Ann Oper Res (2009) 166: 57–71
DOI 10.1007/s10479-008-0404-4

A hybrid optimization approach to index tracking

Rubén Ruiz-Torrubiano · Alberto Suárez

Published online: 5 August 2008

Abstract Index tracking consists in reproducing the performance of a stock-market index by investing in a subset of the stocks included in the index. A hybrid strategy that combines an evolutionary algorithm with quadratic programming is designed to solve this NP-hard problem: Given a subset of assets, quadratic programming yields the optimal tracking portfolio that invests only in the selected assets. The combinatorial problem of identifying the appropriate assets is solved by a genetic algorithm that uses the output of the quadratic optimization as fitness function. This hybrid approach allows the identification of quasi-optimal tracking portfolios at a reduced computational cost.

Keywords Data mining · Financial modeling · Asset management

1 Introduction

Index tracking consists in constructing a portfolio whose behavior during a predefined period of time is as similar as possible to that of the index that is being tracked. Such a portfolio is called a *tracking portfolio*. The index tracking problem arises in the context of asset management. A rational investor typically wishes to obtain the highest possible performance assuming as little risk as possible. There could be additional restrictions for investment, which may arise from conditions imposed by the market (minimum investment in a given asset), be the result of a quantitative analysis (e.g. capital concentration constraints from the Black-Litterman model), or reflect expert knowledge and preferences of the investor. This set of market constraints and investor preferences, together with the expected risk and return of the assets determines the strategy that should be used for fund management. There are two types of approaches to this task (Beasley et al. 2003):

R. Ruiz-Torrubiano (✉) · A. Suárez
Computer Science Department, Universidad Autónoma de Madrid, Calle Francisco Tomás y Valiente, 11, 28049 Madrid, Spain
e-mail: ruben.ruiz@uam.es

A. Suárez
e-mail: alberto.suarez@uam.es

(i) *Active fund management*: Fund managers aim to maximize returns by selling or buying equities based on their previous experience and expert knowledge. This kind of management is appropriate for investors who are less risk averse and who wish to maximize the return of their investments.
(ii) *Passive fund management*: Fund managers are required to follow a strict set of constraints to guarantee a minimum level of return. The *index tracking* problem appears when one of these requirements is to obtain a performance that is as close as possible to that of a reference financial index. This kind of management is more appropriate for conservative investors. Such investments are typically less risky and usually yield returns that are close to the market benchmark established by the tracked index.

Broadly speaking, two different strategies can be used to track a market index. *Full replication* consists in purchasing every single stock included in the index. In practice, this strategy is infeasible because of the high transaction costs incurred. An alternative is to find the portfolio that minimizes the tracking error by investing in only a subset of the assets included in the index. This strategy involves much lower transaction costs, and can in principle achieve acceptable tracking errors. In this work we propose a computationally tractable solution for the design of near-optimal replication strategies in which the investor limits the number of assets that are used to track the reference index.

The tracking performance of a portfolio is measured by the *tracking error*. In the literature several measures of this error have been proposed (Ammann and Zimmermann 2001; Lobo et al. 2007; Shapcott 1992; Beasley et al. 2003; Buckley and Korn 1998; Rudolf et al. 1999). Most of them are based either on correlations between the tracking portfolio and the index returns or on estimates of the variance of the difference between the returns of the index and the returns of the tracking portfolio (Markowitz 1987; Buckley and Korn 1998; Shapcott 1992). However, measures based on the variance of the tracking deviations are flawed. As noted in Beasley et al. (2003), if the difference between the returns of the index and those of the tracking portfolio is constant, then the tracking error would be zero. This is an undesirable result because it does not take into account the tracking bias. In the current investigation the mean squared error for the returns is used as the measure of disagreement between the tracking portfolio and the index which is being tracked. This definition of the tracking error has the advantage of being quadratic. Furthermore it takes into account the bias of the tracking portfolio, so that constant differences are also penalized (Ammann and Zimmermann 2001; Beasley et al. 2003; Gilli and Këllezi 2001).

In the present work, the problem of replicating a financial index using only a subset of the index assets is addressed using a hybrid evolutionary approach. The problem of selecting the optimal subset of assets to be included in the tracking portfolio and the problem of determining the optimal asset weights are handled separately. A genetic algorithm (GA) is used to identify the optimal subset of assets. Each chromosome in the population corresponds to a portfolio that invests only in a subset of assets with the specified cardinality. The error of the optimal tracking portfolio that can be constructed by investing only in the selected assets (as determined by the chromosome) is used as fitness function in the GA. In this manner, the combinatorial search efficacy of the genetic algorithm is combined with the efficiency of quadratic programming to obtain near-optimal solutions to the index-tracking problem.

Most previous work on index tracking focuses on finding the portfolio that is optimal using as inputs the recent historical evolution of the assets. Since we are interested in the future tracking performance of the portfolio, it is necessary to make the assumption that recent historic performance is a good predictor of the performance in the near future. In this investigation, the validity of this assumption is tested estimating both in-sample and out-of sample performance. Using the language of machine learning the data is partitioned into training data and testing data. The training data is used to construct the optimal tracking portfolio investing in a subset of the index assets. The performance of this tracking portfolio is then evaluated not only on the data used in the optimization (in-sample performance), but also on an independent test set, which is not used in the optimization (out-of-sample performance). It is seen that portfolios that are optimal with respect to the training data need not be optimal on the test set. Conversely, portfolios that are suboptimal on the training data can have a better out-of-sample (generalization) performance (i.e. a lower tracking error) on the test set.

The article is structured as follows. In Sect. 2, the index tracking problem is introduced. Section 3 reviews earlier work on this topic. In Sect. 4 a hybrid optimization strategy that combines quadratic programming and a genetic algorithm is proposed. Section 5 presents the results of experiments in which the hybrid optimization strategy is used to solve several benchmark problems. Finally, Sect. 6 summarizes the conclusions of this work and proposes some directions of future research. A proof of the NP-completeness of the index tracking problem with cardinality constraints is given in a technical appendix.

2 Index tracking

Let $\{S_i(t)\}_{i=1}^N$ be the time series of asset prices for the N assets that are included in the market index whose evolution we wish to replicate. Let $I(t)$ be the time series of this index. All time series are defined for equally spaced intervals $t = 1, 2, \dots, T$. Time is measured in units of Δt. The series of relative returns $\{r_i(t)\}_{i=1}^N$ for the assets, and $r_I(t)$ for the index are defined as

$$r_i(t) = \frac{S_i(t+1) - S_i(t)}{S_i(t)}; \quad r_I(t) = \frac{I(t+1) - I(t)}{I(t)}; \quad t = 1, 2, \dots, T \tag{1}$$

The tracking portfolio invests in assets from the set ξ, which is a subset of the assets included in the index. Assuming that the value of the portfolio at time t is

$$P(t) = \sum_{i \in \xi} c_i(t) S_i(t) \tag{2}$$

and that the amounts invested in each of the assets $\{c_i(t)\}_{i \in \xi}$ are constant in the interval $[t, t+1)$, the return of the tracking portfolio in that period is

$$r_P(t) = \frac{P(t+1) - P(t)}{P(t)} = \sum_{i \in \xi} w_i(t) r_i(t); \quad w_i(t) = \frac{c_i(t) S_i(t)}{\sum_{j \in \xi} c_j(t) S_j(t)} \tag{3}$$

The tracking error during the period $[1, \dots, T]$ is defined as the mean squared deviation between the series of returns of the tracking portfolio, $r_P(t)$, and the index returns, $r_I(t)$

$$\mathrm{MSE}(\mathbf{r_P}, \mathbf{r_I}) = \frac{1}{T} \sum_{t=1}^{T} \big(r_P(t) - r_I(t)\big)^2 = \frac{1}{T} \sum_{t=1}^{T} \Big(\sum_{i \in \xi} w_i(t) r_i(t) - r_I(t) \Big)^2 \tag{4}$$

Expression (4) is the objective function that needs to be minimized to solve the index tracking problem. One of the main advantages of using this measure for the tracking error is that the objective function is quadratic in the weights of the assets, so that standard and very efficient quadratic programming algorithms can be used to minimize it. If in (1) logarithmic returns were used instead of discrete-time returns the objective function would not be quadratic. A further advantage of (4) is that it avoids the problems of measures based only on the variance of the deviation, which become zero when the differences between the returns of the index and of the tracking portfolio are constant over time (Ammann and Zimmermann 2001; Shapcott 1992; Buckley and Korn 1998; Gilli and Këllezi 2001).

A possible strategy that can be used to replicate a financial index is to determine an investment in each asset $\{c_i\}_{i\in\xi}$, and to maintain these values constant throughout the tracking period (*buy-and-hold*). Because of the changes in the market prices of the assets $\{S_i(t)\}_{i\in\xi}$ the portfolio weights evolve according to the formula

$$w_i(t) = \frac{c_i S_i(t)}{\sum_{j\in\xi} c_j S_j(t)}, \quad i \in \xi \tag{5}$$

Note that this strategy has low transaction costs because, after the initial investment in each asset, no reallocation of capital is needed. However, it has the inconvenience that obtaining the values $\{c_i\}_{i\in\xi}$ that minimize (4) is a non-linear optimization problem. A second strategy is to hold the weights of the assets in the portfolio constant (Ammann and Zimmermann 2001). This approach requires actively managing the portfolio: The absolute investments in each asset, $\{c_i(t)\}_{i\in\xi}$ needs to be adjusted at every time step to maintain the weights of the portfolio constant

$$w_i = \frac{c_i(t) S_i(t)}{\sum_{j\in\xi} c_j(t) S_j(t)}, \quad i \in \xi \tag{6}$$

The advantage of this approach is that, once the assets to be included in the tracking portfolio are selected, it is possible to solve the index tracking problem efficiently and in an exact manner by quadratic programming. In particular the constant weights $\{w_i\}_{i\in\xi}$ that minimize (4) can be found by solving the constrained mixed-integer quadratic optimization problem

$$\mathrm{Min}_{\mathbf{w}}\left[\frac{1}{2}\mathbf{w}'\cdot\mathbf{H}\cdot\mathbf{w} - \mathbf{g}'\cdot\mathbf{w}\right] \tag{7}$$

with the definitions

$$\begin{aligned} H_{ij} &= \frac{1}{T}\sum_t r_i(t) r_j(t); \quad i,j = 1,2,\dots,N \\ g_i &= \frac{1}{T}\sum_t r_i(t) r_I(t); \quad i = 1,2,\dots,N, \end{aligned} \tag{8}$$

subject to the constraints

$$\sum_{i=1}^{N} w_i = 1 \tag{9}$$

$$w_i \geqslant 0, \quad \forall i = 1,\dots,N \tag{10}$$

$$\mathbf{l} \leqslant \mathbf{A}\cdot\mathbf{w} \leqslant \mathbf{u} \tag{11}$$

$$\sum_{i=1}^{N} z_i \leqslant c \tag{12}$$

$$a_i z_i \leqslant w_i \leqslant b_i z_i, \quad a_i, b_i \geqslant 0,\ \forall i = 1,\dots,N \tag{13}$$

where $\mathbf{w}$ is the $N \times 1$ column vector of asset weights. The auxiliary binary variables $\mathbf{z} = \{z_i\}_{i=1}^N$ have been included to take into account the discrete nature of the asset selection problem. In particular, z_i has the value 1 if asset i is included in the portfolio ($i \in \xi$) and 0 otherwise ($i \notin \xi$). With this definition, the return of the portfolio in the interval $[t, t+1)$ is

$$r_P(t) \equiv r_P(t; \mathbf{z}) = \sum_{i=1}^N z_i w_i r_i(t) \tag{14}$$

Equation (9) is a *budget constraint*. It ensures that 100% of the capital is invested in the portfolio. The *non-negativity constraint*, (10), indicates that no short selling is allowed. Equation (11) expresses the *capital concentration constraints* in matrix form: $\mathbf{A}$ is an $M \times N$ matrix whose rows are N-dimensional vectors containing the coefficients of the M linear restrictions. The $M \times 1$ column vectors $\mathbf{l}$, $\mathbf{u}$ define, respectively, the lower and upper bounds of these restrictions. For example, one such restriction could read $0.1 \leqslant w_1 + 2w_3 + w_4 \leqslant 0.4$, which ensures that no less than 10% and no more than 40% of the capital is invested in the linear combination of the assets labeled 1, 3 and 4. The *cardinality constraint* is given by (12). This inequality prescribes that no more than c assets can be included in the portfolio. In the experiments performed, the index tracking problem with the inequality $\sum_{i=1}^N z_i \leqslant c$, is solved by finding the optimal among the solutions of a collection of c problems with different equality constraints $\sum_{i=1}^N z_i = k$, $k = 1, 2, \dots, c$. Finally, (13) reflects floor and ceiling constraints, which set upper and lower bounds on the fraction of capital that can be allocated to each asset. Note that if asset i is not included in the portfolio then $z_i = 0$ and, consequently, by (13), $w_i = 0$.

In this work, we consider the problem of finding an optimal tracking portfolio for a time period $[1, \dots, T]$, which generally corresponds to the recent past, where the evolution of the assets is known. The ultimate goal is to construct portfolios whose tracking performance in the period $[T+1, \dots, T+L]$ is also close to optimal. However, the evolution of the assets in this period is not known. Therefore, it is necessary to make the assumption that portfolios that are optimal during the initial time period $[1, \dots, T]$ will also perform well given a future horizon L. Provided that $T + L$ is not too far in the future it is reasonable to assume that the statistical properties of the asset returns in the near future are similar to those in the recent past. As $L \to \infty$, the validity of this hypothesis becomes more and more questionable and the tracking performance of the portfolio, which is guaranteed to be optimal only in the initial period, typically deteriorates. Borrowing the terms used in machine learning, we refer to the initial time period $[1, \dots, T]$ as the *training set*, and to the time period $[T+1, \dots, T+L]$ as the *test set*. In the experiments carried out we measure the tracking performance of the selected portfolios both in the training set (in-sample performance) and in the test set (out-of-sample performance).

3 Previous work

The problem of using a small portfolio to track an index has been addressed by numerous authors. Markowitz (1987) reformulates the problem as a mean-variance optimization making some assumptions on the statistical properties of the returns of the index assets. This work does not consider the cardinality constraints that are the main concern of the current investigation. Shapcott (1992) proposes to handle the problem of selecting the optimal subset of assets and the quadratic optimization problem that results once this subset is selected separately. The resulting hybrid evolutionary algorithm is similar to the one investigated in this work: both approaches use the Random Assorting Recombination (RAR) algorithm to perform set recombination, but the objective function minimized by Shapcott is the variance of the difference between the index returns and the tracking portfolio returns. By contrast, the tracking error considered in the present work is the mean squared error (MSE) between those returns. Shapcott mainly focuses in comparing a random search algorithm with a genetic algorithm with and without migrations in a multiprocessor environment, and does not take into account other realistic constraints in the model like minimum and maximum investments in an asset or linear combination of assets. Buckley and Korn (1998) apply optimal impulse control techniques to the index tracking problem with fixed and proportional transaction costs. They also give a proof for the existence of an optimal strategy, which includes the possibility of holding a non-zero amount of cash at all times. They focus on a continuous-time formulation of the problem and consider a diffusion process to model the random cash flows to and from the portfolio. Alexander (1999) proposes the construction of tracking portfolios by analyzing the cointegration structure between the time series of each of the assets and the time series of the tracked index. Ammann and Zimmermann (2001) investigate the relationship between several statistical measures of tracking error (based on correlations between the index and the tracking portfolio or on first and second moments of the tracking deviations) and asset allocation restrictions based on admissible weight ranges. They conclude that the tracking error can be quite small even with fairly large admissible weight ranges. Gilli and Këllezi (2001) propose the use of the threshold accepting (TA) heuristic to solve the problem, including cardinality restrictions and transaction costs. The TA heuristic is a deterministic analog of simulated annealing, where transitions are rejected only when they lead to a deterioration in performance that is above a given threshold. Initially the threshold for rejecting the solution is large. Its value is then gradually decreased until only candidate solutions that improve the performance are accepted. Eventually the algorithm converges to an optimum (possibly a local one). Beasley et al. (2003) address the index tracking problem using evolutionary heuristics with real-valued chromosome representations. As in the current work, the root mean squared error is used as a measure of tracking error. Their investigation is focused on the influence of transaction costs and portfolio rebalancing. Lobo et al. (2007) investigate the portfolio optimization problem with transaction costs, which they address by means of a heuristic relaxation method that consists in solving a small number of convex optimization problems using fixed transaction costs. Then they apply their results to the index tracking problem defining tracking error as the expected square differences of the returns.

Not all work on index tracking uses quadratic objective functions. Rudolf et al. (1999) argue that absolute deviations are more convenient and easier to interpret from a practitioner's point of view than square deviations. They then propose several piecewise linear measures of the tracking error, and solve the problem by means of linear programming. Consiglio and Zenios (2001) also use the linear absolute deviation to quantify the tracking error. In their model, which is used to track a composite government bond index, the decisions about asset allocation among different markets and bond-picking decisions are integrated.

