García García, F.; Guijarro, F.; Oliver-Muncharaz, J. (2018). Index tracking optimization with cardinality constraint: a performance comparison of genetic algorithms and tabu search heuristics. Neural Computing and Applications. 30(8):2625-2641. https://doi.org/10.1007/s00521-017-2882-2

# Index tracking optimization with cardinality constraint: a performance comparison of genetic algorithms and tabu search heuristics

**Authors**

Fernando García
Universidad Politécnica de Valencia
Address: Camí de Vera s/n, Valencia, Spain
e-mail: fergarga@esp.upv.es

Francisco Guijarro[1]
Universidad Politécnica de Valencia
Address: Camí de Vera s/n, Valencia, Spain
e-mail: fraguima@upvnet.upv.es

Javier Oliver
Universidad Politécnica de Valencia
Address: Camí de Vera s/n, Valencia, Spain
e-mail: jaolmun@ade.upv.es

**Abstract**

The aim of this study was to compare the performance of the well-known genetic algorithms and tabu search heuristics with the financial problem of the partial tracking of a stock market index. Although the weights of each stock in a tracking portfolio can be efficiently determined by means of quadratic programming, identifying the appropriate stocks to include in the portfolio is an NP-hard problem which can only be addressed by heuristics. Seven real-world indexes were used to compare the above techniques and results were obtained for different tracking portfolio cardinalities. The results show that tabu search performs more efficiently with both real and artificial indexes. In general, the tracking portfolios obtained performed well in both in-sample and out-of-sample periods, so that these heuristics can be considered as appropriate solutions to the problem of tracking an index by means of a small subset of stocks.

**Keywords**
Index tracking; Heuristics; S&P 500; Artificial index

[1] Corresponding author

## 1. Introduction

Tracking a stock market index consists of creating a portfolio which replicates the performance of the index. The portfolio created is known as *tracking portfolio* and the index being tracked is often called *benchmark*. How well the tracking portfolio performs is usually quantified in terms of the tracking error, a measure of the difference between the index performance and the tracking portfolio performance.

Index tracking has attracted attention from both academics and investment managers. In fact, according to Frino et al. [21], assets benchmarked against the S&P index exceed US$1 trillion. Advocates of this investment strategy as Fama [18] point to the efficient market hypothesis, according to which it is not possible to beat the market in terms of both return and risk simultaneously. In other words, it is not possible to create a portfolio which provides higher returns than the market without taking greater risks than those implicit in market operations. Moreover, it is not feasible to create a portfolio with less risk than the market, without being penalized in terms of lower returns. According to this hypothesis, a rational investor can do no better than track the market and thereby obtain the same return while subject to the same risk.

The strategies employed by portfolio managers can be divided into two large groups: active and passive management strategies. Managers who take an active approach try to beat the market creating portfolios which are balanced in a different way than the index. The aim is to obtain greater returns while taking the same or lesser risks than those of the market, or to take a lesser risk for the same or a greater return than the market. This has been achieved by the Berkshire Hathaway investment fund run by Warren Buffet. In its 50 years of existence it has achieved a mean annual return of 21.6%, significantly more than the 9.9% achieved by the S&P index (including dividends) and with less volatility than the index. This fact proves that in practice the market is not necessarily efficient.

Passive management strategies define portfolios that replicate the index to be tracked. They implicitly assume that it is not possible to beat the market's return-risk combination. Therefore, the approach to the problem is different than the one applied by the traditional mean-variance model by Markowitz [33], where portfolio composition is calculated by means of an optimization model. Passive managers seek to minimize the number of changes to their portfolios and only perform operations to rebalance them to follow the composition and structure of the index. Conversely, active managers usually conduct more trades in an attempt to take advantage of short-term market opportunities. These managers must bear greater transaction costs deriving from the high number of sale and purchase operations, with an impact upon the strategy's net return.

The index tracking problem with cardinality constraint has been extensively analyzed in the literature (Beasley et al. [7]; Ni and Wang [35]; Filippi et al. [19]; Ruiz-Torrubiano and Suárez [37]; Sant'Anna et al. [38]; Mezali and Beasley [34]; Canakgoz and Beasley [9]; Li et al. [27]), as has been the mean-variance model with cardinality constraint (Lwin and Qu [31]; Woodside-Oriakhi et al. [43]; Streichert et al. [40]; Chang et al. [11]; Aouni et al. [4]; Cesarone et al. [10]; Maringer and Kellerer [32]). Some of these papers approach the problem from a traditional optimization perspective, but the most recent studies deal with the problem using heuristics to find solutions in a reasonable computing time which are closed to the optimal solution.

Applying the mean-variance model with cardinality constraint, Woodside-Oriakhi et al. [43] compare genetic algorithm (GA), tabu search (TS) and simulated annealing (SA). They conclude that SA is not competitive with GA and TS, having higher errors and requiring higher computation time; whereas GA and TS errors can be considered similar.

We have found no study which compares the performance of different heuristics on the index tracking problem with cardinality constraint, and which analyzes both the accuracy of the solutions obtained, as well as the computing time required.

Portfolios obtained applying the mean-variance model or index tracking can differ significantly. In the mean-variance model the global portfolio risk is minimized, while in the index tracking approach only the unsystematic component of risk is minimized. Therefore, it is interesting studying the performance of GA and TS heuristics on the index tracking problem with cardinality constraint. Moreover, it is useful to compare their performance with the one obtained in the mean-variance problem with cardinality constraint. In the latter case, as proved by Woodside-Oriakhi et al. [43], GA and TS performance is similar and superior to SA.

The motivation of this paper is twofold: 1) to compare the efficiency of GA and TS heuristics to solve the index tracking problem with cardinality constraint, analyzing and comparing their ability to find solutions which are optimal or closed to the optimum; 2) to compare the efficiency of both heuristics regarding their computational cost to find proper solutions.

Index tracking is mainly used by investors applying a passive strategy and can be performed in two ways: 1) full replication, or 2) partial replication. In the first case, the tracking portfolio has the same cardinality as the benchmark, making the calculation of the weights assigned to the different stocks a trivial matter as probed by Roll [36]. In the second case, tracking is implemented using a subset of stocks and the calculation of their weights is no longer unimportant [7, 9, 14, 16, 37]. The latter approach leads to a loss of tracking accuracy, but provides considerable advantages over a full replication approach: 1) savings on rebalancing costs, as the portfolio only contains a smaller set of stocks, 2) savings arising from not having to include in the portfolio assets whose relative weights are very small, and 3) the possibility of handling some market indices that are too large in nature, e.g. S&P500. These have little impact on the tracking error, but give rise to transaction costs every time the composition of the index changes, as the tracking portfolio needs to be rebalanced.

Any attempt to implement partial tracking faces two problems. The first is selecting the stocks to be included in the tracking portfolio. The second concerns determining the weight of each of the selected stocks. The first problem is NP-hard, as has been pointed out by Ruiz-Torrubiano and Suárez [37]. Therefore, heuristic algorithms must be used to find practical solutions close to the global optimum but which are also computationally feasible. As we show below, the second problem can be efficiently solved by quadratic programming.

The aim of this study is to compare the performance of two well-known heuristics in solving the partial index tracking problem: genetic algorithm (GA) and tabu search (TS). For this purpose, a set of simulations is carried out to estimate the distribution of the computing time necessary to obtain tracking portfolios in two different scenarios. The first scenario deals with the tracking of artificial indices which composition is known and so the optimal solutions which the heuristics have to find. The second scenario uses real indices, whose exact compositions are not known, nor are the optimal tracking portfolios. The simulation is performed using different stock market indices and different cardinalities for the tracking portfolios. As a result, the performance of the two heuristics regarding parameters can be compared in each of the two scenarios.

When undertaking index trading, it must be considered that an index is not a portfolio. Usually, indices are classified into 3 groups depending on the methodology applied to calculate them, as stated by Shoven and Sialm [39]. (1) Price-weighted indices (PWI), are those for which the value of the index for period $t$ is calculated weighting the $i$ stocks by their price: $PWI_t = \frac{1}{d_t}\sum_i P_{i,t}$. The denominator $d_t$ varies over time, avoiding discontinuities in the index every time a stock undertakes a split, pays out a dividend or the composition of the index is changed. Investors seeking to track such an index have to rebalance their portfolio each time the denominator is changed. An example of a price-weighted index is the DJIA, which denominator at the time of writing this paper has a value of 0.14602128.; (2) Value-weighted indices (VWI), are calculated so that every stock has a weight in the index proportional to its market capitalization, so the value of the index for period $t$ is $VWI_t = VWI_{t-1}\sum_i \left(w_{it}\frac{P_{it}}{P_{it-1}}\right)$, been $w_{it} = \frac{P_{it}N_{it}}{\sum_i P_{it}N_{it}}$ and where $N_{it}$ is the number of shares outstanding of company $i$ in period $t$.

Therefore, investors tracking a VWI rebalance their portfolios every time a company issues new shares or repurchases shares. This methodology is the most widely used at present. As an example we can mention the S&P500 index; (3) Equally weighted indices (EWI), are calculated following the expression $EWI_t = EWI_{t-1}\frac{1}{n}\sum_i \frac{P_{it}}{P_{it-1}}$.

In the case of value-weighted indices, they can be compared to portfolios in which weights continuously change.

Finally, the contribution of this paper is closely related with its motivation. To our best knowledge, there is no previous study which compares the performance and the efficiency of GA and TS heuristics in the field of index tracking with cardinality constraint. We use a database employed in several academic studies to compare the tracking performance on both artificial and real indices. Our research shows that TS improves GA in all indices analyzed. This result is opposite to the findings by Woodside-Oriakhi et al. [43] for a similar problem, the comparison of TS and GA on the mean-variance model with cardinality constraint.

The rest of this paper is structured as follows: Section 2 presents a formal description of the problem of partial index tracking. Section 3 reviews the recent literature, while the fourth section summarizes the functioning of the two heuristics applied: genetic algorithms and tabu search. The results obtained after using the two heuristics in different scenarios and databases are presented in the fifth section. Finally, the main conclusions are presented in Section 6.

## 2. Partial index tracking

This section provides a formal definition of partial index tracking, together with the explanation of the notation employed.

The simplest strategy for partial index tracking consists of selecting those stocks included in the index that have the highest market capitalization. However, this does not guarantee optimal tracking due to the effect of the stocks' covariance on the tracking index. The stocks with the highest market capitalization will probably include some from the same sector which are therefore highly correlated with each other. On the other hand, other sectors with a lower market capitalization will not be represented in the tracking portfolio precisely because of the low relative weight of such stocks. For that reason, choosing stocks on the basis of their market capitalization is a generally inefficient partial tracking method.

In order to propose a more accurate strategy, it is necessary to introduce some specific notation to the problem. Henceforth vectors and matrices are written in bold style and scalars in italic style. Let the series of prices over time be $\{p_i(t)\}_{i=1}^{nb}$, with $t = 1 \dots \tau$ and $nb$ the cardinality of the index. The return of stock $i$ at time $t$ is calculated by means of equation (1):

$$r_i(t) = \frac{p_i(t)-p_i(t-1)}{p_i(t-1)}, \quad i = 1 \dots nb, \quad t = 2 \dots \tau \tag{1}$$

Similarly, we can calculate the return from index $r_b(t)$ at time $t$. The return of all the stocks at time $t$ can be expressed as:

$$\mathbf{r}^T(t) = [r_1(t) \quad \dots \quad r_{nb}(t)] \tag{2}$$

where $\mathbf{r}(t)$ is a $nb \times 1$ vector. The return from tracking portfolio $p$ can be obtained by considering the percentage of the initial capital invested (the weight) in each of the stocks:

$$\mathbf{w}^T = [w_1 \quad \dots \quad w_{nb}] \tag{3}$$

where $\mathbf{w}$ is a vector with $nb$ rows, and $w_i$ represents the percentage of the capital invested in stock $i$. If full replication is the case, then $\forall i \; w_i > 0$;. If the tracking is partial, then at least one of the stocks will have a weight of zero in the tracking portfolio. