisfies these constraints in a reasonable time. Gilli and Këllezi (2002), Gilli *et al.* (2006) and Krink and Paterlini (2009) approach the portfolio selection problem under discrete scenarios by optimization heuristics. Apart from one application of DE (Krink and Paterlini (2009)), TA is often the heuristic used for discrete optimization. Winker and Maringer (2004) constructed a hybrid population based algorithm with local search characteristics for the same problem.

In more recent extensions, Gilli and Schumann (2010*c*) presented alternative measures of reward and risk, like the Omega function, and Gilli and Schumann (2009) and Gilli and Schumann (2010*d*) compared the performance of portfolios selected by various risk measures with MV portfolio. TA was used as an optimization algorithm for portfolio construction. A unanimous finding all along was that minimizing risk, as opposed to maximizing reward, often resulted in good prediction of the out-of-sample frontier.

4.2 Robust Model Estimation

Apart from using meta- or hybrid-heuristics for the direct optimization of financial problems, e.g. for asset allocation, much literature is concentrated on the application of heuristics for the indirect optimization of models through parameter estimation. In due course, heuristic algorithms calibrate the parameters of financial and economic nonlinear models that improve the accuracy (precision) and reliability of core financial models.

4.2.1 Nonlinear Model Estimation

Convergence and Estimation of GARCH Models

In reality, sporadic extreme observations can force the distribution of historical stock returns to lean away from the mean. For daily returns the probability of observing this phenomenon is higher than normal, forcing their distribution to exhibit excess kurtosis. Yet, the excess kurtosis decreases with increasing time horizons. As such, daily financial returns exhibit time varying volatility. A *one period* ARCH model depicts this characteristic, $Var(\varepsilon_t) = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2$, where ε_t denotes the volatility of daily returns of an AR(*1*) process (Gouriéroux (1997)). Furthermore, a model conditional also on the previous risk (error) components is a generalized autoregressive conditional heteroskedasticity (GARCH) model.

The parameters of such non-linear models are most often estimated by maximizing the (log)-likelihood function. But, finding the parameters that give an optimal solution is not guaranteed using standard econometric softwares. Sometimes, different softwares result in different optimal solutions (local optima) and the solution is very sensitive to the parameters' starting values. Heuristic optimization overcomes this problem and provides a close to optimum solution for such non-linear models.

Maringer (2005) applied SA to find the parameters of a GARCH model which maximize the (log)-likelihood function. Maringer and Winker (2009) applied TA for the same non-linear problem and simultaneously provided a framework for studying the convergence properties of such estimators based on heuristics. Maringer (2005) used the GARCH(1,1) model to estimate the daily volatility of the German mark/British pound exchange rate. Therefore, the author implemented a SA algorithm like the one presented in Section 2.1.1. In every iteration one parameter was chosen for further alteration. The selected parameter changed by adding a uniform random error. The random error narrowed as the iteration number increased. With $I = 50\ 000$ iterations and a shrinking neighborhood structure, a low deviation was reported between heuristic optimization outcome and true parameters' value. Maringer and Winker (2009) inferred the same superior performance of TA in comparison to standard numerical econometric packages.

A more refined statistical framework is suggested by Maringer and Winker (2009) for analyzing the parameters of heuristics that will further improve the converge of estimators. In due course, they estimate the number of iterations, as a function of sample size, needed by a TA approximation to converge, with a given probability, to the true maximum likelihood estimator of a GARCH model. This is possible, due to the good parameter approximation of search heuristics, reported above. It is evident that the value of the TA maximum likelihood estimator is much closer to the true one compared to what other numerical packages report. This is true in particular for small sample sizes. An interesting extension is the derivation of joint converge properties for higher order GARCH models (or more complex estimators in general). Besides, heuristic techniques can also be considered for selecting the lag order for historic volatility estimation approaches.

Indirect Estimation and Agent Based Models

Modeling the behavior of market participants becomes even more challenging when it cannot be determined by prior information like in a GARCH model. Especially when agents' behavior is either less rational or heterogenous. ABMs are flexible to simulate the behavior of market participants, called agents, and the interaction between them. Simulating the actual be-

havior of agents using ABM can result in a realistic representation of the basic features of daily financial returns, namely excess kurtosis and time varying volatility.

However, the estimation of ABM parameters and the validation of their performance in a realistic market setting with actual data are complex tasks. This is mainly due to the large number of parameters and the non-analytical evaluation of such models. Gilli and Winker (2003) introduced a hybrid TA algorithm to estimate an ABM of foreign exchange markets. The hybrid combined simplex search ideas with TA algorithm. Later on, Winker *et al.* (2007) proposed an evaluation tool for such a model.

Gilli and Winker (2003) used a hybrid TA approach to optimize the ABM parameters. ABM were applied to the model of foreign exchange (DM/US-$) market agents suggested by Kirman (1991) and Kirman (1993). It was tested whether the interaction between two types of agents (fundamentalists and chartists) could realistically explain the excess kurtosis and time varying volatility of returns. For this test, excess kurtosis (k_d) as well as time varying volatility (α_1) of an ARCH(1) model were calculated by simulating the expectations of foreign exchange rate based on agents behavior. The efficiency of agent based modeling was tested with empirical data. A hybrid TA (Section 2.3.2) was applied for optimizing the parameter of ABM. Importantly, in the particular application simplex did not even result in a local optimum due to the high Monte Carlo sampling variance of the initial parameter estimates. To overcome the problem of high Monte Carlo variance on estimating the parameters with relatively low number of repetitions, the authors applied the acceptance criterion of TA. Larger values of threshold sequence correct for the high variance (see also Section 2.1.2). Using these techniques they showed that interactions between market agents could realistically represent foreign exchange market expectations.

Future research can introduce alternative models to ABM, such as MS-AR or SETAR or STAR.[17] In addition, a design approach, like the one discussed in the Appendix, can serve as a benchmark for the estimation of parameters' starting values.

Yield Curve Estimation

Apart from the log-likelihood function of GARCH estimation, there are also other non-linear functions that are used to estimate financial and economic models. Yet, most of these models depend heavily on the parameter

[17]Maringer and Meyer (2007) introduced and compared SA, TA and DE to the estimation of parameters of the STAR model.

estimation value, as ABM do. Likewise, the term structure of interest rates, or as it is widely known the yield curve, is typically estimated (Gimeno and Nave (2009)) using non-linear models, while the input parameters' values are crucial.

The term structure of interest rates refers to market's expectations of interest rates in different time horizons. For example, it depicts the expected interest rate for a loan starting in one, two or ten years from now that lasts for one or several years. Central banks need to calculate the forward-looking interest rates of different maturities on a daily basis. These interest rates serve as an instrument of monetary policy or they are offered by banks to loans of different maturities or they are used in other sectors for asset and derivative pricing. For the calculation of forward rates central banks most frequently use the Nelson-Siegel (Nelson and Siegel (1987)) function or the Nelson-Siegel-Svensson (NSS) function (Svensson (1994)). The model estimates the (spot and) forward interest rates that form the yield curve. For that, the NSS approach models forward interest rates based on short and long term future maturities and the transition between them. This way it can capture the shape of the yield curve.

Financial pricing calls for an accurate estimation of forward interest rates. According to Gimeno and Nave (2009) traditional optimization techniques failed to converge towards a global optimum solution, when estimating the parameters of NSS (a non-linear function). Besides, traditional optimization techniques are sensitive to initial coefficient values, which causes variability in the estimated parameters. The following papers show how this problem can be avoided by using optimization heuristic techniques.

Gimeno and Nave (2009) and Fernández-Rodríguez (2006) suggested application of GA to accurately estimate the coefficients of the Nelson-Siegel function and its extension, the NSS function, while Gilli and Schumann (2010*b*) applied DE. The fitness function corresponds to the weighted sum of squared deviations of the actual (observed) coupon bond prices of different maturities from the ones calculated using the estimated forward interest rates. The error term is a weighted function of the bond's duration period. The authors compared the fitness function value using both traditional optimization techniques, like non-linear least squares optimization, and heuristics for simulated and real world bond prices of different maturities. Overall, GA and DE resulted in lower and less volatile estimation errors.

An interesting extension would be interesting to compare the computational time and efficiency of both GA and DE. An alternative implementation is to use an improved hybrid optimization algorithm which combines a heuristic and a numerical optimization method. Gilli and Schumann (2010*a*) and Gilli and Schumann (2011) discuss the application of DE together with a

direct local search optimizer (Nelder-Mead search) in parameter estimation. They estimated the parameters of the Heston (more easily) and the Bates model, both used in pricing options. Good model fit was reported for the suggested hybrid.

4.2.2 Robust Estimation of Financial Models

Least Median of Squares Estimation

The last core financial model discussed in this section is the CAPM. A substantial amount of research in financial market economics has focused on the robust estimation of the CAPM parameters. Robust estimation techniques have been suggested in the statistical literature as a solution to the low breakdown point of simple estimation approaches, principally, ordinary least squares (OLS). In particular, only a single influential observation can change the parameters of the OLS coefficients significantly (Rousseeuw 1984).

For the linear explanation of the risk premium on individual securities relative to the risk premium on the market portfolio, Chan and Lakonishok (1992) had used least absolute deviations arguing that absolute residuals have smaller influence on LS than squared residuals. Ronchetti and Genton (2008) used shrinkage robust estimators for estimating and predicting the model. Recently, Winker *et al.* (2010) compare OLS and least median of squares (LMS) estimation in the CAPM and the multifactor model of Fama-French factors (Fama and French (1993)). Rousseeuw and Leroy (1987) define the LMS estimator as the solution to the following optimization problem:

$$\min_{\alpha,\beta}(\text{med}(\varepsilon_{i,t}^2))\,, \tag{6}$$

where $\varepsilon_{i,t} = y_{i,t} - \alpha - \beta x_{i,t}$ are the residuals of a factor model. While the LMS estimator has a high breakdown value, its objective function is complex with many local minima. Figure 6 shows the above objective function, evaluated for the CAPM residuals, using the 200 daily stock returns of the IBM stock starting on January 2nd, 1970.

The existence of many local minima is evident, which calls for a heuristic optimization application. Using a rolling window analysis on some stocks from the DJIA index, DE compared to TA allowed a faster and more reliable estimation of β in the factor models. However, the estimates obtained by LMS do not exhibit less variation (from those resulting from OLS) as might have been expected from the outlier related argument. Furthermore, the relative performance of both estimators in a simple one-day-ahead conditional forecasting experiment is mixed. Still, a proof of concept is provided, i.e.,

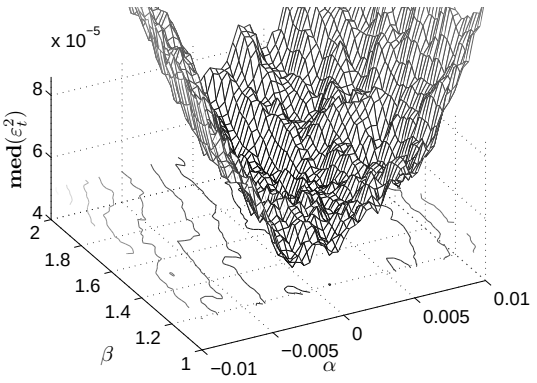


Figure 6: Least median of squared residuals as a function of α and β

LMS estimates obtained by means of heuristic optimization can be used for real life applications.

Some extensions of the paper are straightforward based on the results reported. First, the method should be applied to different data sets, e.g., stock returns from other stock indices or stock markets. Furthermore, it would be of interest to identify in more detail the situations when the estimation and forecast based on LMS outperforms OLS and vice versa.

4.3 Model Selection

This section presents the use of heuristics in selecting a set of independent variables which better explain a given explanatory variable. One can argue that the more factors we include in a model, the better the model fit or its predictive performance will be, as we have more chances to include the important factors. But, the higher the chances are also to include some unimportant factors which worsen the predictive performance of our model. Traditionally, statistical methods are applied for model selection, e.g. discriminant analysis, stepwise analysis etc. However, statistical methods ignore the economic or theoretical relationship between the variables and rely on strong distributional assumption for the factors.

Heuristic optimization methods can be trained, by the objective function, to select some economically meaningful factors. Also, for their application no strong statistical hypotheses are made, e.g. normality assumption for the distribution of factors.

4.3.1 Risk Factor Selection

A different perspective to the asset pricing yields the risk factor selection for the Asset Pricing Theory (APT) model. Like asset selection, risk factor selection is a complex combinatorial problem which in high dimensions is computationally very demanding. More importantly, for problems including more than ten factors, complete enumeration of all possible combinations is difficult to be tested in reasonable time. One alternative is to use heuristic optimization methods for model selection (Winker (2001)).

Maringer (2005), Ch. 7 applied a hybrid meta-heuristic, a MA (Section 2.3.1) to select the appropriate firm and industry specific indices for asset return estimation, at a given point in time, based on the APT model. The hybrid meta-heuristic initially selects a subset of factors and combines the properties of SA and GA (Section 2.1.1 and 2.2.1) to find an optimum subset of MSCI factors that best explain asset returns based on the APT model. That is, get the highest possible explanatory power, adjusted R^2. MA was able to identify the 5 out of 103 MSCI indices that explain a satisfactory percentage of S&P 100 asset price variation.

Future work can concentrate on improving the selection performance of the hybrid meta-heuristic. The use of another trajectory search method, like TA, instead of SA can also be tested. TA has the advantage to construct the candidate solution acceptance criterion based on information from problem's search area. To enhance the economic credibility of the neighborhood construction (factor sets), more weight can be given to neighborhood solutions with economically sound interpretation (and less weight on non-economic sound sets). This will only be possible in the presence of information about the economic factors affecting the returns of a specific asset at a given time. In addition, more emphasis can be given in the model weight selection. A meta-heuristic technique can be used to assign factors' weights based on their economic credibility.

4.3.2 Selection of Bankruptcy Predictors

Factor selection refers also to the selection of predictors so as to evaluate the credibility (solvency) of a debtor (bank client/portfolio). The Basel II Accord on Banking Supervision legislates the framework for banks credit risk assessment (Basel Committee on Banking Supervision 2006). Under this framework banks are required to detain a minimum capital to cover portfolio losses from expected defaults. Expected losses equal the product of the exposure at default (EAD), the loss given default (LGD) and a binary variable that determines the borrower's default grade (D). Based on the

internal rating approach (IRB) banks can estimate internally these three risk components. Extensive literature (see Kumar and Ravi (2007) for a review) is devoted on determining the default grade D. This section addresses the development of a credit model responsible for this binary assignment.

It is evident from the current financial and credit market crisis that credit institutions should develop a more valid credit risk model for the classification of borrowers. Its derivation is subject to a diverse range of risk factors and their weights. Hamerle *et al.* (2003) report a generalized factor model for credit risk composed by firm specific financial data, economic factors and their lags, systematic and idiosyncratic factors. For credit model and weight selection, statistical techniques vary among linear discriminant analysis (Altman (1968), Varetto (1998), etc), conventional stepwise analysis (Butera and Faff (2006)), logit analysis, neural networks (Angelini *et al.* (2008) and Di Tollo and Lyra (2010)) and genetic algorithms (Back *et al.* (1996)).

Back *et al.* (1996) applied discriminant analysis, logit analysis and neural networks for score selection and failure prediction. Each one of this techniques assumes a different interaction among factors. Discriminant analysis assumes a linear relationship, logit analysis adopts a logistic cumulative probability function, whereas neural networks allow a non-linear relationship between explanatory variables. They used stepwise selection to choose the risk factors for the discriminant model and the logistic regression model. Additionally, GA were used in NN for choosing the links. While each technique constructed different models (one, two and three years prior to failure), their predictive ability differ. The GA based model resulted in perceptibly smaller type I and type II errors,[18] at least one and three years prior to failure.

In the same vein Trabelsi and Esseghir (2005), Ahn *et al.* (2006) and Min *et al.* (2006) trained GA that resulted in an optimal selection of bankruptcy predictors. Besides model selection, these papers applied GA to tune the parameters of bankruptcy prediction techniques. Trabelsi and Esseghir (2005) constructed a hybrid algorithm that evolves an appropriate (optimal) artificial neural network structure and its weights. Ahn *et al.* (2006) and Min *et al.* (2006) forecast bankruptcy with SVM. Genetic algorithms select the best set of predictors and estimate the parameters of a kernel function to improve SVM performance using data from Korean companies. The latter paper stressed the potentials of incorporating GA to SVM in future research.

Varetto (1998) applied GA to train neural networks. The meta-heuristic was used not only to model selection, but also to weight estimation. Besides, they used genetic rules for credit clustering. The latter technique is discussed in the following section. NN were compared with a statistical technique,

[18]Smaller type I and type II errors can be interpreted as lower misclassification error.

discriminant analysis, in terms of insolvency diagnosis (failure prediction) for 4738 small Italian companies. To estimate the insolvency criterion, the fitness function, a regression model was fitted using a set of weighted indication variables. GA were used to select the optimal set of financial ratios. For weight estimation, a predefined range of values was assigned to the weights and split in intervals of equal length. GA identified the number of intervals to be added to the lower bound of the range. The above estimation asserts an economic interpretation of the weights in less time than human intuition.

In the context of risk factor selection for bankruptcy prediction other meta-heuristics or hybrid meta-heuristics can also be tested. For example, TA and MA. Alternatively, a different heuristic optimization technique can be applied for weight selection. Given the continuous nature of this problem DE can be suitable. In that respect, a population algorithm for continuous problems can result faster and more efficiently in the optimum weight values. Other heuristics can also be used in factor selection in SVM framework. DE for selecting the parameters of a kernel function and TA for choosing the important factors of the credit risk model. Such an application may further improve the performance of SVM.

What is still an open question in selecting bankruptcy predictors is whether it is necessary (and how) to chose the desired number of risk factors included in a model. The more factors we include in a model, the better will be the model fit, as we have more chances to include the important factors. But, the higher the chances are also to include some unimportant factors ('overfit') which affect (worsen) the predictive performance of our model. Meta-heuristics can be used to endogenously determine the optimal number of factors that can be included in a model (see Lyra *et al.* (2010*b*) and Lyra *et al.* (2010*a*) for an application of TA for determining the optimal number of credit risk grades).