4 A hybrid optimization strategy for index tracking

There exist very efficient algorithms to minimize quadratic objective functions with linear constraints. Unfortunately, not all constraints in the formulation of the index tracking problem given by (9)–(13) are linear. In particular, the restriction on the cardinality of the portfolio (12) is discrete and therefore highly non-linear. If the values of the variables $\mathbf{z} = \{z_i\}_{i=1}^N$ were fixed the problem would be solvable by quadratic programming. Therefore, a possible approach to the index tracking problem is to handle separately the combinatorial optimization problem of selecting the values of the binary variables $\mathbf{z}$ and the quadratic optimization problem that consists in finding the asset weights $\{w_i(\mathbf{z})\}_{i=1}^N$ for a fixed value of $\mathbf{z}$. The combinatorial search is thus guided by the quadratic optimization task. The objective function for each candidate $\mathbf{z}$ is the value of the minimum obtained by solving the quadratic problem.

A naïve approach to this optimization schedule is to apply exhaustive search on the combinatorial part of the problem: For every possible combination of the binary variables, calculate the value of the objective function by means of quadratic optimization. In small and intermediate problems (see, for example, the Hang Seng index problem in the next section) it is possible to obtain a solution with a considerable but realizable computational effort. In larger problems, the complexity of the search space becomes intractable. For instance, in a problem with $N = 100$ assets in the universe and a cardinality constraint $c = 10$, $\sum_{i=1}^{10} \binom{100}{i}$ quadratic optimizations need to be performed. Assuming that each quadratic optimization takes 1 ms, the algorithm would need more than 500 years to find the optimal solution.

A directed search is a more suitable approach for this problem. Several heuristics can be used to explore the combinatorial search space. In this work, we propose a *hybrid genetic algorithm* to solve the problem. Genetic algorithms (Holland 1975) are optimization heuristics which are inspired by the process of the evolution of natural species. The pseudocode for a basic genetic algorithm is the following:

```
1. Generate population of $P$ individuals (candidate solutions).
2. While convergence criteria are not met:
   a. Select a parent set composed of $n_P$ individuals from the population.
   b. While parent set is not empty
        i. Select two individuals.
       ii. Apply crossover to the pair and generate $n_C$ children.
      iii. Apply mutation to the $n_C$ children.
   c. Add to the population the new $n_P n_C/2$ individuals generated.
   d. Select again $P$ individuals and create the population for the next generation.
```

Each individual is assigned a *fitness* value in the population by a function that measures the quality of the solution encoded by the individual. The choice of the fitness function is crucial in the design of a genetic algorithm. In this work we have chosen the function $F(\mathbf{z}) = -\mathrm{MSE}(\mathbf{r_P}(\mathbf{z}), \mathbf{r_I})$, which is calculated by means of quadratic optimization according to the scheme previously outlined.

The representation of each individual (chromosome) is called the *genotype*. Finding an appropriate genotype representation for the individuals in the population is also an important factor in the success of the genetic algorithm. A common choice of chromosome representation in the GA literature is a binary representation. In this encoding, a candidate solution is represented by a binary string of length N. For the index tracking problem, a value of 1 in the i-th position of the string means that asset i is present in the portfolio, whereas a 0 value in that position indicates that asset i is excluded from the investment. For example, in a problem with $N = 5$ and $c = 3$, the individual

Table 1 Example of uniform crossover

Parent 1	**1**10**1**1**0**0**1**001**10**1**11**
Parent 2	0**10**1**1**1**1**0**101**10**1**10
Child	**1101101110110111**

Table 2 Examples of bitwise mutation

Child 1	11**0**11110**1**0010110
Mutated Child 1	11**1**11110**0**0010110
Child 2	01001**0**0100110**1**11
Mutated Child 2	01001**1**0100110**0**11

1	2	3	4	5
0	1	1	0	1

represents the optimal tracking portfolio that can be constructed by investing in the assets labeled 2, 3 and 5 only. The fitness value of such an individual is minus the tracking-error of this optimal portfolio. The permutation of any two bits with different values can be used as a mutation operator that preserves the cardinality of the individuals. For instance, from the individual (01101), a permutation that interchanges the bits in positions 1 and 3 generates the individual (**110**01), which represents the optimal tracking portfolio that can be constructed by investing in assets 1, 2 and 5. Uniform crossover is used to avoid biases from the original labeling of the assets. Examples of these crossover and mutation operators are displayed in Tables 1 and 2. The uniform crossover operator has the drawback that it can generate individuals that fail to satisfy the cardinality constraint. For instance, Table 1 shows an example in which both parents are constrained to have 10 assets, but the resulting child is composed of 12 assets. It is then possible to use either chromosome repair heuristics that restore the constraint, or penalty functions that reduce the fitness of individuals violating the cardinality constraint. Exploratory experiments show that the stochastic search process using this binary representation and penalty functions (either linear, quadratic or logarithmic) is not effective and tends to converge to suboptimal solutions. For this reason, an alternative chromosome representation with especially designed mutation and recombination operators that preserve the cardinality of the individuals is used in this work.

This alternative representation, where the chromosomes are subsets of assets, has been successfully employed in the solution of the closely related problem of optimizing a portfolio with cardinality constrains (Moral-Escudero et al. 2006). Index tracking and portfolio optimization with cardinality constraints are characterized by different objective functions and restrictions. Nonetheless, they are both mixed-integer quadratic problems and can be approached using similar techniques. In the set representation considered in (Moral-Escudero et al. 2006) a candidate solution is encoded as the set containing the labels of the assets that are included in the portfolio. For example, a portfolio containing the assets 2, 3 and 5 would be represented by the set $\{2, 3, 5\}$. A mutation operator that preserves the cardinality of the individual is implemented by swapping one product which is in the set with another product which does not belong to the set. Examples of crossover operators that preserve cardinality are the Random Respectful Recombination (R3) and Random Assorting Recombination (RAR) (Radcliffe 1992) crossover operators. Moral-Escudero et al. (2006) show that R3 has a tendency to overexploit the information available in the parent chromosomes. This property often leads to premature convergence. For this reason RAR is used in our work. The implementation of the RAR crossover operator involves the following operations:

```
1. Create auxiliary sets ($A, B, C, D, E$), where the initial compositions of the sets is
      $A$: Assets present in both parents.
      $B$: Assets not present in any of the parents.
      $C \equiv D$: Assets only present in one parent.
      $E$: Empty set.
2. Build set $G$
      $w$ copies of elements from $A$ and $B$.
      1 copy of elements from $C$ and $D$.
3. Build the child chromosome
   3.1. Repeat
           Extract an element from $G$ (without replacement).
         • If the element comes from $A$ or $C$ and it is not an element of $E$, include it in the
           child chromosome.
         • If the element comes from $B$ or $D$, include it in the set $E$.
           Until chromosome is complete or set $G$ is empty.
   3.2. If chromosome is not complete include assets not yet included at random.
```

The advantage of the RAR operator over other set recombination operators is that the hyperparameter w can be used to regulate the amount of common information from the parents that is retained by the children (i.e., the importance that is given to the common information from the parents). As w increases, the RAR operator exploits more information common to both parents, asymptotically approaching the R3 operator (Radcliffe 1992). A small value of w implies that little information from both parents is retained in the children. Therefore, the probability of selecting an asset that is present in only one of the parents is increased. Another interesting property of this operator is that its application can produce implicit mutations (by step 3.2), and include in the child assets not present in any of the parents.

The GA implemented uses RAR recombination with $w = 1$. The generational substitution scheme is a steady state model. In each epoch of the steady state model, a new child replaces the worst-fitness individual in the previous generation. The parent selection strategy is a binary tournament. These strategy choices result in a GA with high evolutionary pressure, which has shown good performance in the portfolio optimization problem (Moral-Escudero et al. 2006).

5 Experimental results

To assess the performance of the hybrid optimization algorithm several experiments on benchmark problems from the OR-Library (Beasley 1990) are carried out. The OR-library is a publicly available collection of test data sets for a variety of Operations Research (OR) problems. In particular, for the index tracking problem (Beasley et al. 2003) it contains the weekly stock prices for the period 1992 to 1997 of the assets included in major world market indexes, such as Hang Seng (Hong Kong), DAX (Germany), FTSE (Great Britain), Standard and Poor's (U.S.A.) and the Nikkei index (Japan).

Before addressing the index tracking problem with actual market data, the quality of the solutions obtained by the optimization algorithm is tested using an artificial benchmark problem: A synthetic index is generated by investing equal amounts in the last c stocks of each of the market indexes. Then, the hybrid optimization algorithm introduced in Sect. 4 is used to find the tracking portfolio with a cardinality constraint c. In this problem the optimal solution is known: it is a portfolio that invests only in the last c stocks of the index. Table 3

Table 3 Results of the hybrid GA approach to the selection of portfolios that track synthetic indexes

Indexes	Card. const.	MSE (Train)	MSE (Opt)	Norm. MSE	Mean eval.	Time (s)	% Opt
Hang Seng	5	9.203e-6	9.203e-6	8.396e-3	30783	0.48	100.0
($N = 31$)	6	5.465e-6	5.465e-6	4.986e-3	36058	0.53	100.0
	7	5.043e-6	5.043e-6	4.601e-3	41103	0.58	100.0
	8	4.200e-6	4.200e-6	3.832e-3	46370	0.63	100.0
	9	4.336e-6	4.336e-6	3.956e-3	51653	0.69	100.0
	10	4.226e-6	4.226e-6	3.856e-3	56564	0.75	100.0
DAX ($N = 85$)	5	3.406e-6	3.406e-6	8.297e-3	56329	2.12	100.0
	6	3.314e-6	3.314e-6	8.074e-3	66202	2.22	100.0
	7	3.225e-6	–	7.857e-3	75849	2.44	100.0
	8	6.293e-6	–	1.533e-2	85933	2.60	100.0
	9	6.040e-6	–	1.471e-2	95483	2.79	100.0
	10	5.276e-6	–	1.285e-2	105604	2.96	100.0
FTSE ($N = 89$)	5	2.919e-6	2.919e-6	9.626e-3	366065	19.41	100.0
	6	3.108e-6	3.108e-6	1.025e-2	427199	21.65	100.0
	7	2.579e-6	–	8.505e-3	488560	22.86	100.0
	8	2.085e-6	–	6.876e-3	550436	23.92	100.0
	9	2.296e-6	–	7.573e-3	611855	24.53	100.0
	10	2.111e-6	–	6.961e-3	672879	25.59	100.0
S&P ($N = 98$)	5	1.215e-5	1.215e-6	5.252e-2	486830	25.15	100.0
	6	1.057e-5	–	4.555e-2	567589	27.01	100.0
	7	7.653e-6	–	3.297e-2	648999	27.86	100.0
	8	4.534e-6	–	1.954e-2	731965	30.20	100.0
	9	3.274e-6	–	1.411e-2	815576	28.67	100.0
	10	4.835e-6	–	2.083e-3	895356	29.86	100.0
Nikkei ($N = 225$)	5	2.588e-6	–	3.175e-3	904391	31.58	100.0
	6	2.416e-6	–	2.964e-3	1057976	35.12	100.0
	7	2.364-6	–	2.901e-3	1208759	35.37	100.0
	8	1.925e-6	–	2.362e-3	1362428	38.29	100.0
	9	1.812e-6	–	2.223e-3	1514054	42.41	100.0
	10	1.722e-6	–	2.113e-3	1667507	42.82	100.0

summarizes the results for this synthetic problem. Results are averaged over 30 executions with different initial random populations. The optimal mean squared error (4th column) is calculated by exhaustive search whenever the computation is feasible. The tracking errors obtained are non zero because the tracking portfolio has constant weights, whereas the synthetic index is constructed using constant coefficients. Since the prices of different assets may evolve differently, the investment weights in the index vary due to market capitalization. The fifth column displays the tracking error normalized by the variance of the index returns in the period considered. The 6th and 7th columns report the average number of evaluations of the objective function per execution and the average CPU time per execution on an AMD 64-bit dual-core processor 2.01 GHz with 2 GB RAM, respectively. The last column represents the percentage of the 30 executions in which the best solution is reached. In

Table 4 Results of the hybrid GA approach to the selection of portfolios that track market indexes

Indexes	Card. const.	MSE (Train)	MSE Train (Opt)	Norm. MSE (Train)	MSE (Test)	Norm. MSE (Test)	Mean eval.	Time (s)	% Opt
Hang Seng	5	4.135e-5	4.135e-5	2.962e-2	7.218e-5	9.157e-2	30709	0.42	100.0
($N = 31$)	6	3.031e-5	3.031e-5	2.171e-2	4.755e-5	6.031e-2	35796	0.47	100.0
	7	2.371e-5	2.371e-5	1.699e-2	3.810e-5	4.832e-2	40951	0.52	100.0
	8	1.907e-5	1.907e-5	1.366e-2	2.899e-5	3.677e-2	46093	0.59	100.0
	9	1.622e-5	1.622e-5	1.161e-2	2.581e-5	3.274e-2	51288	0.64	100.0
	10	1.346e-5	1.346e-5	9.643e-3	2.057e-5	2.610e-2	56520	0.68	100.0
DAX	5	2.211e-5	2.211e-5	5.699e-2	1.018e-4	2.402e-1	56269	2.10	100.0
($N = 85$)	6	1.764e-5	1.764e-5	4.544e-2	8.938e-5	2.109e-1	65735	2.27	100.0
	7	1.371e-5	–	3.533e-2	8.459e-5	1.996e-1	75260	2.27	100.0
	8	1.110e-5	–	2.860e-2	7.928e-5	1.871e-1	84842	2.43	100.0
	9	9.216e-5	–	2.375e-2	7.776e-5	1.835e-1	94531	2.51	100.0
	10	8.084e-5	–	2.083e-2	7.482e-5	1.766e-1	104355	2.74	100.0
FTSE	5	6.417e-5	6.417e-5	1.721e-1	1.581e-4	6.922e-1	366296	29.73	90.0
($N = 89$)	6	4.961e-5	4.961e-5	1.331e-1	1.119e-4	4.899e-1	426478	30.53	93.3
	7	3.828e-5	–	1.027e-1	9.069e-5	3.970e-1	488776	30.80	90.0
	8	2.903e-5	–	7.788e-2	9.662e-5	4.229e-1	550079	31.27	96.7
	9	2.486e-5	–	6.669e-2	8.592e-5	3.761e-1	611690	28.36	100.0
	10	2.184e-5	–	5.858e-2	8.009e-5	3.506e-1	673641	32.08	100.0
S&P	5	4.497e-5	4.497e-5	2.987e-1	1.142e-4	3.830e-1	485106	25.79	100.0
($N = 98$)	6	3.373e-5	–	2.241e-1	1.007e-4	3.377e-1	566475	25.85	100.0
	7	2.761e-5	–	1.834e-1	7.798e-5	2.615e-1	647951	27.19	100.0
	8	2.274e-5	–	1.510e-1	6.764e-5	2.268e-1	729664	29.72	100.0
	9	1.939e-5	–	1.288e-1	5.910e-5	1.982e-1	811392	30.81	90.0
	10	1.657e-5	–	1.101e-1	5.546e-5	1.859e-1	893635	34.49	93.3
Nikkei	5	5.456e-5	–	6.254e-2	1.629e-4	2.151e-1	903844	42.09	90.0
($N = 225$)	6	4.008e-5	–	4.595e-2	1.468e-4	1.938e-1	1054724	44.83	53.3
	7	3.356e-5	–	3.846e-2	1.324e-4	1.748e-1	1206342	44.90	16.6
	8	2.601e-5	–	2.981e-2	1.104e-4	1.457e-1	1358302	47.97	6.6
	9	2.125e-5	–	2.436e-2	9.803e-5	1.294e-1	1510641	48.69	3.3
	10	1.797e-5	–	2.060e-2	6.471e-5	8.543e-2	1664071	51.16	6.6

all cases where the calculation was feasible, the solution obtained by the algorithm is found to be the optimal solution. For this synthetic problem, the optimum was reached in every one of the 30 executions performed. These results confirm the validity of the hybrid strategy adopted to solve the mixed-integer quadratic programming problem of tracking a synthetic index. We now apply this strategy to track actual market indexes.

In the experiments carried out with real-world indexes, the tracking problem is treated as an inductive learning task: The data of weekly returns of the stocks included in the index are partitioned into a training set containing the first half of the data (145 values) and a test set with the rest of the data (145 values). The training data are used to find the assets that

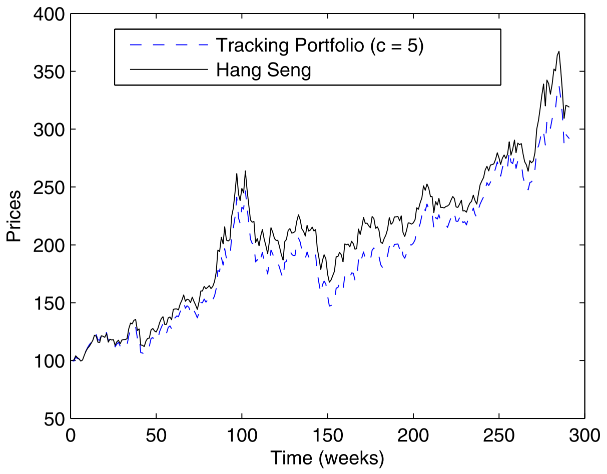

Fig. 1 Normalized time series evolution for the Hang Seng index and the tracking portfolio that invests in only 5 assets

make up the resulting portfolio. The test set is used to estimate the out-of-sample performance of the selected portfolio. As noted in Sect. 1, the underlying assumption is that a portfolio that is optimal during an initial interval in the recent past should also be close to optimal in the near future. Figure 1 represents the evolution of the normalized time series for both the Hang Seng index and the best-known tracking portfolio that invests in only 5 of the 31 assets that are included in the index. Results for all the indexes are summarized in Table 4. There are two extra columns with respect to Table 3: the mean squared error in the test set (6th column) and the value of this error normalized by the variance of the returns (7th column). In all cases investigated where the optimal portfolio can be found by exhaustive search the hybrid optimization algorithm consistently finds the optimal solution. In this kind of problem, the mean squared error over the train set *must* decrease when the number of assets in the cardinality constraint increases, because the more assets are included in the portfolio, the better the index can be tracked. This is not necessarily the case in the problem with synthetic indexes, where for every value of the cardinality constraint a different problem is being solved (a different synthetic index is being tracked). The values for the mean squared error in the train and test sets are similar. In all the cases investigated, the tracking error in the test set is higher than the corresponding training tracking error. Note also that the mean squared error in the test set does not have to decrease as the cardinality constraint increases. Notwithstanding, it does become smaller in the vast majority of cases. This can be taken as an *a posteriori* confirmation of the validity of assuming that the performance on the training data is a good predictor of the generalization performance on the test data.

Taking into account the complexity of the search the execution times are fairly low. Furthermore, the increase of the execution time with the size of cardinality constraint seems to be sub-linear, which is much slower than the increase in the number of candidate solutions for small values of c.

In most of the problems investigated (Hang Seng, DAX, FTSE and S&P), the fraction of executions in which the best-known solution is found is above 90%. By contrast, the Nikkei index problem, composed of 225 assets, seems to be much more difficult. The fraction of successful searches diminishes dramatically as the size of the cardinality constraint increases. To shed some light on the origin of the difficulties in tracking the Nikkei index, we have performed a detailed analysis of the solutions obtained in this problem for $c = 5$.

Table 5 Composition of the optimal and suboptimal portfolios for the cardinality constraint $c = 5$ in the Nikkei index problem

Optimal porfolio	Asset	103	198	153	132	147
	Weight	0.282841	0.264765	0.171233	0.165089	0.116071
Suboptimal portfolio	Asset	103	123	75	153	35
	Weight	0.292244	0.265793	0.154440	0.152062	0.135462

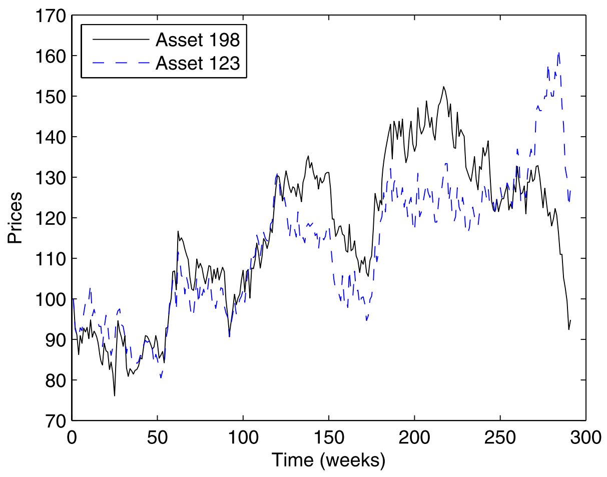

Fig. 2 Normalized time series for assets 198 and 123 in the Nikkei index problem

The composition and the weights of each asset in a best-known solution and in a near-best-known-solution portfolio are shown in Table 5.

The assets in Table 5 are sorted according to their weights in the corresponding portfolios. The two portfolios have only two assets in common, the ones labeled 103 and 153. For the remaining assets, we compare the time series and weekly return series for pairs of assets with similar weights, one from the best-known solution portfolio and one from the near best-known one. Therefore, the pairs of assets compared are 198 and 123, 132 and 75, and 147 and 35. Figure 2 displays in the same graph the price series for the assets labeled 198 and 123. Figure 3 shows a linear regression between the series of returns for both assets. These two series are highly correlated: the coefficient of correlation between the series of weekly returns for both assets is $\rho = 0.6975$. The correlation is also high for the other pairs of assets: In the case of assets 132 and 75, the correlation coefficient is $\rho = 0.6491$, and for the assets 147 and 35, $\rho = 0.6672$.

Therefore, in the near best-known solution, the asset in the best-known solution is replaced by an asset whose time series is highly correlated with it. In the case of a cardinality constraint $c = 5$, the correspondence of correlated series is simple, a one-to-one correspondence. In portfolios containing more assets, more complex linear combinations appear, making the analysis more difficult. When the universe of the problem includes a large number of assets the probability of finding these approximate degeneracies becomes higher.

6 Conclusions

In this work, a hybrid optimization approach is proposed for index tracking. The combinatorial optimization problem of selecting the optimal subset of assets that should be included

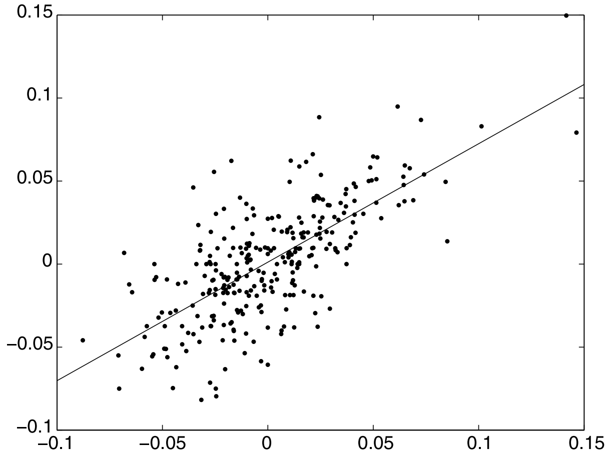

Fig. 3 Weekly returns of asset 198 against those of asset 123 in the Nikkei index problem. The correlation coefficient is $\rho = 0.6975$

in the tracking portfolio is solved by a genetic algorithm. In this GA each individual in the evolving population corresponds to a subset of assets with the specified cardinality. The fitness value of an individual is minus the tracking error of the optimal portfolio that invests only in the assets specified by the individual's chromosome. Since the problem of finding the optimal tracking portfolio that invests only in a specified subset of assets is a quadratic programming problem, the fitness function can be efficiently computed with a quadratic solver. A set representation together with especially designed crossover and mutation operators that preserve the cardinality of the individuals is found to be an appropriate encoding for this problem. This hybrid evolutionary approach yields good results and achieves near optimal solutions in both synthetic and real world problems. However, the presence of highly correlated series when the number of assets in the universe of the problem is high makes the convergence of the algorithm difficult.

Using a similar decomposition for the optimization problem, other heuristics, such as simulated annealing, or ant-colony optimization, can be used to address the combinatorial part of the problem. In this problem we have not taken into account transaction costs: Further research is needed to take into account these costs to design realistic index tracking strategies.

Acknowledgements This research was supported by *Ministerio de Educación y Ciencia* (Spain), project TIN2004-07676-C02-02. Ruben Ruiz-Torrubiano acknowledges the *Universidad Autónoma de Madrid* for financial support under an FPU grant.

Appendix: Proof that the index tracking problem with cardinality constraint is NP-hard

In this appendix we prove that the index tracking problem with cardinality and linear constraints as stated in (9) to (13) is an NP-hard problem. The asset selection problem that needs to be solved in index tracking is closely related to the *Subset Sum* problem, which is known to be NP-complete. The Subset Sum problem consists in extracting from a given set of integers $S = \{s_1, \dots, s_N\}$ a subset of elements whose sum is equal to zero. Suppose that an algorithm **A** that solves the index tracking problem in polynomial time exists. Then, the Subset Sum problem would also be solvable in polynomial time by solving a collection of N

index tracking problems with cardinality constraints $c = 1, \ldots, N$. In these index tracking problems $T = 1$, the return of the tracked index is $r_I(t) = 0$, $\forall t$ and the asset returns are $r_i(t) = s_i$, $\forall t$. The floor and ceiling constraints are $a_i = b_i = 1/c$. Therefore,

$$\mathrm{MSE}(\mathbf{r_P}, \mathbf{r_I}) = \frac{1}{T}\sum_{t=1}^{T}\big(r_P(t) - r_I(t)\big)^2 = \left(\frac{1}{c}\sum_{i=1}^{N} z_i s_i\right)^2 \tag{15}$$

And the problem translates into:

$$\begin{aligned} &\min \left(\frac{1}{c}\sum_{i=1}^{N} z_i s_i\right)^2 \\ &\text{s.t.} \quad \sum_{i=1}^{N} z_i = c \end{aligned} \tag{16}$$

If for some value of c the minimum found is zero, then the values of the indicator variables $\{z_i, i = 1, 2, \ldots, N\}$ corresponding to that minimum are a solution to the Subset Sum problem. If for all values of c the minimum found is larger than zero, then the particular instance of the Subset Sum problem has no solution.

Given this equivalence, unless $P = NP$, no algorithm with the properties of **A** exists. Hence the index tracking problem is at least as hard as any NP-complete problem. This means that index-tracking is NP-hard, and that, unless $P = NP$, no algorithm that guarantees finding the optimal portfolio in polynomial time exists.

References

Alexander, C. (1999). Optimal hedging using cointegration. *Philosophical Transactions of the Royal Society of London. Series A. Mathematical, Physical and Engineering Sciences*, *357*(1758), 2039–2058.

Ammann, M., & Zimmermann, H. (2001). Tracking error and tactical asset allocation. *Financial Analysts Journal*, *57*(2), 32–43.

Beasley, J. E. (1990). OR-Library: distributing test problems by electronic mail. *Journal of the Operational Research Society*, *41*(11), 1069–1072.

Beasley, J. E., Meade, N., & Chang, T. J. (2003). An evolutionary heuristic for the index tracking problem. *European Journal of Operations Research*, *148*(3), 621–643.

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

Consiglio, A., & Zenios, S. A. (2001). Integrated simulation and optimization models for tracking international fixed-income indices. *Mathematical Programming*, *89*(2), 311–339.

Gilli, M., & Këllezi, E. (2001). *Threshold accepting for index tracking. Computing in economics and finance* (*Vol. 72*). Society for Computational Economics.

Holland, J. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: The University of Michigan Press.

Lobo, M., Fazel, M., & Boyd, S. (2007). Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, *152*(1), 341–365. doi:10.1007/s10479-006-0145-1.

Markowitz, H. (1987). *Mean-variance analysis in portfolio choice and capital markets*. Cambridge: Basil Blackwell.

Moral-Escudero, R., Ruiz-Torrubiano, R., & Suarez, A. (2006). Selection of optimal investment portfolios with cardinality constraints. In *Proceedings of the IEEE congress on evolutionary computation 2006* (pp. 2382–2388), 16–21 July 2006.

Radcliffe, N. (1992). Genetic set recombination. In L. Darrell Whitley (Ed.), *Foundations of genetic algorithms* (Vol. 2). San Mateo: Morgan Kaufmann.

Rudolf, M., Wolter, H., & Zimmermann, H. (1999). A linear model for tracking error minimization. *Journal of Banking and Finance*, *23*(1), 85–103.

Shapcott, J. (1992). *Index tracking: genetic algorithms for investment portfolio selection* (Technical report, EPCC-SS92-24). Edinburgh, Parallel Computing Centre.