A Heuristic Framework for the Bi-Objective Enhanced Index Tracking Problem

C. Filippi *G. Guastaroba* *M.G. Speranza*
Department of Economics and Management, C.da S.Chiara 50
University of Brescia, Brescia, Italy
{carlo.filippi, gianfranco.guastaroba, grazia.speranza}@unibs.it

November 30, 2015

Abstract

The index tracking problem is the problem of determining a portfolio of assets whose performance replicates, as closely as possible, that of a financial market index chosen as benchmark. In the enhanced index tracking problem the portfolio is expected to outperform the benchmark with minimal additional risk. In this paper, we study the bi-objective enhanced index tracking problem where two competing objectives, i.e., the expected excess return of the portfolio over the benchmark and the tracking error, are taken into consideration. A bi-objective Mixed Integer Linear Programming formulation for the problem is proposed. Computational results on a set of benchmark instances are given, along with a detailed out-of-sample analysis of the performance of the optimal portfolios selected by the proposed model. Then, a heuristic procedure is designed to build an approximation of the set of Pareto optimal solutions. We test the proposed procedure on a reference set of Pareto optimal solutions. Computational results show that the procedure is significantly faster than the exact computation and provides an extremely accurate approximation.

Keywords: Enhanced Index Tracking, Mixed Integer Linear Programming, Bi-Objective Optimization, Bi-Objective Heuristic Framework.

1 Introduction

In modern economies, market indices serve as indicators of the behavior of financial markets but also as benchmarks in financial trading. Indeed, indices are used, nowadays, as standard benchmarks for evaluating the performance of fund managers and the number of funds managed by index-based investment strategies has grown significantly in recent years. Index-based fund management strategies have been traditionally classified into two broad classes: passive and active management. With *passive management* a fund manager is expected to create a portfolio of assets whose performance replicates, as closely as possible, that of a financial market index (i.e., the *benchmark*) provided by statistical bureaus like Standard & Poor's. This strategy is commonly referred to as *index tracking*. The goal is to minimize a *tracking error*, that measures the deviations of the performance of the portfolio from the benchmark. With *active management* a fund manager has the goal of yielding a higher return than the benchmark. A common active management strategy consists in underweighting or overweighting stocks compared with the benchmark based on the manager beliefs (e.g., see Li *et al.* [38]). Therefore, fund managers adopting an active management strategy frequently rebalance their portfolio composition in an attempt to beat the benchmark [37], but incurring expensive transaction and management costs. Additionally, several researchers highlighted that the majority of actively managed funds do not outperform the benchmark in the long term (see, among others, Gruber [23]). Because of these issues, active management strategies are often adopted over a limited portion of the fund investment, whereas the remaining part is managed by passive management strategies [38].

Enhanced index tracking, also referred to as enhanced indexation, is a recent management strategy based upon the idea of combining passive and active management in order to capture the strengths of both approaches (see [34]). The goal of enhanced index tracking is to slightly outperform the benchmark with minimal additional risk. On one hand, enhanced index tracking resembles passive management because the fund manager is not allowed to build portfolios that significantly deviate from the benchmark. On the other hand, it resembles active management since the fund manager can engage limited additional risk (i.e., larger deviations from the benchmark) in order to yield a larger return than the benchmark.

Several studies show that the amount invested in index funds steadily increased in the last two decades. Frino *et al.* [20] claim that the total assets benchmarked to the S&P500 Index exceeded USD 1 trillion. Koshizuka *et al.* [35] mention that more than half of the total funds traded in the Tokyo stock exchange are invested into index funds and that a significant amount of funds is managed by enhanced index tracking approaches. Jorion [34] points out that surveys conducted among fund managers of US institutional tax-exempt assets indicate that, over the period 1994 to 2000, enhanced index funds have grown from USD 33 to USD 365 billion, a ten-fold factor, accounting for 24% of total index assets for this group. Jorion [34] also reports that, in the same period, passively managed funds have grown slower than enhanced index funds. The author relates the growing popularity of enhanced index funds to two factors: firstly, they developed in response to the poor performance of actively managed funds and, secondly, they allow investors to maintain their desired risk profile.

Several market indices are available and their composition is usually publicly accessible. They mainly differ in terms of geographical region, type of financial market, section of the financial market, and length of the membership list, which may range from some tens to 2-3 thousands (e.g., see [19, 21, 30, 31, 32]).

The seminal work by Markowitz [41] has provided the fundamental basis for a large part of modern portfolio theory. Markowitz was the first to formalize the portfolio optimization problem as a risk-return bi-criteria optimization problem where the expected return is maximized and the variance as a risk measure is minimized. Since then, several alternative portfolio optimization models have been proposed where a different risk measure than the variance is adopted. Nevertheless, the Markowitz model is still widely used among both academics and practitioners [39]. The usual approach to tackle the aforementioned risk-return bi-objective models is to cast them into a mono-objective formulation, i.e., either minimize the risk for a given level of expected return (the most common choice) or maximize the expected return for a given level of risk. In recent years a growing body of work dealing with multi-objective approaches to solve financial problems appeared in the literature. Among them, multi-objective evolutionary algorithms play a relevant role (see Metaxiotis and Liagkouras [44] for a recent overview of these algorithms applied to portfolio optimization problems).

Several mathematical formulations have been proposed for the index tracking problem. Most of them are mono-objective as the only goal in index tracking is to minimize the tracking error. On the contrary, the intrinsic nature of the enhanced index tracking problem is bi-objective: to maximize a performance measure of the portfolio while minimizing the tracking error. Nevertheless, also the enhanced index tracking problem is commonly treated as a mono-objective problem. Some of these optimization models include several real-life features that are frequently encountered in financial markets and in the portfolio optimization literature. Indeed, in real-life situations, it is necessary to consider some trading requirements and other aspects of the portfolio composition. For instance, it is important (i) to consider transaction costs as they impact on the return of the investment (e.g., see [9] and [22]), (ii) to restrict the total number of stocks composing the portfolio (e.g., see [40] and [48]), and (iii) to limit the amount of capital invested in a stock in order to avoid very small holdings invested in some assets, or very large holdings invested in one or very few stocks (e.g., see [48]).

Guastaroba and Speranza [26] introduce Mixed Integer Linear Programming (MILP) formulations for the index tracking and the (mono-objective) enhanced index tracking problems and apply a heuristic algorithm, called Kernel Search, to solve the index tracking problem. These optimization models include the real-life features mentioned above, and can deal with the case the investor holds a current portfolio and wants to rebalance its composition as well as with the case the investor wants to build a portfolio from scratch. The authors also present an out-of-sample comparison between the portfolios selected adopting the proposed index tracking formulation and those found using the approach described in Beasley *et al.* [8]. Note that rebalancing portfolio optimization models have appeared in recent literature on portfolio optimization (see [24], [26], [46], [49], [57] and references therein). Note also that introducing a cardinality constraint increases dramatically the computational burden (see, among others, Maringer and Oyewumi [40] who study the impact of such a constraint in an index tracking problem).

The Kernel Search is a heuristic framework with a general and flexible structure applicable to any MILP problem with binary variables (see [5]). The heuristic is based on the idea of identifying subsets of the decision variables and then solving MILP subproblems, each subproblem restricted to a subset of variables. It relies on the high performance of the general-purpose solvers available for the solution of Linear Programming (LP) and MILP problems, and requires a small implementation effort. The Kernel Search was introduced by Angelelli *et al.* in [5] and [6] for the solution of the multi-dimensional knapsack and a portfolio optimization problem, respectively. Enhancements overcoming some limitations affecting the original framework were introduced in Guastaroba and Speranza [26], [27] and [28] where the Kernel Search was successfully applied to the index tracking, the multi-source capacitated facility location and the single-source capacitated facility location problems, respectively. In all the aforementioned contributions, the Kernel Search was used to solve mono-objective optimization problems.

Contributions of the Paper. The contribution of this paper is twofold. Firstly, we propose a mathematical formulation for the bi-objective enhanced index tracking problem including several real-life features. We present some valid inequalities and variable reduction procedures in order to strengthen the mathematical formulation. Secondly, we extend the Kernel Search algorithm designed in [26] for the index tracking problem to the solution of the bi-objective enhanced index tracking problem. More precisely, we embed the Kernel Search framework into a simple strategy, derived from the classic ϵ-constraint method, in order to approximate the Pareto optimal solutions. This is the first application of the Kernel Search method to a bi-objective problem. The extension of the Kernel Search algorithm that we propose is not a straightforward implementation of the original framework, but is rather designed to take advantage of the bi-objective structure of the problem to identify the subsets of the decision variables.

Structure of the Paper. The remainder of the paper is organized as follows. In Section 2 we discuss the literature related to enhanced indexation, multi-objective portfolio optimization and multi-objective solution methods for MILP problems. In Section 3 we introduce a bi-objective formulation for the enhanced index tracking problem. In Section 4 we describe the bi-objective Kernel Search framework, whereas in Section 5 we present extensive computational experiments. Finally, in Section 6 we draw some concluding remarks.

2 Literature Review

In this section, we first survey the literature on the enhanced index tracking problem and on multi-objective approaches applied to portfolio optimization problems. Then, we provide an overview of the foremost contributions on multi-objective approaches for MILP problems.

2.1 Enhanced Index Tracking and Multi-Objective Portfolio Optimization

As an extensive overview of the literature on enhanced indexation is provided in Beasley *et al.* [8], Canakgoz and Beasley [10], and Guastaroba and Speranza [26], we primarily discuss the additional literature not included in those references. Furthermore, we refer the reader interested in metaheuristics for the index tracking problem to the survey by di Tollo and Maringer [15].

Koshizuka *et al.* [35] formulate the enhanced index tracking problem as a convex minimization model with linear objective function and quadratic constraints. The proposed model selects the optimal portfolio minimizing the tracking error from an index-plus-alpha portfolio among those with a composition similar to that of the benchmark. Index-plus-alpha portfolio is a term sometimes encountered in the literature on enhanced indexation and identifies a portfolio that outperforms the benchmark by a small amount α. The authors propose two alternative measures of the tracking error: the absolute deviation and the downside absolute deviation between the portfolio and the index-plus-alpha portfolio values. Mezali and Beasley [45] apply quantile regression to index tracking and enhanced indexation. Their model includes a limit on the total amount spent in proportional transaction costs (fixed transaction costs are not considered), a cardinality constraint and lower and upper bounds on the proportion of capital invested in each stock. The resulting formulation is a MILP model. Valle *et al.* [55] present a three-stage solution approach for the problem of selecting an absolute return portfolio, and show how it can be extended to the enhanced index tracking problem.

The enhanced index tracking problem has been investigated also from a stochastic programming perspective. Lejeune [36] formulates the enhanced index tracking problem as a stochastic game theoretic model where the probabilistic excess return of the portfolio over the benchmark is maximized while ensuring that the risk, measured by the downside absolute deviation, does not exceed a chosen maximum level. Lejeune and Samatlı-Paç [37] introduce a stochastic mixed integer non-linear model for the enhanced index tracking problem where stock returns and the return covariance terms are treated as random variables. Two enhanced index tracking formulations based on the second-order stochastic dominance are presented in Roman *et al.* [51]. Both models select a portfolio whose return distribution dominates the distribution of the benchmark with respect to the second-order stochastic dominance paradigm. A tutorial on the portfolio optimization problem, with a more general perspective, but including the index tracking and the enhanced indexation problems, has been recently provided by Beasley [7].

Some papers recently appeared on approaches to portfolio optimization that consider more criteria than the classical expected return and variance. In [17] two approaches to the solution of multi-objective problems for portfolio optimization are compared. A tri-objective optimization problem is studied by Chiam *et al.* [12]. The three objective functions considered are: the minimization of the portfolio variance, the maximization of the expected return and the maximization of the portfolio Sharpe ratio. The model includes both minimum and maximum cardinality constraints, as well as lower and upper bounds on the investment in each stock. An evolutionary multi-objective algorithm is designed to solve the problem. Hirschberger *et al.* [29] consider a general tri-criterion portfolio selection problem comprising two linear objectives, one quadratic convex objective (portfolio variance), and linear constraints. A special pivoting technique is proposed to describe the resulting piecewise quadratic and convex efficient set. Anagnostopoulos and Mamanis [3] investigate a tri-objective optimization problem where the portfolio variance and the number of securities included in the portfolio are minimized, while the portfolio expected return is maximized. They introduce lower and upper bounds on each stock, as well as lower and upper class limits (i.e., constraints on the investment in stocks that show common characteristics). The optimization problem is solved by means of three evolutionary algorithms.

The following contributions propose multi-objective approaches to the problems of index tracking and enhanced indexation. Chiam *et al.* [13] propose a bi-objective formulation for the index tracking problem in a rebalancing context. The two goals to be simultaneously minimized are the tracking error and the total amount of transaction costs paid. The model includes both fixed and proportional transaction costs, minimum transaction lots, and lower and upper bounds on each stock. A multi-objective evolutionary heuristic is used to solve the problem. Wu *et al.* [58] propose a goal programming approach for the bi-objective enhanced index tracking problem. The authors formulate the tracking error (to be minimized) as the standard deviation of the portfolio return from the benchmark. The second objective (to be maximized) is the excess portfolio return over the benchmark. Li *et al.* [38] formulate the enhanced index tracking problem as a bi-objective optimization model where the tracking error, measured by the downside standard deviation of the portfolio return from the benchmark, is minimized, whereas the excess portfolio return is maximized. The model includes a cardinality constraint, lower and upper bounds on each stock, and proportional transaction costs. An immunity-based multi-objective algorithm is presented and tested.

Recently, some researchers showed that the idea of index tracking can also be applied to track benchmarks for other applications. Andriosopoulos *et al.* [4] tackle the problem of reproducing the performance of international market capitalization shipping stock indices and physical shipping indices. The authors propose a model that aims at optimizing a weighted function of the tracking error, measured by the standard deviation of the portfolio return from the benchmark, and the excess portfolio return over the benchmark. A differential evolution algorithm and a genetic algorithm are used to solve the problem.

2.2 Multi-Objective Approaches for General MILP Problems

A primary goal in multi-objective optimization is the design of efficient methods for building the set of *Pareto optimal* solutions, the so-called *Pareto set*. Typically, the cardinality of the Pareto set is very high or even infinite. Finding the complete Pareto set may be computationally challenging and generate an excessive amount of information. As a consequence, most of the literature focuses either on finding a representative subset of the Pareto solutions, or on meta-heuristics (mainly evolutionary algorithms) to generate an approximation of the Pareto set, or on interactive methods, where the decision maker actively guides the exploration of the Pareto set looking for a small set of preferred solutions. Several surveys are available in the literature on specific aspects of multi-objective optimization. In particular, we mention Ruzika and Wiecek [52] for an overview on approximation methods, Coello Coello *et al.* [14] for an extensive bibliography on evolutionary approaches to general problems, and Alves and Clímaco [2] for a review of interactive approaches for integer and MILP problems. Here, we discuss a few specific references that are more closely related to our work.

While there is a considerable amount of literature on multi-objective (pure) integer linear programming problems, work on Multi-Objective MILP (MOMILP) problems is less common. A few papers tackle the exact description of the Pareto set for a MOMILP problem. Mavrotas and Diakoulaki [42] design a depth-first, binary branch-and-bound method for MOMILP problems with binary variables. In [43] and [56] the approach proposed in [42] is corrected and improved. Recently, in [54] a new branch and bound method is presented for solving a class of MOMILP problems, where only two objectives are allowed, the integer variables are binary, and one of the two objectives has only integer variables. An exact algorithm is proposed by Özpeynirci and Köksalan [50] to find all extreme supported efficient points (see Ehrgott [16]) of MOMILP problems. This algorithm uses a composite linear objective function and finds all the desired points in a finite number of steps by changing the weights of the objective functions in a systematic way. All the other approaches to MOMILP problems are focused on finding a representative subset of the Pareto solutions or on interactive methods. Solanki [53] propose a two-phase method for bi-objective mixed integer (not necessarily binary) linear optimization, which returns a finite set of points approximating the complete Pareto set. Points are generated recursively until the maximum possible error is not larger than a specified threshold value. Alves and Clímaco [1] propose an interactive reference point method. An initial reference point is fixed and a problem is solved in order to find an efficient point with the smallest Tchebycheff distance from the reference point. Then, according to the preferences expressed by the decision maker, the reference point is moved and a new efficient point with the smallest Tchebycheff distance is searched for. This process continues until the decision maker is satisfied. Each optimization problem is solved by a branch-and-bound algorithm.

3 The Optimization Model

This section is devoted to the enhanced index tracking model. In Section 3.1 we provide a detailed description of the bi-objective enhanced index tracking formulation we propose. Then, in Section 3.2 we introduce a variable reduction procedure and some valid inequalities in order to strengthen the mathematical formulation.

3.1 The Enhanced Index Tracking Model

We consider a financial market where n stocks are available for the investment and assume that we observe the financial market over time periods $t = 0, 1, \dots, T$. These time periods can be seen as scenarios generated with some techniques, for instance any of the generation techniques considered in [25]. The enhanced index tracking model we propose aims at determining at time period T the optimal portfolio composition to hold for a number of time periods immediately following T (i.e., the investment horizon). To this aim, we use a classic historical look-back approach, that is, the optimal portfolio composition is determined using the data observed in a number of time periods prior to the date the portfolio is selected. The assumption here is that past observations are good predictors of the future. In each time period t, $t = 0, 1, \dots, T$, and for each stock j, $j = 1, \dots, n$, we observe its price, denoted as q_{jt}, and the value of the benchmark, denoted as I_t. Stock prices and benchmark values are then used to compute stock and benchmark rates of return. For each time period t, $t = 1, \dots, T$, and each stock j, $j = 1, \dots, n$, we compute its rate of return as $r_{jt} = \frac{q_{jt} - q_{j,t-1}}{q_{j,t-1}}$, while the rate of return of the benchmark is computed as $r_{It} = \frac{I_t - I_{t-1}}{I_{t-1}}$.

We consider an investor holding a portfolio of stocks, hereafter referred to as the *current portfolio.* We are interested in rebalancing the current portfolio composition in time period T maximizing the portfolio average excess return over the benchmark while simultaneously minimizing the tracking error. Let the parameters X_j^0, $j = 1, \ldots, n$, be the number of units of stock j included in the current portfolio. The decision variables X_j^1, $j = 1, \ldots, n$, represent the number of units of stock j included in the new portfolio after rebalancing, hereafter referred to as the *rebalanced portfolio.* We assume that short sales are not allowed, i.e., both X_j^0 and X_j^1 are constrained to be non-negative.

Let us assume that the investor has a capital C available for the investment in time period T. The value of C is equal to the value of the current portfolio in time period T (i.e., $\sum_{j=1}^n q_{jT} X_j^0$) plus cash change (denoted as τ), that can be either an additional fund or a withdrawal, i.e., $C = \sum_{j=1}^n q_{jT} X_j^0 + \tau$. We assume that the investor can either purchase or sell a stock to rebalance the current portfolio composition. Selling a stock is possible only up to its amount in the current portfolio. In case the investor buys $X_j^1 - X_j^0$ units of stock j, i.e., $X_j^1 - X_j^0 > 0$, the investor pays a transaction cost that is proportional to the value of stock j that has been bought, i.e., $c_j^b(X_j^1 - X_j^0)q_{jT}$, where c_j^b is the proportional transaction cost paid for buying stock j. Conversely, if $X_j^0 - X_j^1 > 0$, the investor pays a transaction cost that is proportional to the value of stock j that has been sold, i.e., $c_j^s(X_j^0 - X_j^1)q_{jT}$, where c_j^s is the proportional transaction cost paid for selling stock j. To model these costs, we introduce two non-negative decision variables, denoted as b_j and s_j, $j = 1, \ldots, n$, where b_j represents the value purchased of stock j, possibly zero, and s_j represents the value sold of stock j, possibly zero. Additionally, whenever stock j is traded (either bought or sold) a fixed transaction cost, denoted as f_j, $j = 1, \ldots, n$, is paid. Let w_j, $j = 1, \ldots, n$, be a binary variable that takes value 1 if the investor buys or sells any fraction of stock j, and 0 otherwise.

We assume that the total transaction cost cannot exceed a fraction ρ of the capital, and that if stock j is selected in the optimal solution, the fraction of capital must be within a lower bound, denoted as λ_j, and an upper bound, denoted as ν_j. To this aim, we introduce a binary variable y_j, $j = 1, \ldots, n$, that takes value 1 if the investor holds any unit of stock j, and 0 otherwise. We also assume that all parameters X_j^0 fulfill these lower and upper bounds. Finally, in order to maintain a low portfolio cardinality, we constrain the number of stocks included in the rebalanced portfolio to be less than or equal to a certain threshold k.

3.1.1 Objective Functions

The enhanced index tracking formulation we propose is bi-objective. On one hand, the model aims at maximizing the average excess return compared with the benchmark. We formulate the average excess return objective function as

$$\text{maximize} \quad z_1 = \frac{1}{T} \sum_{t=1}^{T} \left[\sum_{j=1}^{n} r_{jt} q_{jT} X_j^1 - r_{It} C \right], \tag{1}$$

where $\sum_{j=1}^n r_{jt} q_{jT} X_j^1$ is the portfolio return in time period t, whereas $r_{It}C$ is the return yielded in the same time period if the capital C is invested in the benchmark.

On the other hand, we adopt the absolute deviation to measure the tracking error between the benchmark and the rebalanced portfolio, to be minimized. The tracking error is formulated as

$$TrE = \sum_{t=1}^{T} \left| \theta I_t - \sum_{j=1}^{n} q_{jt} X_j^1 \right|, \tag{2}$$

where $\theta = C/I_T$ is used to scale the value of the benchmark to be tracked. Standard methods to linearize the absolute deviation (see [26] for further details) lead to a linear formulation of the second objective function

$$\text{minimize} \quad z_2 = \sum_{t=1}^{T}(d_t + u_t) \tag{3}$$

requiring the introduction of the following constraints

$$d_t - u_t = \left(\theta I_t - \sum_{j=1}^{n} q_{jt} X_j^1\right) \quad t = 1, \ldots, T \tag{4}$$

for non-negative variables d_t and u_t, $t = 1, \ldots, T$. For a given t, $t = 1, \ldots, T$, d_t represents the downside absolute deviation of the portfolio value compared with the benchmark, whereas u_t is the upside absolute deviation. We chose to formulate the tracking error using the absolute deviation primarily because of the computational advantages of using a more tractable measure of risk than other more standard measures of deviations, such as the variance.

3.1.2 Constraints

In addition to (4), the following sets of constraints are needed. Constraints that impose that the amount of capital invested in each stock respects the minimum and maximum limits chosen are required. If a stock is not selected (that is if $y_j = 0$) obviously no capital can be invested in the stock. Constraints are also required to impose that the portfolio does not include a number of stocks larger than k, i.e., to limit the number of stocks selected in a portfolio. Finally, we need to limit the investment (the rebalanced portfolio cannot cost more than the capital available, that is the value of the current portfolio plus cash change) and the transaction costs (that are paid out of a different account and cannot exceed a percentage of the capital available for the investment). To express the latter constraint we need variables b_j, s_j and w_j that trigger the proportional transaction costs charged for buying and for selling stock j, respectively, and the fixed transaction cost.

More formally, constraints

$$\lambda_j C y_j \leq X_j^1 q_{jT} \leq \nu_j C y_j \quad j = 1, \ldots, n \tag{5}$$

ensure that if stock j is selected in the rebalanced portfolio, i.e. $X_j^1 > 0$, then the associated binary variable y_j takes value 1. In this case, the value of stock j in the optimal portfolio is constrained to be within the user-defined lower and upper bounds $\lambda_j C$ and $\nu_j C$.

The cardinality constraint on the number of stocks included in the rebalanced portfolio can be formulated as

$$\sum_{j=1}^{n} y_j \leq k. \tag{6}$$

The budget constraint

$$\sum_{j=1}^{n} q_{jT} X_j^1 = C \tag{7}$$

imposes that the capital invested in the rebalanced portfolio is equal to the capital available for the investment in time period T, that is the value of the current portfolio $\sum_{j=1}^{n} q_{jT} X_j^0$ plus cash change τ.

The two variables b_j and s_j are defined by means of the following constraints

$$b_j - s_j = (X_j^1 - X_j^0) q_{jT} \quad j = 1, \dots, n. \tag{8}$$

The constraints

$$b_j + s_j \leq \nu_j C w_j \quad j = 1, \dots, n \tag{9}$$

ensure that if stock j is transacted, i.e., either variable b_j or s_j takes a non-zero value, then variable w_j is forced to take value 1.

The total transaction cost paid by the investor is constrained to be less than or equal to a proportion ρ of the capital available for investment in time period T by means of the following constraint

$$\sum_{j=1}^{n} (c_j^b b_j + c_j^s s_j + f_j w_j) \leq \rho C. \tag{10}$$

Finally, the following constraints define the decision variables

$$X_j^1, b_j, s_j \geq 0 \quad j = 1, \dots, n \tag{11}$$

$$d_t, u_t \geq 0 \quad t = 1, \dots, T \tag{12}$$

$$y_j, w_j \in \{0, 1\} \quad j = 1, \dots, n. \tag{13}$$

The case of an optimal portfolio that has to be selected from scratch (i.e., no current portfolio is available) is considered setting $X_j^0 = 0$, $j = 1, \dots, n$. If proportional transaction costs for buying are equal to those for selling, then $c_j^b = c_j^s$. If proportional or fixed transaction costs are not present, c_j^b or c_j^s or f_j are set to 0, as needed. If no cardinality constraint has to be considered, then $k = n$.

3.2 Strengthening the Mathematical Formulation

This section is devoted to the introduction of a variable reduction procedure and some valid inequalities aimed at strengthening the above mathematical formulation. Note that if stock j is not present in the current portfolio (i.e., $X_j^0 = 0$), variables w_j and y_j take the same value: $w_j = y_j = 1$ if stock j is bought, or $w_j = y_j = 0$ if stock j is not bought. We can then reduce the size of the model by taking this into account. For each stock j such that $X_j^0 = 0$ we replace w_j with y_j. Depending on the composition of the current portfolio, the procedure may lead to a remarkable reduction of the number of binary variables. Constraints (9) are taken from the formulation for the index tracking problem proposed in [26]. A strengthened formulation can be obtained by replacing constraints (9) with the following two sets of constraints

$$b_j \leq (\nu_j C - X_j^0 q_{jT}) w_j \quad j = 1, \dots, n, \tag{14}$$

$$s_j \leq X_j^0 q_{jT} w_j \quad j = 1, \dots, n. \tag{15}$$

Constraints (14) bound the amount that the investor can further invest in each stock j. Indeed, for each stock j, given its value in the current portfolio $X_j^0 q_{jT}$, and the maximum amount that can be invested $\nu_j C$, the investor can increase its investment in stock j by at most $(\nu_j C - X_j^0 q_{jT})$. On the other hand, for each stock j, constraint (15) ensures that the investor does not sell more than the amount in the current portfolio. In Section 5.3, we validate constraints (14) and (15) through computational experiments.

Henceforth, we refer to the optimization model (1)-(13) where constraints (9) are replaced with inequalities (14) and (15) as the *Bi-Objective Enhanced Index Tracking (BOEIT) model.* We refer to the mono-objective formulation obtained by removing objective function (3) from the BOEIT model and adding the following constraint

$$\sum_{t=1}^{T}(d_t + u_t) \leq \xi C \tag{16}$$

where ξ is a given parameter, as the *EIT model.* The procedure mentioned above that replaces some binary variable w_j with the corresponding y_j is applied to both optimization models.

4 Kernel Search for the Bi-Objective Enhanced Index Tracking Model

In this section, we consider the EIT and the BOEIT models. We first describe the main ingredients of the mono-objective Kernel Search designed for the EIT model, and then we describe in detail its extension to the BOEIT model.

4.1 Kernel Search for the EIT Model

The basic idea of the *Kernel Search* is to intensively explore a sequence of relatively small/moderate-size portions of the solution space. The exploration is guided by the identification of subsets of decision variables and the subsequent solution of the resulting restricted problems by means of a general-purpose MILP solver used as a black-box. We apply the Kernel Search framework to the EIT model along the lines of the Basic Kernel Search designed in [26] for the index tracking problem. For the sake of brevity, we present here only the basic elements of the framework, and refer to [26] for further details.

The EIT model may be formulated as follows

$$\begin{array}{rl} \max & z_1 = g(\mathbf{x}) \\ \text{subject to} & f(\mathbf{x}) \leq \xi C \\ & \mathbf{x} \in \mathcal{X}, \end{array} \tag{17}$$

where $\mathbf{x}$ is the vector of variables, g describes the excess return, f is the tracking error, and $\mathcal{X}$ is the feasible region defined by constraints (4)-(8), (10)-(13), (14) and (15).

In the EIT model a set $\mathcal{N} = \{1, \ldots, n\}$ of stocks is given. Two binary variables, namely y_j and w_j, and three non-negative continuous variables, X_j^1, b_j and s_j, are associated with each stock j. From now on, we identify stock j with all the former variables associated with it. We denote as EIT($\mathcal{N}$) the EIT model that takes into account the whole set of stocks $\mathcal{N}$ (i.e., the original problem) and LP($\mathcal{N}$) the linear relaxation of EIT($\mathcal{N}$). We define as *kernel*, denoted as K, a subset of stocks.

The mono-objective Kernel Search is composed of two main phases. In the first phase, called the *initialization phase*, problem LP($\mathcal{N}$) is solved. If the optimal solution of LP($\mathcal{N}$) is integer, then the algorithm stops. Otherwise, the optimal solution of LP($\mathcal{N}$) provides an upper bound on the optimal cost to the original problem and is used to identify the promising stocks. All the stocks in set $\mathcal{N}$ are sorted and a ranked list R is created such that the stocks that are likely to be selected in an optimal solution of the original problem are ranked in the first positions. Subsequently, kernel K is initialized including the first m stocks in list R, where m is a given parameter, whereas the remaining $n-m$ stocks are partitioned into NB ordered clusters, called *buckets* and denoted as $\{B_h\}_{h=1,\dots,NB}$. We create a sequence of disjoint buckets each with cardinality equal to *lbuck*, a parameter, with the possible exception of the last one that may contain a smaller number of stocks. A lower bound is computed by solving problem EIT(K) restricted to the stocks included in the initial kernel. The second phase, referred to as the *solution phase*, is iterative. At each iteration h, with $h = 1,\dots,NB$, a restricted problem EIT$^\star(K \cup B_h)$ is solved, where EIT$^\star(K \cup B_h)$ is EIT($K \cup B_h$) with two additional constraints: (i) the lower bound is used as a cutoff value to the objective function, and (ii) the solution of the restricted problem must include at least one stock from the current bucket B_h. If EIT$^\star(K \cup B_h)$ is feasible, then at least one stock from the current bucket B_h is selected in its optimal solution. These stocks are added to the kernel. On the other hand, if a stock in the current kernel has not been selected in the optimal solution of EIT$^\star(K \cup B_h)$ and also in p of the restricted problems solved since it has been added to kernel K, where p is a given parameter, then that stock is removed from the kernel. Finally, if EIT$^\star(K \cup B_h)$ is feasible then the lower bound is updated. We refer to the algorithm described above as the *mono-objective Kernel Search*.

4.2 Bi-Objective Kernel Search for the BOEIT model

Using the same notation of model (17), the BOEIT model may be formulated as follows

$$\begin{aligned} \max \quad & z_1 = g(\mathbf{x}) \\ \min \quad & z_2 = f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X}. \end{aligned} \tag{18}$$

Let $\mathcal{Y} = \{(f(\mathbf{x}), g(\mathbf{x})) : \mathbf{x} \in \mathcal{X}\}$ be the image of the feasible set on the 2-dimensional objective space. Given two points $(\phi,\gamma), (\phi',\gamma') \in \mathcal{Y}$, we say that (ϕ,γ) is *dominated* by (ϕ',γ') if $\phi \geq \phi'$ and $\gamma \leq \gamma'$ and at least one inequality is strict. A point $(\phi,\gamma) \in \mathcal{Y}$ is *efficient* if it is not dominated by any point in $\mathcal{Y}$. A feasible solution $\mathbf{x} \in \mathcal{X}$ is *Pareto optimal* if its image $(f(\mathbf{x}), g(\mathbf{x}))$ is efficient. Let the *efficient set* $\mathcal{E} \subset \mathbb{R}^2$ be the set of all efficient points for a BOEIT instance.

The solution of the BOEIT model would imply an exact description of $\mathcal{E}$ along with the corresponding Pareto optimal solutions. However, such a solution would be impracticable from a computational point of view and would yield an overwhelming amount of information for the decision maker. A moderate size subset of $\mathcal{E}$ might be generated, possibly with an error guarantee on both objectives, as done in [18]. However, this approach would require the iterative application of an exact solution method for a mono-objective model derived from BOEIT, such as EIT. Again, this would be in general impracticable, due to the hardness of EIT and the size of real life instances. For instance, considering the S&P500 index over a time horizon of two-year weekly prices would result in an EIT model including 1000 binary variables, slightly more than 1700 continuous variables, and more than 2500 constraints.

Our aim is to find a moderate size set of points in the objective space, each one corresponding with a feasible solution of BOEIT, which approximates set $\mathcal{E}$. Hereafter, this set of points is denoted as $\mathcal{AF}$. Note that we do not require the points to be efficient. Hence, we determine them by a heuristic method. We pursue the task by embedding a modified version of the mono-objective Kernel Search described above into a variant of the ϵ-constraint method, one of the most known and widely adopted approaches for solving multi-objective optimization problems (see, e.g., [11, 47]). We call the resulting approach *Bi-Objective Kernel Search*.

4.3 Initialization Step

The first step in the method is to identify the area in the 2-dimensional objective space where the search has to take place (e.g., see [18]). With reference to the BOEIT model (18), the *ideal point* (ϕ_I, γ_I) is defined as

$$\phi_I = \min_{\mathbf{x}\in\mathcal{X}}\{f(\mathbf{x})\} \quad \text{and} \quad \gamma_I = \max_{\mathbf{x}\in\mathcal{X}}\{g(\mathbf{x})\} \tag{19}$$

and the *nadir point* (ϕ_N, γ_N) is defined as

$$\phi_N = \min_{\mathbf{x}\in\mathcal{X}}\{f(\mathbf{x}) : g(\mathbf{x}) \geq \gamma_I\} \quad \text{and} \quad \gamma_N = \max_{\mathbf{x}\in\mathcal{X}}\{g(\mathbf{x}) : f(\mathbf{x}) \leq \phi_I\}. \tag{20}$$

Such points define tight lower and upper bounds on the coordinates of all efficient points [16]. By construction, the points (ϕ_I, γ_N) and (ϕ_N, γ_I) are efficient, and every efficient point is contained in the rectangle having (ϕ_I, γ_N) as lower left corner and (ϕ_N, γ_I) as upper right corner.

The Bi-Objective Kernel Search approximates the above corner points by heuristically solving the corresponding optimization problems. More precisely, let $\mathbf{x}_I^f$ and $\mathbf{x}_I^g$ be the solutions obtained by applying the mono-objective Kernel Search to both problems in (19), respectively. Moreover, let $\mathbf{x}_N^f$ be the solution obtained by applying the mono-objective Kernel Search to the left problem in (20), with the unknown right-hand side γ_I replaced by $g(\mathbf{x}_I^g)$. Finally, let $\mathbf{x}_N^g$ be the solution obtained by applying the mono-objective Kernel Search to the right problem in (20), with the unknown right-hand side ϕ_I replaced by $f(\mathbf{x}_I^f)$. Then, the set of points $\mathcal{AF}$ is initialized by adding the approximate corners

$$(f_I, g_N) = \big(f(\mathbf{x}_N^g), g(\mathbf{x}_N^g)\big) \tag{21}$$

and

$$(f_N, g_I) = \Big(f(\mathbf{x}_N^f), g(\mathbf{x}_N^f)\Big)$$

as approximations of (ϕ_I, γ_N) and (ϕ_N, γ_I), respectively.

4.4 Iteration Step

In general, the ϵ-constraint method generates a sequence of mono-objective optimization problems, referred to as *ϵ-constraint problems*, by transforming all the objective functions but one (called the primary objective) into constraints. The choice of which objective is the primary one is usually driven by the structure of the problem. Since from our point of view the excess return is the most important objective, we select it as the primary one and constrain the value of the tracking error. In this way, the mono-objective problem to be solved at each iteration is an EIT model where the right-hand side of constraint (16) is replaced with a chosen value.

The Bi-Objective Kernel Search is based on the generation of a sequence of ϵ-constraint problems, where at each iteration i the problem

$$\max_{\mathbf{x}\in\mathcal{X}}\{g(\mathbf{x}) : f(\mathbf{x}) \leq \epsilon_i\} \tag{22}$$

is created by setting the value of ϵ_i equal to its value in the previous ϵ-constraint problem (f_I, if it is the first problem in the sequence) plus a constant step Δ. Note that (22) is an instance of the EIT (17). We introduce the integer parameter ϵ_{CP} representing the number of ϵ-constraint problems generated in the Iteration Step. The value of Δ is thus equal to $(f_N - f_I)/(\epsilon_{CP} + 1)$. Each ϵ-constraint problem i, $i = 1, \dots, \epsilon_{CP}$, is solved by means of the mono-objective Kernel Search described above with the following modifications.

Once ϵ-constraint problem $i-1$ is solved, the stocks are sorted. Let us assume that the current kernel, after solving ϵ-constraint problem $i-1$, contains m' stocks. A new ranked list R' is created where the m' stocks in the current kernel are placed in the first positions, while the remaining ones are sorted according to their position in ranked list R before the solution of problem $i-1$. The ranked list R', along with the value m', is used by the mono-objective Kernel Search to solve ϵ-constraint problem i, where R is set equal to R' and m is set equal to m'. The ranking R and the value m for the ϵ-constraint problem $i = 1$ are determined, taking as ϵ-constraint problem $i = 0$ the problem that determined the value f_I in the initialization step. Then, the kernel K and the sequence of buckets are generated according to ranked list R and parameter m along the lines described in Section 4.1 for the mono-objective Kernel Search, and point $(f(\mathbf{x}_i), g(\mathbf{x}_i))$ is added to $\mathcal{AF}$, where $\mathbf{x}_i$ is the solution found for ϵ-constraint problem i.

Finally, note that the best solution found for problem $i-1$ is feasible for all the successive ϵ-constraint problems. Hence, it is used as the initial solution for the mono-objective Kernel Search solving ϵ-constraint problem i.

An illustration of the evolution of the kernel and buckets is shown in Figure 1. Note that in this example the length of each bucket is chosen equal to the number of variables composing the initial kernel. Algorithm 1 outlines the Kernel Search for the BOEIT.

Algorithm 1 Bi-Objective Kernel Search for BOEIT.

```
1: Compute the approximated ideal and nadir points using a mono-objective kernel search
2: Set $\mathcal{AF} := \{(f_I, g_N), (f_N, g_I)\}$, $i := 1$ and $\epsilon_i := f_I + \Delta$.
3: Repeat the following while $i \leq \epsilon_{CP}$
    (a) identify the initial kernel $K$ and the sequence of buckets $\{B_h\}_{h=1,\dots,NB}$, using the ranking provided by the previous iteration;
    (b) solve EIT($K$) using the best solution found at the previous iteration as initial solution;
    (c) for all $h = 1, \dots, NB$
        (i) solve EIT$^\star$($K \cup B_h$);
        (ii) update the current kernel $K$.
    (d) let $\mathbf{x}_i$ be the best solution found (to problem (22)), add $(f(\mathbf{x}_i), g(\mathbf{x}_i))$ to $\mathcal{AF}$;
    (e) rank the variables according to the information collected during steps (b)-(c);
    (f) set $\epsilon_{i+1} := \epsilon_i + \Delta$ and $i := i + 1$.
```

4.5 Discussion and Comparison with the Literature

Similar to other implementations of the ϵ-constraint method (see, e.g., Mavrotas and Diakoulaki [42]), the Bi-Objective Kernel Search varies the right-hand side of the ϵ-constraint with

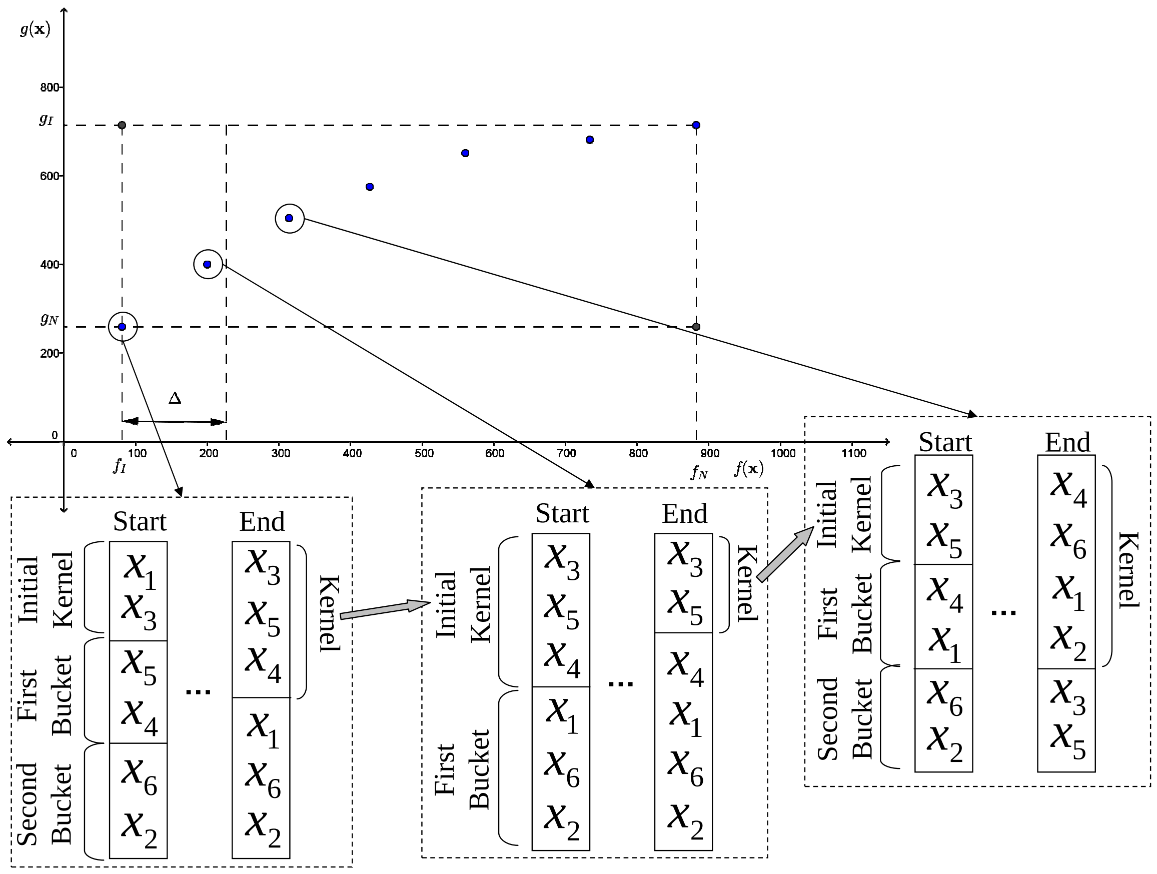


Figure 1: An illustrative example of the Kernel Search for BOEIT problems. For simplicity, x_j denotes the whole set of variables associated with stock j.

a fixed step Δ. However, there are two important differences between the method we propose and a straightforward application of a standard ϵ-constraint method.

Firstly, in a standard ϵ-constraint method, the right-hand side of the constraint associated with a minimization objective is reduced from an initial large value in order to cut off previously obtained points in the objective space. In the Bi-Objective Kernel Search we adopt the opposite direction, i.e., increase the right-hand side from an initial small value. The main rationale of this choice is the possibility of taking advantage of the kernel and of the buckets at the end of the solution of an ϵ-constraint problem in order to identify the initial kernel and the sequence of buckets to use for the solution of the following ϵ-constraint problem. Furthermore, as mentioned above, in the method we design the best solution found to a given ϵ-constraint problem is feasible for each of the following problems in the sequence. Hence, it can be used as initial solution for each of the latter problems. To the contrary, in a standard ϵ-constraint method, the best solution found to a given ϵ-constraint problem is infeasible for each of the following problems in the sequence.

Secondly, a standard ϵ-constraint method is usually designed to find efficient points in the objective space by applying an exact method. In the algorithm we propose, a heuristic method is applied. As a result, the obtained points are approximations of the efficient ones. Furthermore, since we apply a relatively large grid of possible values for the right-hand side of the ϵ-constraint, a moderate number of alternative points and solutions is returned. In this way, we are able to provide a quite accurate description of the trade-off between excess return and tracking error in reasonable computational time.

5 Experimental Analysis

This section is devoted to the presentation and discussion of the computational experiments. They were conducted on a PC Intel Xeon with 3.33 GHz 64-bit processor, 12.0 Gb of RAM and Windows 7 64-bit as Operating System. Algorithms and optimization models were implemented in Java. The LP and MILP problems were solved by means of CPLEX 12.2. After extensive preliminary experiments, we set the following CPLEX parameters. We set the number of passes that CPLEX performs when solving the root node of a MILP problem to 1 (parameter CutPass), the relative MIP gap tolerance equal to 10^{-6} (parameter EpGap), and we turned off the periodic heuristic (parameter HeurFreq). All the other CPLEX parameters were kept to their default values.

The present section is organized as follows. In Section 5.1 we briefly describe the testing environment. In Section 5.2 we describe the settings of the Bi-Objective Kernel Search used in the computational experiments and introduce the evaluation of its performance. In Section 5.3 we empirically validate the effectiveness of constraints (14) and (15). In Section 5.4 we validate the proposed mathematical formulation providing an analysis of the out-of-sample behavior for some of the optimal portfolios. In Section 5.5 we discuss the quality of the output and computational times of the Bi-Objective Kernel Search. Finally, in Section 5.6 we compare Algorithm 1 with a standard implementation of the ϵ-constraint method.

5.1 Testing Environment

We built two data sets for the computational experiments. Both data sets are derived from the 8 benchmark instances for the index tracking problem currently belonging to the OR-Library. The instances comprise stocks traded in different stock market indices: the Hang Seng market index (related to the Hong Kong stock market), DAX100 (Germany), FTSE100 (United Kingdom), S&P100 (USA), Nikkei225 (Japan), S&P500 (USA), Russell2000 (USA) and Russell3000 (USA). The number of stocks included in each instance ranges from 31 to 2151. The first five benchmark instances have been introduced in [8] (data collected from March 1992 to September 1997). The remaining three benchmark instances have been introduced in [10]. For each security, 291 weekly prices are provided. We considered the first 105 prices and computed the rate of return of stock j in time period t, $t = 1, \dots, T$, as $r_{jt} = \frac{q_{jt} - q_{j,t-1}}{q_{j,t-1}}$, where q_{jt} is the price of stock j in time period t and is equal to the $(t+1)$-th price provided in the original benchmark instance. Rates of return for the benchmarks have been computed similarly. In our experiments we have used sample data, following most of the literature we cited. The reader should be aware that using sample data might impact the out-of-sample performance of *any* strategy. As a consequence, the results concerning the out-of-sample validation of the proposed optimization model must be interpreted with care, since they might contain noise that is difficult to separate from the intrinsic performance of the proposed methodology.

As already mentioned, we generated two sets of instances. A first set assumes that the investor does not hold any current portfolio, i.e., $X_j^0 = 0$, $j = 1, \dots, n$. These instances are referred to as indtrack$I$, $I = 1, \dots, 8$ and Table 1 summarizes their main characteristics. Note that parameter k

in constraint (6) has been set to different values, depending on the instance size.

Instance	Benchmark	n	T	k
indtrack1	Hang Seng	31	104	10
indtrack2	DAX100	85	104	10
indtrack3	FTSE100	89	104	10
indtrack4	S&P100	98	104	10
indtrack5	Nikkei225	225	104	10
indtrack6	S&P500	457	104	40
indtrack7	Russell2000	1318	104	70
indtrack8	Russell3000	2151	104	90

Table 1: The main characteristics of the first set of instances.

A second set of instances was generated assuming a current portfolio composed of k stocks randomly selected in equal proportions, i.e., $X_j^0 = \frac{C/k}{q_{jT}}$ for each randomly selected j and $X_j^0 = 0$ otherwise. These instances are denoted as indtrack$I$rand, with $I = 1, \ldots, 8$. Any other characteristic of each instance in this second set is akin to the corresponding instance in Table 1.

To provide further insights on the BOEIT model, we conducted some experiments on a third data set that spans different market trends. This data set was introduced in Guastaroba *et al.* [25] and a short description is provided in Section 5.4. The three sets of instances are available at `http://or-brescia.unibs.it`.

We considered a budget available for the investment equal to 10^5. Furthermore, we chose $\lambda_j = 0.01$ and $\nu_j = 0.1$, $j = 1, \ldots, n$. We set $c_j^b = c_j^s = 0.01$ and $f_j = 12$, $j = 1, \ldots, n$. Finally, we used $\rho = 0.015$ and $\tau = 0$.

5.2 Kernel Search Parameters and Performance Evaluation

An effective implementation of the Bi-Objective Kernel Search framework involves an accurate calibration of its parameters. After preliminary experiments and based upon the computational results presented in [26], we made the following choices.

Concerning variable sorting, in the mono-objective Kernel Search used to approximate the ideal and nadir points we proceeded as follows. Let $\hat{c}(X_j^1)$ be the reduced cost of variable X_j^1 in the optimal solution of LP($\mathcal{N}$). The stocks are sorted in non-increasing order of the capital invested in each stock and, for those stocks not selected in the optimal solution of LP($\mathcal{N}$), in non-decreasing order of absolute values $\hat{c}(X_j^1)$. In the mono-objective Kernel Search used to solve the ϵ-constraint problems, the sorting criterion is driven by the kernel and buckets at the end of the solution of the previous problem. In order to approximate the ideal and nadir points we set parameter m equal to the number of stocks with a positive value in the optimal solution of LP($\mathcal{N}$). In order to solve the ϵ-constraint problems, parameter m is set equal to the number of stocks in the kernel at the end of the solution of the previous problem. In both cases, we set $lbuck = m$ and therefore $NB := \lceil \frac{n-m}{m} \rceil$. We set $p = 3$, and $\epsilon_{CP} = 98$, so that the number of points in set $\mathcal{AF}$, is not greater than 100. All the instances were solved setting a threshold on the computational time equal to 3600 seconds for the solution of each mono-objective problem.

Finally, we compared the set of points $\mathcal{AF}$ determined by means of the Bi-Objective Kernel Search to a reference set, denoted as $\mathcal{RS}$, obtained using CPLEX. More precisely, we applied Algorithm 1 where each optimization problem is solved by CPLEX. Furthermore, the best solution for problem $i - 1$ that CPLEX finds is used as the initial solution for ϵ-constraint problem i. The output of the Kernel Search for BOEIT is compared with the reference set using some of the measures proposed by Jaszkiewicz [33] for the evaluation of multi-objective metaheuristics. In order to complete the assessment, we also compared the computational times required by Algorithm 1 using the Kernel Search heuristics to those required by Algorithm 1 using CPLEX, where the time threshold for the solution of each mono-objective problem is set equal to 3600 seconds.

5.3 Strengthening the Mathematical Formulation: Computational Validation

In order to validate the effectiveness of inequalities (14) and (15) we computed, for each instance, two sets of efficient points in the objective space. The points are obtained by applying the procedure described in Section 5.2 to compute set $\mathcal{RS}$ with two exceptions. The first difference is that we optimize the LP relaxation of either the EIT model (including constraints (14) and (15)) or the variant of the EIT model where constraints (9) replace inequalities (14) and (15). We call *mEIT model* the latter variant of the EIT model. The second difference is that no initial solution is provided for the solution of any ϵ-constraint problem i. The results are shown in Table 2 where, for each instance and for each model type, 100 efficient points are computed. In the fourth column (labeled *#Impr.*) we give the number of times that the value of the optimal solution of the LP relaxation of the EIT model improved the optimal solution of the LP relaxation of the mEIT model. In the following column (labeled *Av. Impr.*) we show the average improvement, in percentage, of the EIT model compared with the mEIT model. The average improvement is computed as $100 \cdot \frac{z_{LP}(\text{EIT})-z_{LP}(\text{mEIT})}{z_{LP}(\text{mEIT})}$, where $z_{LP}(\cdot)$ is the optimal value of the LP relaxation of the corresponding model. The figures in column *Max Impr.* refer to the maximum improvement, in percentage, compared with the mEIT model.

		mEIT model	**EIT model**			
Instance	n	CPU (secs.)	# Impr.	Av. Impr.	Max Impr.	CPU (secs.)
indtrack1	31	1.201	0	0.00	0.00	1.107
indtrack2	85	1.919	0	0.00	0.00	1.779
indtrack3	89	1.872	0	0.00	0.00	2.730
indtrack4	98	2.060	0	0.00	0.00	2.138
indtrack5	225	4.087	0	0.00	0.00	4.274
indtrack6	457	7.129	0	0.00	0.00	7.207
indtrack7	1318	21.668	0	0.00	0.00	21.590
indtrack8	2151	37.191	0	0.00	0.00	37.830
indtrack1rand	31	1.326	0	0.00	0.00	1.108
indtrack2rand	85	2.028	0	0.00	0.00	1.840
indtrack3rand	89	1.934	0	0.00	0.00	1.966
indtrack4rand	98	2.121	0	0.00	0.00	1.997
indtrack5rand	225	3.885	0	0.00	0.00	4.056
indtrack6rand	457	7.285	99	-0.53	-0.78	7.222
indtrack7rand	1318	22.449	99	-1.29	-1.37	24.195
indtrack8rand	2151	39.562	99	-1.80	-1.92	39.451
Average		9.857		-0.23	-0.25	10.031

Table 2: Computational validation of inequalities (14) and (15).

Computing times of the two formulations are similar, but the EIT model slightly improves the mEIT model in terms of quality of the solution. Though on most instances there is no improvement, the three largest-scale instances that consider a current portfolio show remarkable improvements (an improvement occurred for almost all the efficient points). We made a similar test using the EIT model plus constraints (9). We obtained the same solutions of the EIT model but with longer computational times.

5.4 Bi-Objective Enhanced Index Tracking Model: Out-of-Sample Validation

The goal of this section is to provide evidence on the effectiveness of the BOEIT model in guiding financial decisions under different market conditions. To this aim, we decided to consider the instances introduced in Guastaroba *et al.* [25], in addition to the instances built upon those belonging to the OR-Library. The data set taken from Guastaroba *et al.* [25] comprises 4 instances built from historical weekly rates of return, computed by using the closing stock prices of the 100 securities composing the FTSE100 Index at the date of September 25th, 2005. These instances were generated to span four different market trends. More specifically, the first instance is characterized by an increasing trend of the market index (i.e., it is moving *Up*) in the in-sample period as well as in the out-of-sample period, and is hereafter referred to as instance *Up-Up*. The second instance has an increasing trend of the market index in the in-sample period and a decreasing one (i.e., it is moving *Down*) in the out-of-sample period, and from now on it is referred to as instance *Up-Down*. The third instance (henceforth referred to as instance *Down-Up*) is characterized by a decreasing trend in the in-sample period and by an increasing one in the out-of-sample period. Finally, the last instance (referred to as instance *Down-Down*) is characterized by a decreasing trend in both the in-sample and the out-of-sample periods. The temporal positioning of each instance is shown in Figure 2. Each of the former four instances assumes that the investor does not hold any current portfolio (i.e., $X_j^0 = 0$, $j = 1, \dots, n$), and was solved setting $k = 10$. All the other parameters were set as described in Section 5.1.

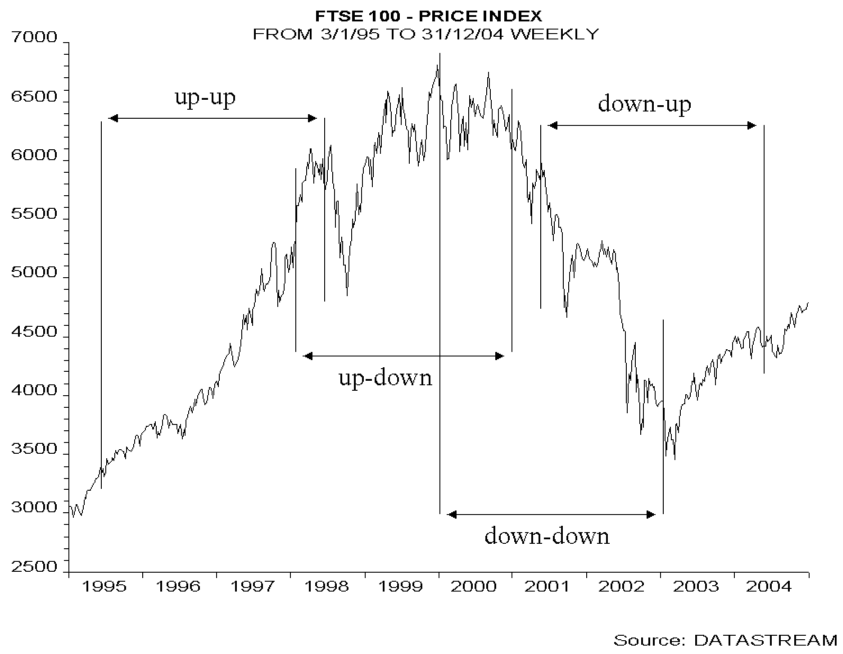


Figure 2: The four different market periods in the data set taken from Guastaroba *et al.* [25].