4.4 Financial Clustering

Risk factor selection, credit risk quantification and credit risk assignment are complementary procedures for qualified credit risk management. While Section 4.3 presented the development of a credit model and the quantification of credit risk using meta-heuristics, the clustering of credit risk using heuristic techniques is addressed in this section.

Data clustering has always been on the epicentrum of research work using either statistical or intelligent techniques, namely discriminant analysis and logit or neural networks, support vector machines and case-based reasoning, respectively. Yet, classification of data in several groups with constraint consideration can make the problem NP-hard (Brucker (1978)). Heuristics

are proven to be a reliable tool for financial data classification.

4.4.1 Credit Risk Assignment

Basel II (Basel Committee on Banking Supervision (2006)) requires banks to detain sufficient capital to bare losses that arise not only from expected but from unexpected economic downturns. Provision can cover expected losses, however additional capital or regulatory capital (RC) is required to cover unexpected losses. To calculate RC banks should first assign their client into groups based on their credit risk characteristics, i.e. the probability that borrowers will default the subsequent year (PD).

Then, each group is assigned a 'pooled PD' or a 'mean PD' which distinguishes it from other groups. The deviation from the 'mean PD' determines the required capital detained and consequently the lending rate assigned to each risk group. An important aspect for banks is to avoid misstatement (under/over statement) of capital requirements resulting from misclassification of clients. To achieve this, borrowers should be assigned to a number of homogeneous groups. Therefore, an efficient classification tool is required that minimizes the misclassification error (or the regulatory capital) and satisfies the other constraints imposed by Basel II.

The design of a risk rating system for the assignment of borrowers into homogenous grades is subject to a number of constraints imposed by Basel II. Apart from having at least seven clusters for non-defaulted borrowers (§ 404 of Basel Committee on Banking Supervision (2006)), the EAD in each bucket shall be no higher than 35% of the total borrowers' exposure in a given portfolio (Krink *et al.* (2007)). Thus, we avoid having high concentration of exposure in a given grade. Further, the 'mean PD' for each grade should exceed 0.03%, (§ 285 of Basel Committee on Banking Supervision (2006)). An additional constraint regarding the lower bound on the number of borrowers in each bucket is necessary to ensure that no bucket is empty and that the number of borrowers in each bucket is adequate to statistically evaluate ex-post the precision of the classification system.[19] Introducing the above constraints into the clustering problem increases its complexity. Thus, heuristics have been suggested in the recent literature to tackle financial classification problems.

Krink *et al.* (2007) and Krink *et al.* (2008) contribute to this context by clustering borrowers to a fixed (given) number of buckets and optimizing different objective functions. In either case, DE compared with other heuristics,

[19]Previous literature (Krink *et al.* (2007)) specifies that it should exceed a given percentage, e.g. 1%, of the total number of borrowers in a given portfolio. In Lyra *et al.* (2010*a*), the lower bound is set using statistical benchmarks.

like GA and Particle Swarm Optimization (PSO), showed consistently superior performance in the discrete problem of credit risk rating. In addition, little parameter tuning was required compared with GA and PSO. Lyra *et al.* (2010*b*) extent the previous literature by determining not only the optimal size but also the optimal number of clusters. By doing so, the precision of the classification approach could be statistically evaluated ex-post. They proposed a TA algorithm to exploit the inherent discrete nature of the clustering problem. This algorithm is found to outperform alternative methodologies already proposed in the literature, such as standard k-means and DE. While DE performed reliably for a set of small and medium size Italian companies, TA was more efficient (better grouping solutions) and faster.

The superiority of TA in this problem instance stems from the fact that a fast update of the objective function is feasible and thus computation time is largely independent of the number of clusters. However, for larger data sets and for other more realistic linear and non-linear constraints (such as the modeling of the dependence structure of defaults or the application of heuristics in a dynamic clustering framework) the problem becomes computationally more demanding. It is an open challenge to improve the computational efficiency of heuristics in the computationally demanding problem of financial clustering.

4.4.2 Portfolio Performance Improvement by Clustering

So far, we addressed the direct application of heuristics in asset selection (Section 4.1) when complex utility functions and constraints are considered. The discussed literature reported, in many instances, higher portfolio returns, for a given risk, using heuristics. An indirect application of heuristics in portfolio selection (asset allocation) is found in Zhang and Maringer (2010*a*) and Zhang and Maringer (2010*b*). They suggest pre-clustering assets according to their performance so as to improve the asset allocation procedure.

The clustering of assets before portfolio selection serves as an alterative to the equally weighted asset allocation $1/N$ or the Markowitz allocation based on risk-return equilibrium. In latter case, high dimensional portfolios or possible inaccurate estimations of expected returns or covariance matrices,[20] may result in a sub-optimal allocation of resources. In due course, pre-clustering may help reduce the dimensionality of the problem and thus facilitate the optimization procedure. Pre-clustering involves first clustering assets into a given number of clusters so as to improve the overall portfolio SR. Then,

[20]The equally weighted asset allocation is data independent. So, its performance does not depend on a portfolio's dimensions.

with-in class, traditional asset allocation procedures are applied. Finally, a weighted composition of these clusters constructs the target portfolio.

Zhang and Maringer (2010*a*) used and compared two different heuristics, DE and GA, to clustering financial data (FTSE assets). Compared with traditional non-clustering allocation approaches, pre-clustering of assets improved the SR distribution of clustered portfolios, according to both in-sample and out-of-sample simulation studies. Heuristics are proven to be a reliable tool for clustering financial data, as indicated by the improved SR distributions. Nevertheless, GA resulted in better SRs than DE in all restarts. Besides, GA outperformed DE. GA resulted in higher and more stable fitness values. As indicated from the results, GA performed well in the discrete problem of clustering, mainly because of their discrete nature (construction), Section 2.2.1.[21]

In the above application a predefined number of clusters was used to partition assets. A possible extension in this direction is to determine the optimal number of clusters based on the sample size and the index composition used. Further, the potential performance of the suggested clustering design in the presence of highly volatile asset returns should further be investigated.

4.4.3 Identification of Mutual Funds Style

Clustering data, especially data with complex schemes, can help extract all the available information. Besides, as discussed in Section 4.4.2, clustering avoids, to a high extent, the 'curse of dimensionality' problem often faced in large portfolio management. The aim of clustering is the partition of data in order to minimize the within (homogeneity) and maximize he between clusters variance (heterogeneity). Brucker (1978) has shown that for a number of clusters higher than three the problem is highly complex. So, the recent literature has developed a number of new classification approaches which combine classical statistical methodologies, e.g. principal component analysis (PCA), and intelligent techniques, e.g. DE, NN to tackle the problem.

Pattarin *et al.* (2004) proposed a combination of PCA and GA to group the historical returns of the Italian mutual funds and identify their style. First, PCA selected the order of the process (the lag order). Second, the evolutionary algorithm (GA for medoids evolution - GAME) classified the

[21] To adjust the continuous optimization technique, DE, to a discrete problem, a random noise (z) term is added to the scaled difference of two randomly selected vectors, during the differential mutation process, $P^{(v)}_{.,i} = P^{(0)}_{.,r_1} + (F + z_{.,1}) \times (P^{(0)}_{.,r_2} - P^{(0)}_{.,r_3} + z_{.,2})$. The random noise is added with a given probability and it follows a normal distribution. The candidate solutions are rounded up to the nearest integer.

historical returns in homogenous groups.[22] The new classification method based on GA was successful in identifying the mutual fund style with less information than the classification method of the Italian mutual fund managers association ('Assogestioni').

Future applications can consider the use of other heuristics in financial data clustering. Das and Sil (2010) recommended a revised DE algorithm in clustering the pixels of an image. They used a kernel induced similarity function to measure the distance from the cluster center. The advantage of this measure, in relation to the Euclidean distance, is that it allows non-linear classification of data. That implementation improved the accuracy, the speed and the robustness of DE compared to GA. A similar implementation could be used in financial clustering problems.

5 Conclusions

This survey gives an overview of some modern financial optimization problems. Often the additional constraints and the high-dimensionality of these problems, make them complex and thus difficult to solve by standard optimization algorithms. The paper suggests heuristic optimization techniques as an alternative. In the presented literature heuristics provide reliable approximations and are successfully applied to optimize financial problems with great potentials. The paper shows some promising fields of application of heuristic optimization techniques and some possible extensions in that direction.

Optimization heuristic techniques are flexible to tackle a broad variety of complex optimization problems. This is due to the stochastic elements introduced during the optimization process. Explicitly, the optimization process involves stochastic initialization, intermediate stochastic selection and acceptance of the candidate solution. Nonetheless, the outcomes (optimization results) should be carefully interpreted, while numerous repetitions of the algorithm are recommended.

The choice of the appropriate heuristic technique is of particular importance for its efficiency. Heuristic methods differ in their acceptance criteria, the actual way of creating new solution and the number of solutions they maintain and generate in every iteration. In the literature discussed in this survey, DE has shown remarkable performance in continuous numerical problems, e.g. parameter estimation, when compared with other heuristics, i.e.

[22] GAME offers the flexibility of either using heuristics or statistical approaches in choosing the number of lags included in the model. In that application a PCA was used for the selection.

TA. Even if DE is specialized on continuous numerical problems, it has already shown better performance than GA and PSO in tackling the credit risk bucketing. Nonetheless, local search heuristics, particularly TA, is faster and more robust to deal with the discrete problem of credit risk assignment.

It seems that there is no clear-cut recipe when to use which type of heuristics. It may depend on the nature of the problem (discrete or continuous search space), the computational resources and the application. Future research might analyze further the criteria for optimal algorithm selection.

An additional aspect affecting the efficiency of heuristics is the initial parameter settings. They are vital, for they provide a trade-off between computational time and solution quality. An open challenge in the application of heuristics is to set a more refined statistical framework for estimating the heuristic parameters, which will further improve the converge of the estimation results. Maringer and Winker (2009) estimated the number of iterations, as a function of sample size, needed by a TA to precisely converge, with a given probability, to the true maximum likelihood estimator of a GARCH model. An interesting extension in this direction is the derivation of joint convergence properties for more complex estimators.

To sum up, heuristic methods have been applied to many promising fields of commercial applications, like portfolio selection with real world constraints, robust model estimation and selection and financial clustering with good potential. They are flexible enough to account for different constraints and provide reliable results where traditional optimization methods fail to work. Still, more research should be devoted to identifying the problem framework (what is a 'good problem for heuristics') in which heuristics clearly outperform traditional optimization methods or in which some heuristics are preferred instead of other heuristic methods.

Appendix

Central Composite Design

This section provides a detailed explanation and a numerical representation of the CCD experiments. First, for the factorial design we assign to the parameters two discrete 'levels'($F = 0.7, 0.9$, $CR = 0.7, 0.9$ and $n_p = 20, 50$). This range constitutes the experimental region and is carefully specified around an initial value (center point) for each parameter. The initial levels depend upon a priori knowledge about the best parameter combination that optimize

a given objective function.[23] Three different levels of computational resources are considered, with $n_G \times n_p$= 500, 2500 and 5000 meaning, low, medium and high complexity, respectively. Since the product of $n_p \times n_G$ should be constant, an adjustment is made on n_G in each alteration. The aim is to find the combination of parameters that optimize an objective function value using these fixed computational times.

Second, we conduct an additional experiment using the center points. The center points equal the median (center) of the 'levels' used in the above factorial design ($F = 0.8$, $CR = 0.8$ and $n_p = 35$). This experiment is repeated 4 times to measure the magnitude of random error (σ^2), resulting from different values of the response variable for identical settings (Fang *et al.* (2006)). Table 3 illustrates the experimental region where the three factor levels, low, center and high, are coded -1, 0 and 1, respectively.[24]

Table 3: Experimental Design Coding.

Code	F	CR	n_p
-1	0.7	0.7	20
0	0.8	0.8	35
1	0.9	0.9	50

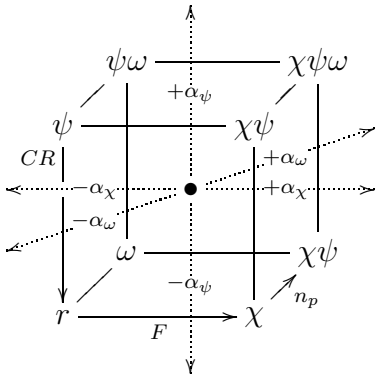
Finally, two more extreme levels are considered for each input variable which extent the experimental region by α times the original range, defined above e.g. $\pm\alpha\cdot\Delta$F = $\pm\alpha\cdot$0.1. A proportion of α determines the distance of the additional extreme levels from the center. The value α is calculated in this experiment based on orthogonal and rotatable designs (Box and Draper (1987) and Barker (1985)). Each design suggests a different distance from the center points. An orthogonal design is more concentrated around the center point while a rotatable design covers a wider surface area. When little is known about the search domain a rotatable design might be more helpful in exploring it, since it allows for a higher dispersion around the origin. However, one should be careful in choosing the distribution of experimental points around the origin, since unreasonably low or high values might result

[23]Practical advice for optimizing objective functions with DE is given on www.icsi.berkeley.edu/∼storn/.

[24]To code the factor levels the following rule is applied, e.g. for the high level of F:

$$\frac{F_{high} - F_{center}}{(F_{high} - F_{low})/2} = \frac{0.9 - 0.8}{(0.9 - 0.7)/2} = 1 \tag{7}$$

Figure 7: CCD Graph for three factors.



in high uncertainty of the response output.[25]

These additional levels are coded with α and are sequentially placed at $\pm\alpha$. When α is positive, the equivalent proportion of α is +0.1414, +0.1414 and +7 value for F, CR and n_p, respectively. The equivalent α is added to the center point of each input parameter (e.g. for F 0.8 + α). When α is negative, a proportion of it is deducted from the center point (e.g. for F 0.8 - α).[26] Only one parameter is altered every time, while all other parameters take the center value. In a similar way as above, the Greek letters indicate which factor deviates from the center point.[27]

Table 4 demonstrates the experimental design with all 18 level combinations considered in the exercise. The graphical representation of these combinations is shown in Figure 7. There are only 15 possible factor level combinations and the experiment using factors' center levels ($F = 0.8$, $CR = 0.8$ and $n_p = 35$) is repeated four times. Also, Figures 8 and 9 illustrate all factor level-combinations used to form the experimental region for orthogonal and rotatable designs, respectively.

All 18 set-ups of the experimental design are used to estimate Equation 2. Then, a second order polynomial is fitted and the optimal combination of DE factors is determined. Finally, Equation 2 for CAMP is re-estimated

[25]While experimental methodologies should be applied when there is some knowledge about the levels of the input variables, Lin *et al.* (2010) suggest that uniform design might be even better than orthogonal and rotatable ones in the absence of design knowledge.

[26]The decimal positions are omitted for n_p value since they do not improve or destroy the integrity (mathematical results) of the design.

[27]For the explicit calculation of α's based on orthogonal and rotatable design see Hill and Lewicki (2006).

Table 4: Experimental Design (Simulation Setup).

Design	Combination	Setup	F	CR	n_p
Factorial	r	1	-1	-1	-1
	χ	2	1	-1	-1
	ψ	3	-1	1	-1
	$\chi\psi$	4	1	1	-1
	ω	5	-1	-1	1
	$\chi\omega$	6	1	-1	1
	$\psi\omega$	7	-1	1	1
	$\chi\psi\omega$	8	1	1	1
Center	node	9	0	0	0
		10	0	0	0
		11	0	0	0
		12	0	0	0
Orthogonal	$+\alpha_\chi$	13	α	0	0
	$-\alpha_\chi$	14	$-\alpha$	0	0
	$+\alpha_\psi$	15	0	α	0
	$-\alpha_\psi$	16	0	$-\alpha$	0
	$+\alpha_\omega$	17	0	0	α
	$-\alpha_\omega$	18	0	0	$-\alpha$

based on the optimal factor values and its performance is tested against the LMS residuals reported in Winker *et al.* (2010).

Figure 8: CCD with orthogonal design

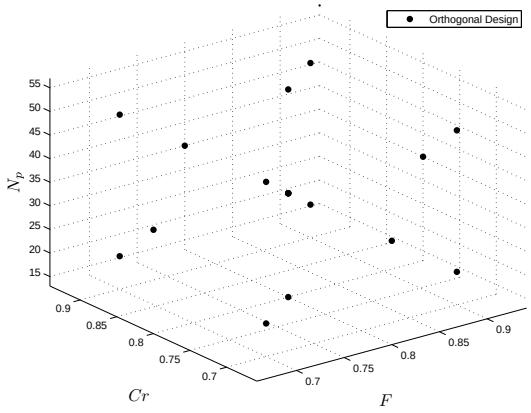
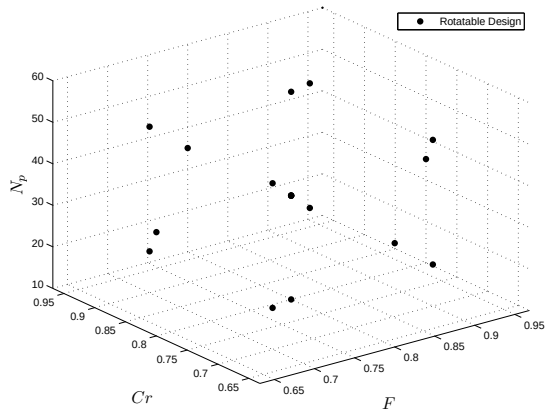


Figure 9: CCD with rotatable design



References

Ahn, H., K. Lee and K. J. Kim (2006). Global optimization of support vector machines using genetic algorithms for bankruptcy prediction. *Lecture Notes in Computer Science* **4234**, 420–429.

Altman, E. (1968). Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *The Journal of Finance* **23**, 589–601.

Angelini, E., G. di Tollo and A. Roli (2008). A neural network approach for credit risk evaluation. *The Quarterly Review of Economics and Finance* **48**, 733–755.

Back, B., T. Laitinen, K. Sere and M. V. Wezel (1996). Choosing bankruptcy predictors using discriminant analysis, logit analysis, and genetic algorithms. Technical Report 40. Computational Intelligence for Business. Turku Center for Computer Science.

Barker, T. B. (1985). *Quality by Experimental Design*. Marcel Dekker, New York.

Basel Committee on Banking Supervision (2006). International convergence of capital measurement and capital standards a revised framework comprehensive version. Technical report. Bank for International Settlements.

Box, G. E. P. and N. R. Draper (1987). *Empirical Model Building and Response Surface*. Wiley, New York.

Brucker, P. (1978). On the complexity of clustering problems. In: *Optimisation and Operations Research* (M. Beckmenn and H.P. Kunzi, Eds.). Vol. 157 of *Lecture Notes in Economics and Mathematical Systems*. pp. 45–54. Springer. Berlin.

Butera, G. and R. Faff (2006). An integrated multi-model credit rating system for private firms. *Review of Quantitative Finance and Accounting* **27**, 311–340.

Chan, L. K. C. and J. Lakonishok (1992). Robust measurement of beta risk. *Journal of Financial and Quantitative Analysis* **27**, 265–282.

Chang, T.-J., N. Meade, J.E. Beasley and Y.M. Sharaiha (2000). Heuristics for cardinality constrained portfolio optimisation. *Computers and Operations Research* **27**(13), 1271–1302.

Cuthbertson, K. and D. Nitzsche (2005). *Quantitative Financial Economics*. John Wiley & Sons Ltd. Chichester.

Das, S. and S. Sil (2010). Kernel-induced fuzzy clustering of image pixels with an improved differential evolution algorithm. *Information Sciences* **180**, 1237–1256.

Di Tollo, G. and M. Lyra (2010). Elman nets for credit risk assessment. In: *Decision Theory and Choices: a Complexity Approach* (M. Faggini and P. C. Vinci, Eds.). pp. 147–170. Springer. Milan.

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

Dueck, G. and T. Scheuer (1990). Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *J. Comp. Phys* **90**, 161–175.

Fama, E. F. and K. R. French (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics* **33**, 3–56.

Fang, K. T., R. Lin and A. Sudjianto (2006). *Design and Modeling for Computer Experiments*. Chapman and Hall/CRC.

Fastrich, B. and P. Winker (2010). Robust portfolio optimization with a hybrid heuristic algorithm. Technical Report 41. COMISEF Working Papers Series.

Fernández-Rodríguez, F. (2006). Interest rate term structure modeling using free-knot splines. *The Journal of Business* **79**, 3083–3099.

Gilli, M. and E. Këllezi (2002). Portfolio optimization with VaR and expected shortfall. *in: EJ. Kontoghiorghes, B. Rustem, S. Siokos (eds), Computational Methods in Decision Making, Economics and Finance.* pp. 167–183.

Gilli, M. and E. Schumann (2009). An empirical analysis of alternative portfolio selection criteria. Technical Report No. 09-06. Swiss Finance Institute Research Paper.

Gilli, M. and E. Schumann (2010*a*). Calibrating the Heston model with differential evolution. In: *Lecture Notes in Computer Science* (number 6025 EvoApplications 2010, Part II, Ed.). pp. 242–250. Springer.

Gilli, M. and E. Schumann (2010*b*). Calibrating the Nelson Siegel Svensson model. Technical Report 31. COMISEF Working Papers Series.

Gilli, M. and E. Schumann (2010*c*). Distributed optimization of a portfolio's Omega. *Parallel Computing* **36**(7), 381–389.

Gilli, M. and E. Schumann (2010*d*). Risk–reward optimization for long-run investors: an empirical analysis. *European Actuarial Journal (forthcoming)*.

Gilli, M. and E. Schumann (2011). Calibrating option pricing models with heuristics. In: *Natural Computing in Computational Finance* (B. Anthony and M. O'Neill, Eds.). Vol. 4. Springer (forthcoming).

Gilli, M. and P. Winker (2003). A global optimization heuristic for estimating agent based models. *Computational Statistics and Data Analysis* **42**, 299–312.

Gilli, M. and P. Winker (2009). Heuristic optimization methods in econometrics. In: *Handbook on Computational Econometrics* (E. Kontoghiorghes et al., Ed.). pp. 81–119. Elsevier. Amsterdam.

Gilli, M., D. Maringer and P. Winker (2008). Applications of heuristics in finance. In: *Handbook On Information Technology in Finance* (D. Seese, C. Weinhardt and F. Schlottman, Eds.). pp. 635–653. International Handbooks on Information Systems. Springer, Germany.

Gilli, M., E. Këllezi and H. Hysi (2006). A data-driven optimization heuristic for downside risk minimization. *Journal of Risk* **8**(3), 1–18.

Gimeno, R. and J. M. Nave (2009). A genetic algorithm of the term structure of interest rates. *Computational Statistics and Data Analysis* **53**, 2236–2250.

Glover, F. and M. Laguna (1997). *Tabu Search*. Kluwer Academic Publishers. Boston, MA.

Gouriéroux, C. (1997). *ARCH Models and Financial Applications*. Springer. New York.

Hamerle, A., T. Liebig and D. Rösch (2003). Credit risk factor modeling and the Basel II IRB approach. Technical Report 2. Banking and Financial Supervision. Deutsche Bundesbank.

Hill, T. and P. Lewicki (2006). *Statistics: Methods and Applications. A Comprehensive Reference for Science, Industry and Data Mining*. StatSoft, Inc.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Harbor.

Hoos, H. H. and T. Stützle (2005). *Stochastic Local Search: Foundations and Applications*. Elsevier Inc.

Jacobson, S. H., S. N. Hall and L. A. McLay (2006). Visiting near-optimal solutions using local search algorithms. *In COMPSTAT 2006, Proceedings in Computational Statistics (Alferdo Rizzi and Maurizio Vichi, Ed.)* pp. 471–481.

Kirman, A. (1991). Epidemics of opinion and speculative bubbles in financial markets. In: *Money and Financial Markets* (M. Taylor, Ed.). pp. 354–368. Macmillan. New York.

Kirman, A. (1993). Ants, rationality and recruitment. *Quarterly Journal of Economics* **108**, 137–156.

Krink, T. and S. Paterlini (2009). Multiobjective optimization using differential evolution for real-world portfolio optimization. *Computational Management Science (forthcoming)*.

Krink, T., S. Mittnik and S. Paterlini (2009). Differential evolution and combinatorial search for constrained index tracking. *Annals of Operations Research* **152**(1), 153–176.

Krink, T., S. Paterlini and A. Resti (2008). The optimal structure of pd buckets. *Journal of Banking and Finance* **32**(10), 2275–2286.

Krink, T., S. Paterlini and A. Restic (2007). Using differential evolution to improve the accuracy of bank rating systems. *Computational Statistics & Data Analysis* **52**(1), 68–87.

Kumar, P. R. and V. Ravi (2007). Bankruptcy prediction in banks and firms via statistical and intelligent techniques - a review. *European Journal of Operational Research* **180**, 1–28.

Lagarias, J. C., J. A. Reeds, M. H. Wright and P. E. Wright (1999). Convergence behavior of the Nelder-Mead simplex algorithm in low dimensions. *SIAM J. Optimization* **9**, 112–147.

Lin, D. K.J., C. Sharpe and P. Winker (2010). Optimized U-type designs on flexible regions. *Computational Statistics & Data Analysis* **54**(6), 1505–1515.

Lyra, M., A. Onwunta and P. Winker (2010*a*). Threshold accepting for credit risk assessment and validation. Technical Report 39. COMISEF Working Paper Series.

Lyra, M., J. Paha, S. Paterlini and P. Winker (2010*b*). Optimization heuristics for determining internal rating grading scales. *Computational Statistics & Data Analysis* **54**(11), 2693–2706.

Mansini, R. and M. G. Speranza (1999). Heuristic algorithms for the portfolio selection problem with minimum transaction lots. *European Journal of Operational Research* **114**, 219–233.

Maringer, D. (2005). *Portfolio Management with Heuristic Optimization*. Springer. Dordrecht.

Maringer, D. and M. Meyer (2007). Smooth transition autoregressive models - new approaches to the model selection problem. *Studies in Nonlinear Dynamics and Econometrics (forthcoming)*.

Maringer, D. and O. Oyewumi (2007). Index tracking with constrained portfolios. *Intelligent Systems in Accounting, Finance and Management* **15**, 57–71.

Maringer, D. and P. Winker (2009). The convergence of estimators based on heuristics: Theory and application to a GARCH model. *Computational Statistics* **24**, 533–550.

Min, S. H., J. Lee and I. Han (2006). Hybrid genetic algorithms and support vector machines for bankruptcy prediction. *Expert Systems with Applications* **31**, 652–660.

Moscato, P. (1989). On evolution, search, optimization, genetic algorithms and martial arts – towards memetic algorithms. Technical Report 790. CalTech California Institute of Technology.

Moscato, P. and J.F. Fontanari (1990). Stochastic versus deterministic update in simulated annealing. *Physics Letters A* **146**(4), 204–208.

Nelson, C. R. and A. F. Siegel (1987). Parsimonious modeling of yield curves. *Journal of Business* **60**(4), 473–489.

Pattarin, F., S. Paterlini and T. Minerva (2004). Clustering financial time series: an application to mutual funds style analysis. *Computational Statistics and Data Analysis* **47**, 353–372.

Price, K. V., R. M. Storn and J. A. Lampinen (2005). *Differential Evolution: A Practical Approach to Global Optimization*. Springer, Germany.

Ronchetti, E. and M. Genton (2008). Robust prediction of beta. In: *Computational Methods in Financial Engineering* (E. Kontoghiorghes, B. Rustem and P. Winker, Eds.). pp. 147–161. Springer. Berlin.

Rousseeuw, P. J. (1984). Least median of squares regression. *Journal of the American Statistical Association* **79**, 871–880.

Rousseeuw, P. J. and A. M. Leroy (1987). *Robust Regression and Outlier Detection*. John Wiley & Sons, New Jersey.

Savin, I. and P. Winker (2010). Heuristic optimization methods for dynamic panel data model selection. Application on the Russian innovative performance. Technical Report 27. COMISEF Working Papers Series.

Storn, R. and K. Price (1997). Differential evolution: a simple and efficient adaptive scheme for global optimization over continuous spaces. *J. Global Optimization* **11**, 341–359.

Svensson, L. E. O. (1994). Estimating and interpreting forward interest rates: Sweden 1992-1994. Technical Report 4871. National Bureau of Economic Research.

Talbi, E. G. (2002). A taxonomy of hybrid metaheuristics. *Journal of Heuristics* **8**, 541–564.

Trabelsi, A. and M. A. Esseghir (2005). New evolutionary bankruptcy forecasting model based on genetic algorithms and neural networks. *in: Proceedings of the eleventh IEEE conference on tools with artificial intelligence (ICTAI-2005).*

Varetto, F. (1998). Genetic algorithms applications in the analysis of insolvency risk. *Journal of Banking and Finance* **22**, 1421–1439.

Winker, P. (2001). *Optimization Heuristics in Econometrics: Applications of Threshold Accepting*. Wiley, New York.

Winker, P. and D. Maringer (2004). The hidden risks of optimizing bond portfolios under VaR. *Deutsche Bank Research Notes 13, Frankfurt a.M.*

Winker, P. and D. Maringer (2007). The threshold accepting optimisation algorithm in economics and statistics. In: *Optimisation, Econometric and Financial Analysis* (Erricos J. Kontoghiorghes and C. Gatu, Eds.). Vol. 9 of *Advances in Computational Management Science*. pp. 107–125. Springer.

Winker, P. and K.-T. Fang (1997). Application of threshold accepting to the evaluation of the discrepancy of a set of points. *SIAM Journal on Numerical Analysis* **34**(5), 2028–2042.

Winker, P., M. Gilli and V. Jeleskovic (2007). An objective function for simulation based inference on exchange rate data. *Journal of Economic Interaction and Coordination* **2**, 125–145.

Winker, P., M. Lyra and C. Sharpe (2010). Least median of squares estimation by optimization heuristics with an application to the CAPM and multi factor models. *Computational Management Science (forthcoming)*.

Zhang, J. and D. Maringer (2010*a*). Asset allocation under hierarchical clustering. Technical Report 36. COMISEF Working Papers Series.

Zhang, J. and D. Maringer (2010*b*). A clustering application in portfolio management. In: *Electronic Engineering and Computing Technology* (S. Ao and L. Gelman, Eds.). Vol. 60 of *Lecture Notes in Electronical Engineering*. pp. 309–321. Springer.