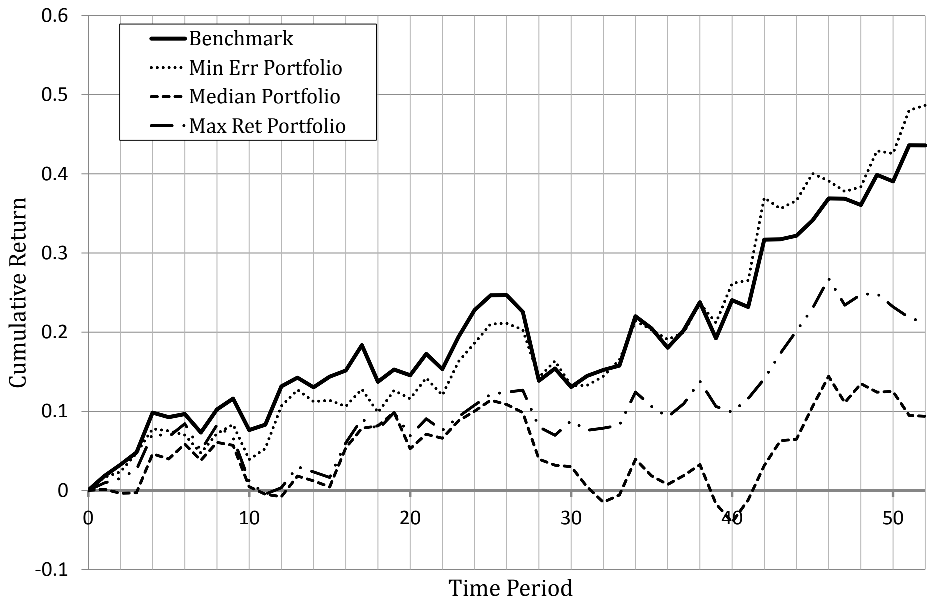
To illustrate how the portfolios selected by the BOEIT model perform out-of-sample, we observed their behavior in the 52 weeks following the date of portfolio selection and compared it to that of the benchmark. We divided the presentation of the out-of-sample results into two parts. We first compare the out-of-sample cumulative returns yielded by the benchmark to those achieved by some portfolios selected by the BOEIT model. Despite cumulative returns are not one of the standard measures used by portfolio managers to compare different strategies, they provide an immediate and easy-to-understand description of the dynamic behavior of the portfolios throughout the out-of-sample period. Subsequently, we analyze some performance measures computed at two instants of the out-of-sample period. In both cases, as CPLEX encountered computational problems when solving the largest instances (see the following section for further details), we considered the optimal portfolios corresponding with the points composing the reference set $\mathcal{RS}$ only for the data set taken from [25] and the instances indtrack1 to indtrack4 and indtrack1rand to indtrack4rand. Cumulative returns and performance measures for the remaining instances refer to the portfolios corresponding with the points composing set $\mathcal{AF}$ determined by means of the Bi-Objective Kernel Search. Finally, cumulative returns, as well as all the performance measures, have been computed using the capital invested in each security, i.e., the value taken by variable X_j^1, $j = 1, \ldots, n$. We recall that the transaction costs are constrained not to exceed a fraction $\rho = 0.015$ of the capital (see inequality (10)). We will show at the end of this section that including the transaction costs in the ex-post analysis of the optimal portfolios would cause negligible changes.

For the sake of clarity, in each figure where the cumulative returns are shown three portfolios are considered: the portfolio with minimum tracking error (denoted as *Min Err Portfolio*), corresponding with the solution of the right problem in (20); the closest portfolio to the median value of the tracking error computed over all the points found (denoted as *Median Portfolio*); and the portfolio with maximum average excess return (denoted as *Max Ret Portfolio*), corresponding with the solution of the left problem in (20). In each figure, the ex-post cumulative returns yielded by the benchmark are compared with those returned by these portfolios. For the sake of brevity, we present in the paper only the results concerning the data set from [25] (in Figures from 3(a) to 4(b)) and the instances indtrack2rand and indtrack6 (in Figures 5(a) and 5(b), respectively). The complete set of figures is available in the electronic supplementary material.

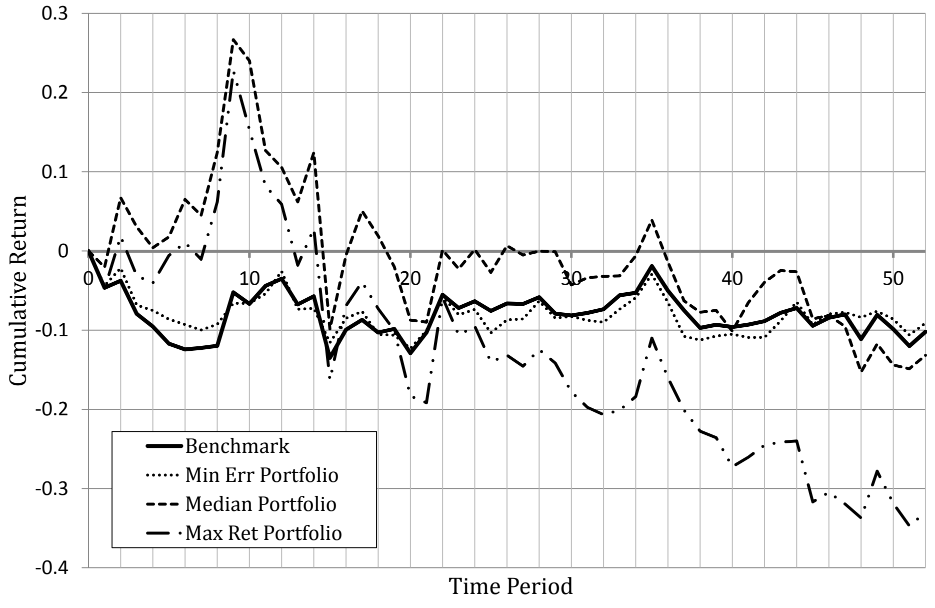
Figures 3 and 4 suggest that, according to the BOEIT model, if the in-sample period is characterized by an increasing trend of the market index, then portfolios with minimum tracking error are more advisable, whereas if the in-sample period is characterized by a decreasing trend, then portfolios that privilege the average excess return are preferable.

It is worth noting that in most of the figures, the Min Err Portfolio tracks closely the benchmark over all the out-of-sample period, performing worse than the other two portfolios in terms of excess return, as expected. This behavior is especially evident in Figure 5(b) for instance indtrack6. These results highlight that minimizing the tracking error in-sample by means of the BOEIT model may provide portfolios that successfully mimic the benchmark in the out-of-sample period. On the other hand, the figures display that if the investor is willing to take more risk, as measured by the tracking error, in order to yield larger excess returns, the portfolios selected in-sample by means of the BOEIT model may outperform the benchmark in the out-of-sample period. Indeed, in the majority of the instances (see also the electronic supplementary material) both the Median Portfolio and the Max Ret Portfolio considerably outperform the benchmark as well as the Min Err Portfolio. On the other hand, the former two portfolios replicate the fluctuations of the benchmark, although larger changes sometimes occur.

Similar conclusions can be drawn for the values of the performance measures of Table 3. For the sake of brevity, we present in this table only the performance measures computed for the Min Err Portfolio and the Median Portfolio. Additionally, we observed the behavior of the two portfolios after 26 weeks (columns 3 to 8) and 52 weeks (columns 9 to 14). We computed three performance measures. The first measure (columns with header *ER*) represents the average out-of-sample excess return of the portfolio over the benchmark, in percentage and on weekly basis.

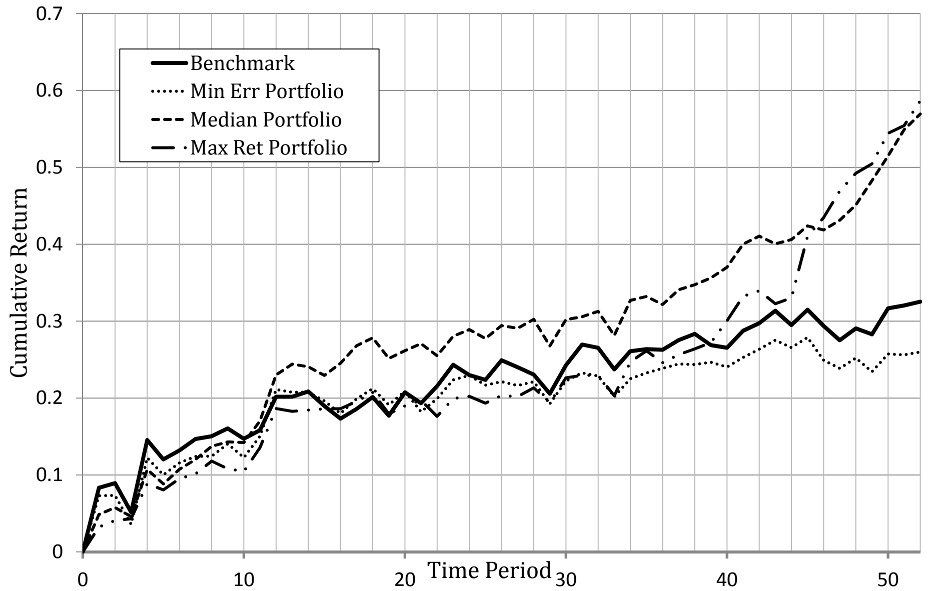


(a) Instance Up-Up.

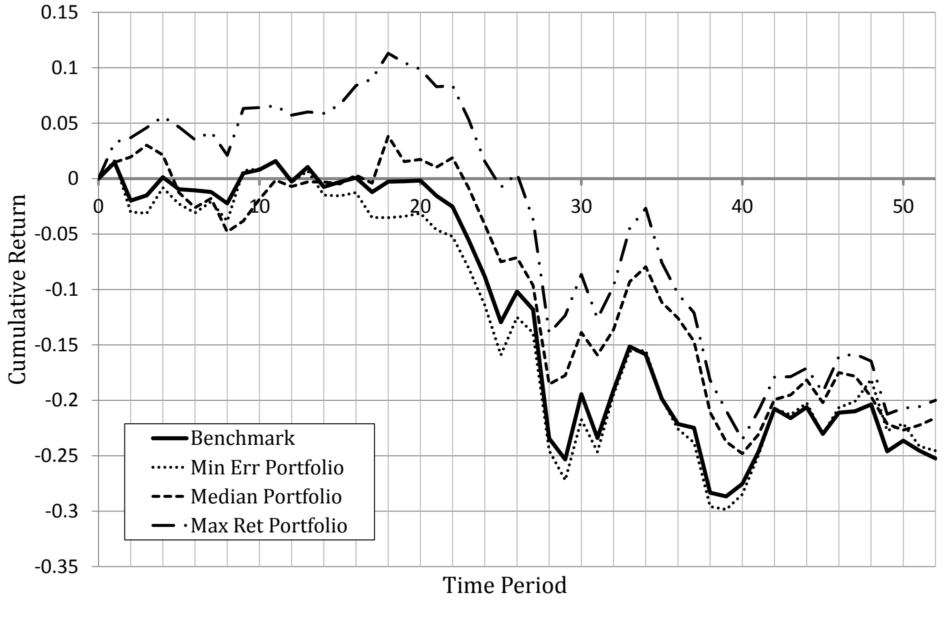


(b) Instance Up-Down.

Figure 3: Out-of-sample cumulative returns: BOEIT model solving instances Up-Up and Up-Down.

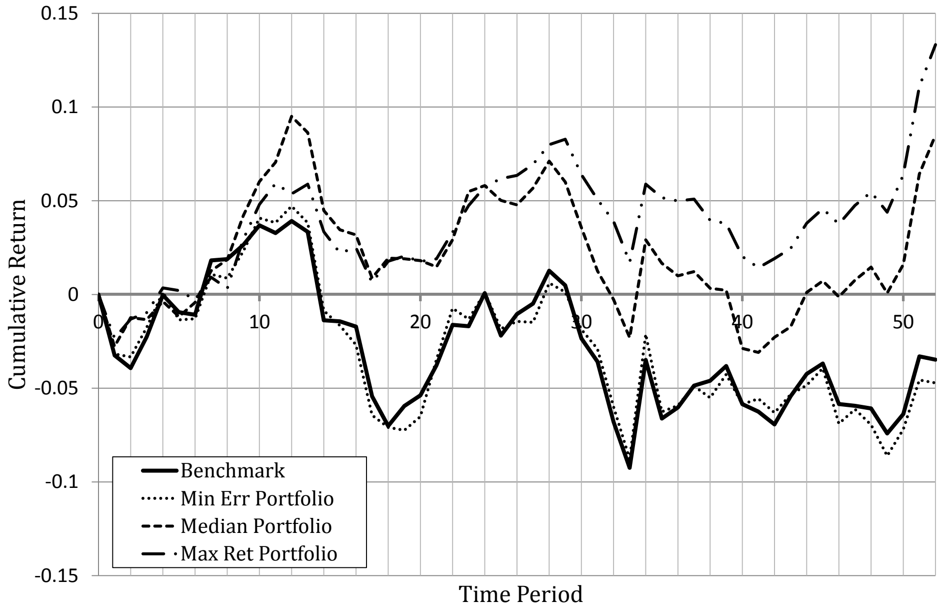


(a) Instance Down-Up.

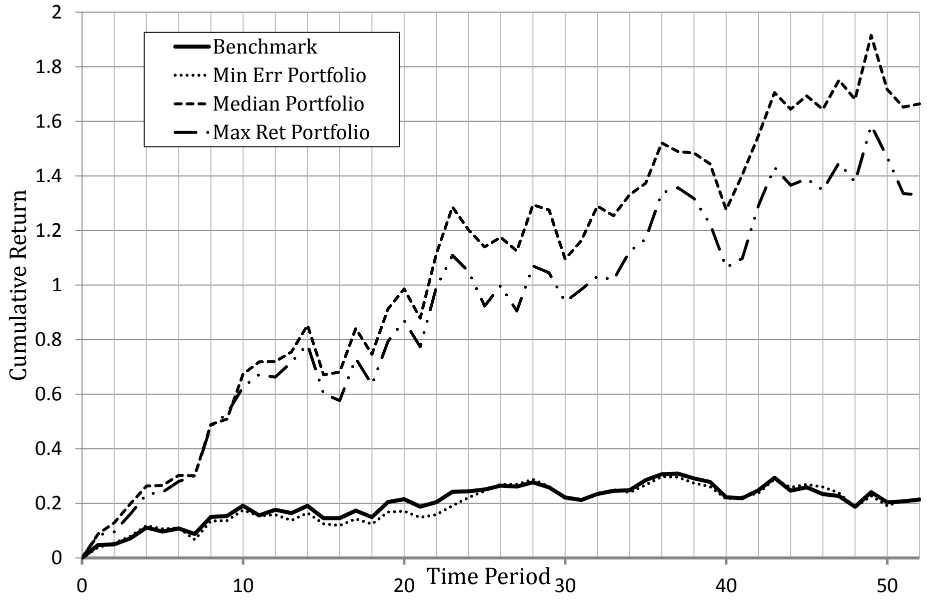


(b) Instance Down-Down.

Figure 4: Out-of-sample cumulative returns: BOEIT model solving instances Down-Up and Down-Down.



(a) Instance indtrack2rand.



(b) Instance indtrack6.

Figure 5: Out-of-sample cumulative returns: BOEIT model solving instances indtrack2rand and indtrack6.

		After 26 Weeks						After 52 Weeks					
		Min Err Portfolio			**Median Portfolio**			**Min Err Portfolio**			**Median Portfolio**		
Instance	n	ER	TE	IR	ER	TE	IR	ER	TE	IR	ER	TE	IR
Up-Up	100	-0.1125%	0.0062	-0.2279	-0.5270%	0.0147	-0.5243	0.0638%	0.0077	0.1044	-0.4675%	0.0162	-0.3695
Up-Down	100	-0.1022%	0.0117	-0.0952	0.4778%	0.0391	0.1302	0.0158%	0.0103	0.0168	0.0448%	0.0275	0.0162
Down-Up	100	-0.0894%	0.0064	-0.2143	0.0680%	0.0138	0.0581	-0.1000%	0.0058	-0.2705	0.2609%	0.0118	0.2449
Down-Down	100	-0.0914%	0.0053	-0.2823	-0.2142%	0.0142	-0.1920	0.0243%	0.0058	0.0662	-0.0602%	0.0169	-0.0543
indtrack1	31	-0.0698%	0.0058	-0.1470	0.3927%	0.0130	0.7411	-0.0771%	0.0058	-0.1347	0.0784%	0.0118	0.0988
indtrack1rand	31	-0.0698%	0.0058	-0.1470	0.1880%	0.0090	0.3200	-0.0771%	0.0058	-0.1347	-0.1293%	0.0094	-0.1645
indtrack2	85	0.0474%	0.0047	0.1283	0.1915%	0.0137	0.3498	-0.0585%	0.0051	-0.1326	0.2035%	0.0120	0.3813
indtrack2rand	85	-0.0143%	0.0050	-0.0445	0.2146%	0.0123	0.4393	-0.0225%	0.0050	-0.0740	0.2209%	0.0107	0.5061
indtrack3	89	0.1933%	0.0057	0.6529	0.0367%	0.0108	0.0530	0.1032%	0.0055	0.2839	0.0979%	0.0094	0.1562
indtrack3rand	89	0.0322%	0.0052	0.0812	0.1136%	0.0122	0.1407	0.0164%	0.0047	0.0473	0.1293%	0.0098	0.1709
indtrack4	98	0.0373%	0.0056	0.1031	-0.1813%	0.0110	-0.1737	-0.0221%	0.0054	-0.0516	0.0082%	0.0102	0.0098
indtrack4rand	98	0.1946%	0.0081	0.4191	-0.2748%	0.0097	-0.2831	0.0447%	0.0067	0.1108	-0.1337%	0.0092	-0.1618
indtrack5	225	-0.1046%	0.0056	-0.3864	0.0800%	0.0071	0.2955	-0.0662%	0.0055	-0.1877	-0.0771%	0.0073	-0.1621
indtrack5rand	225	0.1539%	0.0054	0.5879	-0.1610%	0.0054	-0.3608	0.0748%	0.0064	0.1627	-0.1685%	0.0066	-0.2715
indtrack6	457	0.0153%	0.0072	0.0298	2.2602%	0.0389	1.7694	-0.0077%	0.0069	-0.0175	1.6389%	0.0329	1.0942
indtrack6rand	457	0.0057%	0.0053	0.0119	1.9755%	0.0343	0.9327	0.0615%	0.0053	0.1495	1.2695%	0.0288	0.7734
indtrack7	1318	-0.3811%	0.0085	-0.4189	1.3890%	0.0330	0.8066	0.0935%	0.0085	0.1116	2.8169%	0.0512	1.1420
indtrack7rand	1318	0.1512%	0.0110	0.1510	1.6429%	0.0328	0.8876	0.1732%	0.0127	0.1363	2.8803%	0.0484	1.6173
indtrack8	2151	0.2602%	0.0075	0.4368	1.8543%	0.0384	1.2934	0.0932%	0.0080	0.1619	1.7937%	0.0367	1.3258
indtrack8rand	2151	0.3974%	0.0075	1.2641	1.6633%	0.0344	1.1582	0.2682%	0.0092	0.3986	0.7227%	0.0281	0.4167
Average		0.0227%	0.0067	0.0951	0.5595%	0.0199	0.3921	0.0301%	0.0068	0.0373	0.5565%	0.0197	0.3385

Table 3: Out-of-sample statistics for the Min Err Portfolio and the Median Portfolio.

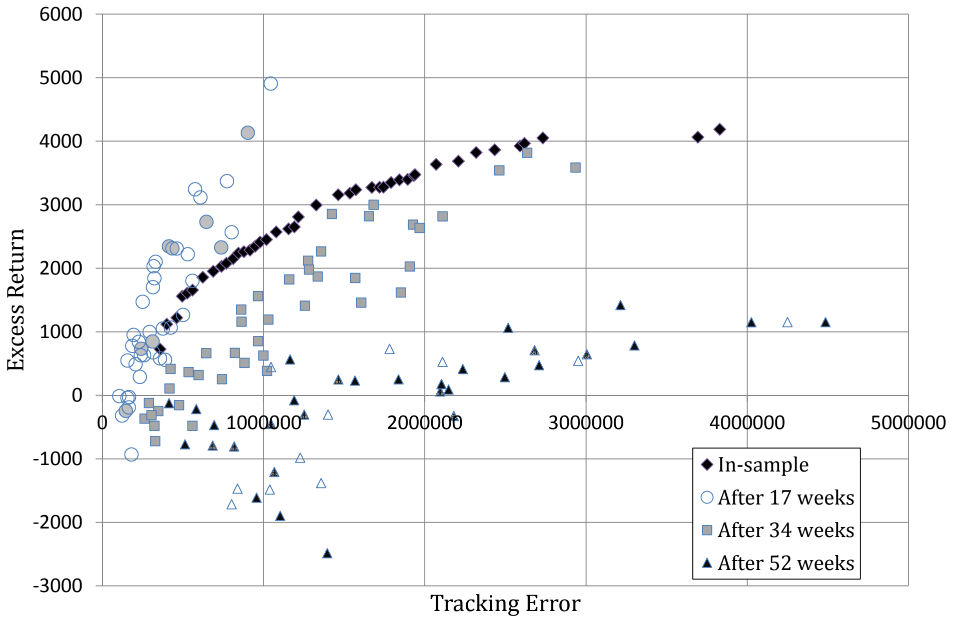


Figure 6: In-sample reference set and out-of-sample performance for instance indtrack1.

The second statistic (TE) gives a measure of the average ex-post tracking error, computed as the average absolute deviation between the portfolio and the benchmark rates of return. Finally, the third measure (IR) is a variant of the Information Ratio computed as the average excess return divided by the downside-standard deviation of the excess return, where the latter deviation is computed considering only the ex-post periods when the portfolio rate of return was worse than that yielded by the benchmark. This variant of the Information Ratio measures the average excess return of a portfolio per unit of downside risk incurred. All the three statistics computed for the Min Err Portfolios take, on average, smaller values than when calculated for the Median Portfolios. Hence, the average values confirm the conclusions we drew above: the Min Err Portfolio yields a worse return than the Median Portfolio (see the average figures for ER) but, on the other side, tracks more closely the benchmark (see the average figures for TE). Furthermore, the Median Portfolio achieved an average risk-adjusted return that is larger than that yielded by the Min Err Portfolio (see the average figures for IR). It is worth noting that the Median Portfolios achieved large excess returns for the largest instances, that are the instances that refer to the solutions computed by the Bi-Objective Kernel Search because of the problems caused by CPLEX.

Figure 6 compares the in-sample reference set $\mathcal{RS}$ computed for instance indtrack1 and its out-of-sample performance. The irregularities in the sequence of points composing set $\mathcal{RS}$ are explained by discontinuities in the efficient set, due to the presence of binary variables. For each point of set $\mathcal{RS}$, we depicted three points that show the ex-post tracking error (computed as in equation (2)) and the ex-post excess return (computed as in objective function (1)), calculated after 17, 34, and 52 weeks, respectively. Note that the tracking error (x-axis) depicted in Figure 6 tends to increase with the number of weeks. This behavior is mainly due to formula (2) that computes the tracking error as a summation over time. On the other hand, the excess return (y-axis) tends to decrease with the number of weeks. This behavior indicates that, sometimes but not on average as mentioned above, the performance of a portfolio deteriorates over time. In these cases, an optimization model such as the BOEIT can be used to rebalance the portfolio composition. A similar analysis of the out-of-sample performance of the efficient sets is provided in Figure 7 for the 4 instances taken from Guastaroba *et al.* [25]. Here, the in-sample reference set is omitted for simplicity.

Finally, Figures 8(a) and 8(b) give examples of the cumulative returns yielded by the optimal portfolios after the transaction costs paid have been discounted. To the sake of brevity, we report here only the pictures related to the instances Up-Up and Up-Down. Therefore, the latter two figures have to be compared with the plots depicted in Figures 3(a) and 3(b), respectively. The pictures illustrating the cumulative returns for the remaining instances that are shown in this section are available in the electronic supplementary material. The comparison between Figures 3(a) and 8(a), and between Figures 3(b) and 8(b) show that the differences are negligible. Two are the main reasons of this outcome. Firstly, transaction costs are paid only once, at the date of portfolio selection. Hence, they affect only the 'single-period' return related to the first week of the out-of-sample period. Secondly, the small value $\rho = 0.015$ used in constraint (10) limits the total amount paid in transaction costs.

5.5 Bi-Objective Kernel Search: Computational Results

In order to test the quality of the output and the computational times of the Bi-Objective Kernel Search, we now focus on the OR-library instances.

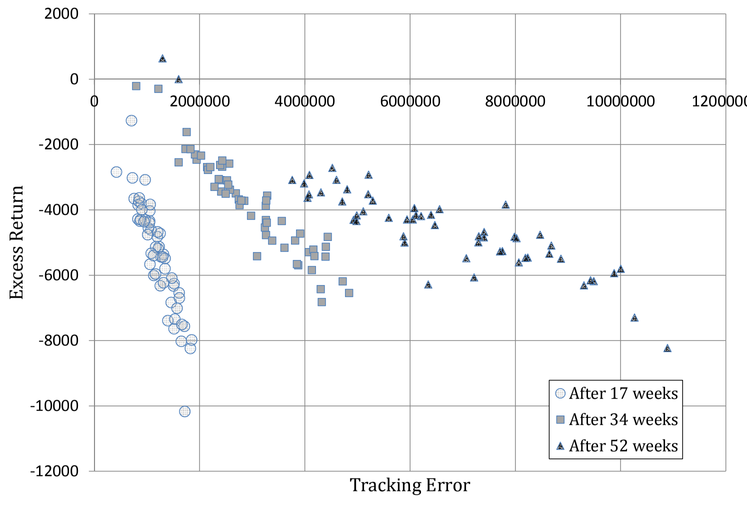
The computational results are organized in two groups. Instances indtrack1-indtrack4 and indtrack1rand-indtrack4rand are referred to as the small-scale instances, while the remaining instances are denoted as the large-scale instances. In Tables 4 and 5 we show the performance of the Bi-Objective Kernel Search for the small-scale and the large-scale instances, respectively. We divided the instances in two groups since CPLEX could not solve within the time limit the left-hand side problem in (19) for any of the large-scale instances. In these cases, the points found by CPLEX, if any (since for some instances no feasible solution was found), were dominated by the corresponding points found by the heuristic. These points were removed from the computation of the statistics of Table 5.

In Table 4, columns 3 and 4 refer to CPLEX. Specifically, column labeled $|\mathcal{RS}|$ gives the cardinality of the reference set. For each instance, the figures in column CPU (secs.) are the computational times (in seconds) required by CPLEX to compute the whole reference set. The following columns refer to the performance of the Bi-Objective Kernel Search. Columns $|\mathcal{AF}|$ and CPU (secs.) give, for each instance, the cardinality of the set of points approximating the efficient set and the time to compute it, respectively. Statistic Q_2 (the notation is taken from [33]) is a measure of the approximated points not dominated by any point in the reference set $\mathcal{RS}$. It is computed as

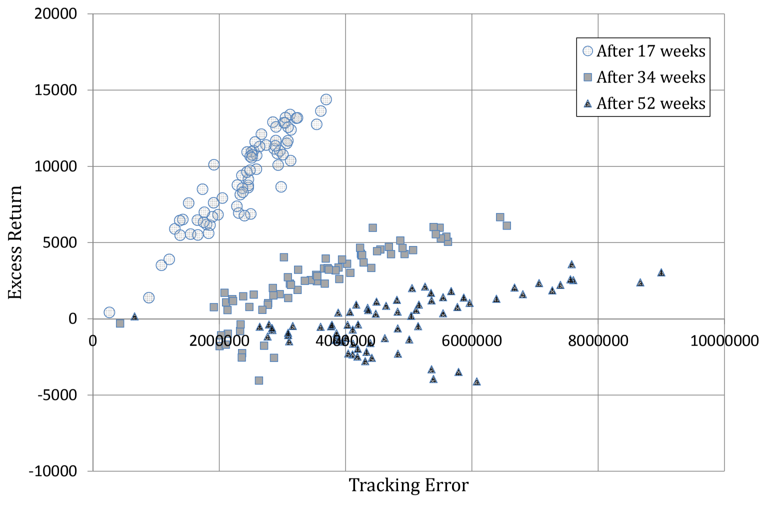
$$Q_2 = \frac{|\{(f,g) \in \mathcal{AF} : \nexists (\phi,\gamma) \in \mathcal{RS} \text{ such that } (\phi,\gamma) \text{ dominates } (f,g)\}|}{|\mathcal{AF}|}.$$

In Table 5, additional columns are considered, corresponding with the following statistics. Statistic d_f is a measure of the average error, in percentage, with respect to objective function f (i.e., the tracking error). It is computed as

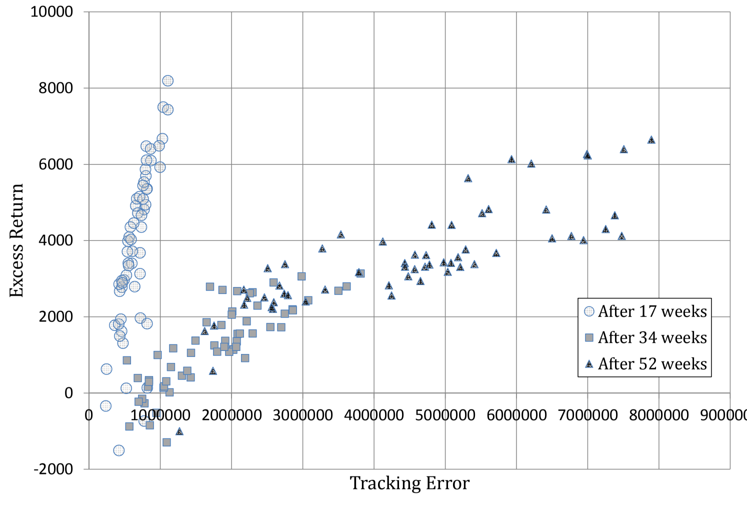
$$d_f = \frac{100}{|\mathcal{RS}|} \cdot \sum_{(\phi,\gamma)\in\mathcal{RS}} \min_{(f,g)\in\mathcal{AF}} \frac{|f-\phi|}{\phi}.$$



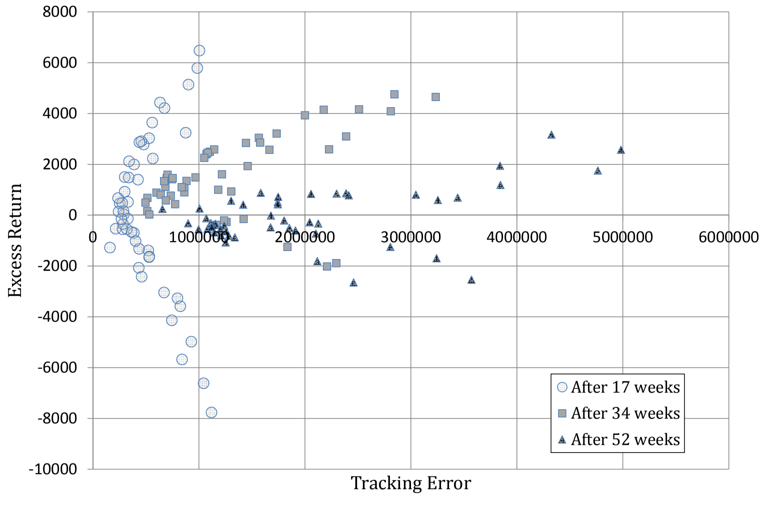
(a) Instance Up-Up.



(b) Instance Up-Down.

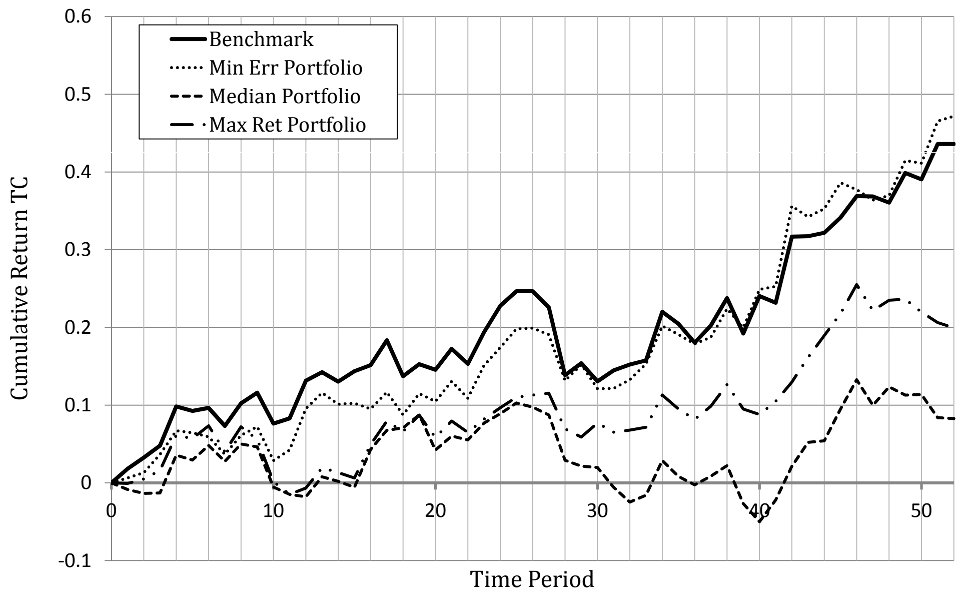


(c) Instance Down-Up.

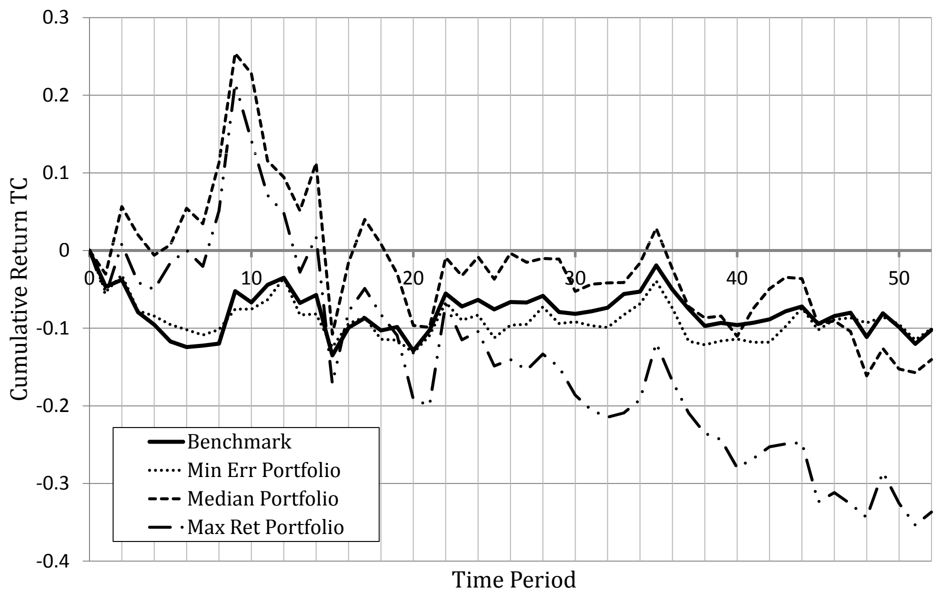


(d) Instance Down-Down.

Figure 7: Out-of-sample performance for the instances taken from Guastaroba *et al.* [25].



(a) Instance Up-Up.



(b) Instance Up-Down.

Figure 8: Out-of-sample cumulative returns after discounting transaction costs: BOEIT model solving instances Up-Up and Up-Down.

		CPLEX 12.2		Bi-Objective Kernel Search		
Instance	n	$\|\mathcal{RS}\|$	CPU (secs.)	$\|\mathcal{AF}\|$	CPU (secs.)	Q_2
indtrack1	31	42	10.951	42	13.790	1.000
indtrack2	85	64	239.211	64	125.424	1.000
indtrack3	89	81	579.213	81	214.017	1.000
indtrack4	98	76	809.818	76	347.490	1.000
indtrack1rand	31	41	10.327	41	11.232	1.000
indtrack2rand	85	55	157.030	55	90.651	1.000
indtrack3rand	89	83	120.729	83	67.283	1.000
indtrack4rand	98	79	240.225	79	86.517	1.000
Average		65.125	270.938	65.125	119.551	1.000

Table 4: Average statistics for the Bi-Objective Kernel Search (small-scale instances).

Similarly, the percentage average error with respect to objective function g (i.e., the average excess return) is computed as

$$d_g = \frac{100}{|\mathcal{RS}|} \cdot \sum_{(\phi,\gamma)\in\mathcal{RS}} \min_{(f,g)\in\mathcal{AF}} \frac{|g-\gamma|}{\gamma},$$

and statistic d_z is computed as

$$d_z = \frac{100}{|\mathcal{RS}|} \cdot \sum_{(\phi,\gamma)\in\mathcal{RS}} \min_{(f,g)\in\mathcal{AF}} \max\left\{\frac{|f-\phi|}{\phi}, \frac{|g-\gamma|}{\gamma}\right\}.$$

Statistic d_z^{max} is computed as

$$d_z^{max} = 100 \cdot \max_{(\phi,\gamma)\in\mathcal{RS}} \min_{(f,g)\in\mathcal{AF}} \max\left\{\frac{|f-\phi|}{\phi}, \frac{|g-\gamma|}{\gamma}\right\}.$$

The last four statistics are not given in Table 4 because their value is systematically null. In Table 5, for each instance, the figure in the fourth column shows the percentage error generated by CPLEX solving the left-hand side problem in (19), i.e., computing the first coordinate of the ideal point in reference set $\mathcal{RS}$. The error is computed as

$$\widehat{d_f} = 100 \cdot \frac{\widehat{\phi_I} - f_I}{f_I},$$

where $\widehat{\phi_I}$ is the f-value of the best solution found by CPLEX within the given time limit (*Inf.* identifies the instances where the solver did not find any feasible solution), and f_I is the f-value of the approximated ideal point computed by the mono-objective Kernel Search (see (21)). This statistic gives a measure of the error generated by CPLEX solving the large-scale instances (see the beginning of the present section).

Finally, statistics *Fr.* $d'_z < 1.0\%$, *Fr.* $d'_z < 0.5\%$ and *Fr.* $d'_z < 0.1\%$, in the last three columns, count the number of times that, in percentage, the error

$$d'_z = 100 \cdot \min_{(f,g)\in\mathcal{AF}} \max\left\{\frac{|f-\phi|}{\phi}, \frac{|g-\gamma|}{\gamma}\right\}$$

was smaller than 1.0%, 0.5% and 0.1%, respectively.

Looking at Tables 4 and 5, we see that the Bi-Objective Kernel Search is faster than CPLEX: the time reduction is on average 56% on the small-scale instances and 44% for the large-scale instances.

		CPLEX 12.2			Bi-Objective Kernel Search									
Instance	n	$\|\mathcal{RS}\|$	$\widehat{d_f}$	CPU (secs.)	$\|\mathcal{AF}\|$	CPU (secs.)	Q_2	d_f	d_g	d_z	d_z^{max}	Fr. d'_z $< 1.0\%$	Fr. d'_z $< 0.5\%$	Fr. d'_z $< 0.1\%$
indtrack5	225	84	2.461	7639.019	84	3799.542	0.988	0.039	0.072	0.073	4.684	97.590	97.590	97.590
indtrack6	457	100	117.752	5289.876	100	2935.867	0.970	0.000	0.004	0.004	0.226	100.000	100.000	97.980
indtrack7	1318	70	714.958	6598.656	70	3179.723	0.870	0.000	0.079	0.090	2.879	97.101	95.652	88.406
indtrack8	2151	82	*Inf.*	8219.280	82	3009.510	0.926	0.000	0.100	0.100	5.423	98.765	95.062	93.827
indtrack5rand	225	71	5.635	7362.751	71	3896.794	0.986	0.034	0.009	0.034	1.911	98.571	98.571	97.143
indtrack6rand	457	100	66.742	5725.225	100	4366.401	0.980	0.000	0.016	0.016	1.507	98.990	98.990	98.990
indtrack7rand	1318	84	*Inf.*	7272.093	84	4156.799	0.952	0.000	0.084	0.084	5.084	97.590	96.386	96.386
indtrack8rand	2151	66	*Inf.*	44007.178	66	25975.033	0.908	0.000	0.142	0.142	3.466	95.385	93.846	92.308
Average		82.125		11514.260	82.125	6414.959	0.948	0.009	0.063	0.068	3.148	97.999	97.012	95.329

Table 5: Average statistics for the Bi-Objective Kernel Search (large-scale instances).

		CPLEX 12.2 (standard ϵ-constraint)		Bi-Objective Kernel Search (standard ϵ-constraint)		
Instance	n	$\vert\mathcal{RS}\vert$	CPU (secs.)	$\vert\mathcal{AF}\vert$	CPU (secs.)	Q_2
indtrack1	31	35	9.953	35	9.890	1.000
indtrack2	85	52	319.598	52	804.551	1.000
indtrack3	89	62	941.192	62	206.184	1.000
indtrack4	98	62	1140.440	62	1643.931	1.000
indtrack1rand	31	33	8.253	33	7.613	1.000
indtrack2rand	85	46	221.115	46	180.960	1.000
indtrack3rand	89	65	209.275	65	357.693	1.000
indtrack4rand	98	61	475.427	61	553.505	1.000
Average		52.000	415.657	52.000	470.541	1.000

Table 6: Average statistics for a standard ϵ-constraint approach (small-scale instances).

The Bi-Objective Kernel Search provides high quality approximations. In particular, for the small-scale instances (see the figures in Table 4) set $\mathcal{AF}$ determined by means of the Bi-Objective Kernel Search comprises the same set of points included in the reference set $\mathcal{RS}$ (see, in particular, the values of Q_2). For the large-scale instances (see Table 5), out of a total of 657 points computed the maximum attained error on one of the objectives is 5.423% (see statistic d_z^{max}). However, 98% of the points are within an error of 1% on both objectives, and more than 95% are within a negligible error of 0.1% (see statistics Fr. $d'_z < 1\%$ and Fr. $d'_z < 0.1\%$, respectively).

5.6 Bi-Objective Kernel Search: A Computational Comparison with a Standard ϵ-Constraint Method

In order to assess the modified, with respect to the standard, ϵ-constraint method (see Section 4.5), we performed some computational experiments where both CPLEX and the Bi-Objective Kernel Search are embedded into a standard ϵ-constraint method. The results are given in Table 6 for small-scale instances and in Table 7 for large-scale instances.

Both CPLEX and the Bi-Objective Kernel Search turned out to be slower when embedded into a standard ϵ-constraint method. This is especially true for the small-scale instances (see Table 6). Here, with a standard ϵ-constraint implementation, the Bi-Objective Kernel Search is actually slower than CPLEX. Statistics d_f, d_g, d_z, and d_z^{max} are not shown in Table 6 since their value is systematically null. As long as the large-scale instances are concerned (Table 7), both CPLEX and Bi-Objective Kernel Search slowed down, but still the latter is the fastest. The standard implementation of the Bi-Objective Kernel Search returns negligible improvements of the efficient frontier, when compared with the non-standard implementation (the average errors achieved by the latter implementation are indeed quite small) with the exception of the largest errors (statistic d_z^{max}). Nevertheless, the slight improvements in terms of solution quality do not justify the longer computational times taken by the standard implementation. This is especially true if one considers the small-scale instances where both implementations of the Bi-Objective Kernel Search return the exact reference set, while the standard implementation is nearly 4 times slower than the non-standard one.

Finally, statistic $\widehat{d_f}$, in Table 5, is not shown in Table 7. The reason is that this statistic is not influenced by the implementation of the ϵ-constraint method.

		CPLEX 12.2 (standard ϵ-constraint)		**Bi-Objective Kernel Search (standard ϵ-constraint)**									
Instance	n	$\lvert\mathcal{RS}\rvert$	CPU (secs.)	$\lvert\mathcal{AF}\rvert$	CPU (secs.)	Q_2	d_f	d_g	d_z	d_z^{max}	Fr. d'_z $< 1.0\%$	Fr. d'_z $< 0.5\%$	Fr. d'_z $< 0.1\%$
indtrack5	225	71	10996.429	71	3741.689	1.000	0.000	0.000	0.000	0.000	100.000	100.000	100.000
indtrack6	457	100	5336.218	100	5822.616	0.980	0.000	0.006	0.006	0.532	100.000	98.990	98.990
indtrack7	1318	70	9995.279	70	3303.602	0.884	0.000	0.040	0.040	1.952	98.413	98.413	93.651
indtrack8	2151	81	7086.470	81	4632.772	0.938	0.016	0.031	0.047	1.450	97.500	96.250	96.250
indtrack5rand	225	60	8282.806	60	4756.341	1.000	0.000	0.000	0.000	0.000	100.000	100.000	100.000
indtrack6rand	457	100	5737.828	100	5096.373	0.929	0.000	0.008	0.008	0.354	100.000	100.000	96.970
indtrack7rand	1318	83	7125.894	84	7477.889	0.831	0.006	0.025	0.029	1.113	98.780	97.561	95.122
indtrack8rand	2151	63	39810.445	62	21169.071	0.852	0.009	0.051	0.060	1.591	98.361	96.721	85.246
	Average	78.500	11796.421	78.500	7000.044	0.927	0.004	0.020	0.024	0.874	99.132	98.492	95.779

Table 7: Average statistics for a standard ϵ-constraint approach for the Bi-Objective Kernel Search (large-scale instances).

6 Conclusions

In this paper we have introduced a bi-objective approach for enhanced index tracking problems, where the considered objectives are the average excess return and the deviation from a financial market index, respectively. We have proposed a bi-objective Mixed Integer Linear Programming formulation for the problem that includes several real-life features, and shown that this optimization model selects portfolios providing good results under different market conditions. The results related to a set of instances generated to span different market trends suggest that, according to the proposed bi-objective model, if the in-sample period is characterized by an increasing trend of the market index, then portfolios with minimum tracking error are more advisable, whereas if the in-sample period is characterized by a decreasing trend, then portfolios that privilege the average excess return are preferable. The results obtained solving a set of benchmark instances also highlight that minimizing the tracking error in-sample by means of the proposed model often provides portfolios that successfully mimic the market index in the out-of-sample period. On the other hand, they indicate that if the investor is willing to take more risk, as measured by the tracking error, in order to yield larger excess returns, the portfolios selected in-sample by means of the model may outperform the market index in the out-of-sample period. We have designed a solution approach that combines the Kernel Search (a heuristic framework for MILP problems) and a modified ϵ-constraint method. During the execution, a sequence of mono-objective optimization problems is solved, and the results obtained for every problem are exploited in the solution of the following one. Testing on benchmark instances shows that the proposed approach is able to compute in reasonable computational time a very accurate approximation of the trade-off curve between the two objectives, where each point of the curve is associated with an actual portfolio composition. The approximated curve may allow a fund manager to decide more carefully the most suitable portfolio composition.

Future work may take two directions. The first is the application of the approach to different bi-objective portfolio optimization problems. The second is the extension of the method to general bi-objective or multi-objective MILP problems.

Acknowledgements

We wish to acknowledge the comments and suggestions of the associate editor and three anonymous reviewers which allowed us to improve a former version of this paper.

References

[1] M.J. Alves and J. Clímaco. An interactive reference point approach for multiobjective mixed-integer programming using branch-and-bound. *European Journal of Operational Research*, 124(3):478–494, 2000.

[2] M.J. Alves and J. Clímaco. A review of interactive methods for multiobjective integer and mixed-integer programming. *European Journal of Operational Research*, 180(1):99–115, 2007.

[3] K.P. Anagnostopoulos and G. Mamanis. A portfolio optimization model with three objectives and discrete variables. *Computers & Operations Research*, 37(7):1285–1297, 2010.

[4] K. Andriosopoulos, M. Doumpos, N.C. Papapostolou, and P.K. Pouliasis. Portfolio optimization and index tracking for the shipping stock and freight markets using evolutionary algorithms. *Transportation Research Part E: Logistics and Transportation Review*, 52(0):16–34, 2013.

[5] E. Angelelli, R. Mansini, and M.G. Speranza. Kernel search: A general heuristic for the multi-dimensional knapsack problem. *Computers & Operations Research*, 37(11):2017–2026, 2010.

[6] E. Angelelli, R. Mansini, and M.G. Speranza. Kernel search: A new heuristic framework for portfolio selection. *Computational Optimization and Applications*, 51(1):345–361, 2012.

[7] J.E. Beasley. Portfolio optimisation: Models and solution approaches. In Topaloglu H., editor, *Tutorials in Operations Research, Vol. 10*, pages 201–221. INFORMS, 2013.

[8] J.E. Beasley, N. Meade, and T.-J. Chang. An evolutionary heuristic for the index tracking problem. *European Journal of Operational Research*, 148(3):621–643, 2003.

[9] I.R.C. Buckley and R. Korn. Optimal index tracking under transaction costs and impulse control. *International Journal of Theoretical and Applied Finance*, 1(3):315–330, 1998.

[10] N.A. Canakgoz and J.E. Beasley. Mixed-integer programming approaches for index tracking and enhanced indexation. *European Journal of Operational Research*, 196(1):384–399, 2009.

[11] V. Chankong and Y.Y. Haimes. *Multiobjective Decision Making: Theory and Methodology.* North-Holland, 1983.

[12] S.C. Chiam, K.C. Tan, and A. Al Mamum. Evolutionary multi-objective portfolio optimization in practical context. *International Journal of Automation and Computing*, 5(1):67–80, 2008.

[13] S.C. Chiam, K.C. Tan, and A. Al Mamun. Dynamic index tracking via multi-objective evolutionary algorithm. *Applied Soft Computing*, 13(7):3392–3408, 2013.

[14] C.A. Coello Coello, G.B. Lamont, and D.A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Springer, 2nd edition, 2007.

[15] G. di Tollo and D. Maringer. Metaheuristics for the index tracking problem. In M.J. Geiger, W. Habenicht, M. Sevaux, and K. Sörensen, editors, *Metaheuristics in the Service Industry, Lecture Notes in Economics and Mathematical Systems, Volume 624*, pages 127–154. Springer, 2009.

[16] M. Ehrgott. *Multicriteria Optimization.* Springer, 2nd edition, 2005.

[17] M. Ehrgott, C. Waters, R. Kasimbeyli, and O. Ustun. Multiobjective programming and multiattribute utility functions in portfolio optimization. *INFOR: Information Systems and Operational Research*, 47(1):31–42, 2009.

[18] C. Filippi and E. Stevanato. Approximation schemes for bi-objective combinatorial optimization and their application to the TSP with profits. *Computers & Operations Research*, 40(10):2418–2428, 2013.

[19] Börse Frankfurt. `http://www.boerse-frankfurt.de/en/equities/indices`.

[20] A. Frino, D.R. Gallagher, and T.N. Oetomo. The index tracking strategies of passive and enhanced index equity funds. *Australian Journal of Management*, 30(1):23–55, 2005.

[21] FTSE. `http://www.ftse.com/products/indexmenu`.

[22] L.J. Gitman, M.D. Joehnk, S. Smart, R.H. Juchau, D.G. Ross, and S. Wright. *Fundamentals of Investing*. Pearson Australia, 3rd edition, 2011.

[23] M.J. Gruber. Another puzzle: The growth in actively managed mutual funds. *The Journal of Finance*, 51(3):783–810, 1996.

[24] G. Guastaroba, R. Mansini, and M.G. Speranza. Models and simulations for portfolio rebalancing. *Computational Economics*, 33(3):237–262, 2009.

[25] G. Guastaroba, R. Mansini, and M.G. Speranza. On the effectiveness of scenario generation techniques in single-period portfolio optimization. *European Journal of Operational Research*, 192(2):500–511, 2009.

[26] G. Guastaroba and M.G. Speranza. Kernel search: An application to the index tracking problem. *European Journal of Operational Research*, 217(1):54–68, 2012.

[27] G. Guastaroba and M.G. Speranza. Kernel search for the capacitated facility location problem. *Journal of Heuristics*, 18(6):877–917, 2012.

[28] G. Guastaroba and M.G. Speranza. A heuristic for BILP problems: The single source capacitated facility location problem. *European Journal of Operational Research*, 238(2):438–450, 2014.

[29] M. Hirschberger, R.E. Steuer, S. Utz, M. Wimmer, and Y. Qi. Computing the nondominated surface in tri-criterion portfolio selection. *Operations Research*, 61(1):169–183, 2013.

[30] Nikkei Indexes. `http://indexes.nikkei.co.jp/en/nkave`.

[31] S&P Dow Jones Indices. `http://www.spindices.com/`.

[32] Russell Investments. `http://www.russell.com/indexes/americas/default.page`.

[33] A. Jaszkiewicz. Evaluation of multiple objective metaheuristics. In X. Gandibleux, M. Sevaux, K. Sörensen, and V. T'kindt, editors, *Metaheuristics for Multiobjective Optimisation, Lecture Notes in Economics and Mathematical Systems, Volume 535*, pages 65–89. Springer, 2004.

[34] P. Jorion. Enhanced index funds and tracking error optimization. *Unpublished Paper, Graduate School of Management, University of California at Irvine*, 2002. Currently available at `http://merage.uci.edu/~jorion/papers/enh.pdf`.

[35] T. Koshizuka, H. Konno, and R. Yamamoto. Index-plus-alpha tracking subject to correlation constraint. *International Journal of Optimization: Theory, Methods and Applications*, 1(2):215–224, 2009.

[36] M.A. Lejeune. Game theoretical approach for reliable enhanced indexation. *Decision Analysis*, 9(2):146–155, 2012.

[37] M.A. Lejeune and G. Samatlı-Paç. Construction of risk-averse enhanced index funds. *INFORMS Journal on Computing*, 25(4):701–719, 2013.

[38] Q. Li, L. Sun, and L. Bao. Enhanced index tracking based on multi-objective immune algorithm. *Expert Systems with Applications*, 38(5):6101–6106, 2011.

[39] C.-C. Lin and Y.-T. Liu. Genetic algorithms for portfolio selection problems with minimum transaction lots. *European Journal of Operational Research*, 185(1):393–404, 2008.

[40] D. Maringer and O. Oyewumi. Index tracking with constrained portfolios. *Intelligent Systems in Accounting, Finance and Management*, 15(1-2):57–71, 2007.

[41] H. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

[42] G. Mavrotas and D. Diakoulaki. A branch and bound algorithm for mixed zero-one multiple objective linear programming. *European Journal of Operational Research*, 107(3):530–541, 1998.

[43] G. Mavrotas and D. Diakoulaki. Multi-criteria branch and bound: A vector maximization algorithm for mixed 0-1 multiple objective linear programming. *Applied Mathematics and Computation*, 171(1):53–71, 2005.

[44] K. Metaxiotis and K. Liagkouras. Multiobjective evolutionary algorithms for portfolio management: A comprehensive literature review. *Expert Systems with Applications*, 39(14):11685–11698, 2012.

[45] H. Mezali and J.E. Beasley. Quantile regression for index tracking and enhanced indexation. *Journal of the Operational Research Society*, 64(11):1676–1692, 2013.

[46] H. Mezali and J.E. Beasley. Index tracking with fixed and variable transaction costs. *Optimization Letters*, 8(1):61–80, 2014.

[47] K.M. Miettinen. *Nonlinear multiobjective optimization.* Kluwer Academic, 1999.

[48] G. Mitra, T. Kyriakis, C. Lucas, and M. Pirbhai. A review of portfolio planning: Models and systems. In Satchell S. and Scowcroft A., editors, *Advances in Portfolio Construction and Implementation*, pages 1–39. Butterworth-Heinemann, 2003.

[49] C.B. Oğuzsoy and S. Güven. Robust portfolio planning in the presence of market anomalies. *Omega*, 35(1):1–6, 2007.

[50] Ö. Özpeynirci and M. Köksalan. An exact algorithm for finding extreme supported nondominated points of multiobjective mixed integer programs. *Management Science*, 56(12):2302–2315, 2010.

[51] D. Roman, G. Mitra, and V. Zverovich. Enhanced indexation based on second-order stochastic dominance. *European Journal of Operational Research*, 228(1):273–281, 2013.

[52] S. Ruzika and M.M. Wiecek. Approximation methods in multiobjective programming. *Journal of Optimization Theory and Applications*, 126(3):473–501, 2005.

[53] R. Solanki. Generating the noninferior set in mixed integer biobjective linear programs: An application to a location problem. *Computers & Operations Research*, 18(1):1–15, 1991.

[54] T. Stidsen, K.A. Andersen, and B. Dammann. A branch and bound algorithm for a class of biobjective mixed integer programs. *Management Science*, 60(4):1009–1032, 2014.

[55] C.A. Valle, N. Meade, and J.E. Beasley. Absolute return portfolios. *Omega*, 45(0):20–41, 2014.

[56] T. Vincent, F. Seipp, S. Ruzika, A. Przybylski, and X. Gandibleux. Multiple objective branch and bound for mixed 0-1 linear programming: Corrections and improvements for the biobjective case. *Computers & Operations Research*, 40(1):498–509, 2013.

[57] M. Woodside-Oriakhi, C. Lucas, and J.E. Beasley. Portfolio rebalancing with an investment horizon and transaction costs. *Omega*, 41(2):406–420, 2013.

[58] L.-C. Wu, S.-C. Chou, C.-C. Yang, and C.-S. Ong. Enhanced index investing based on goal programming. *The Journal of Portfolio Management*, 33(3):49–56, 2007.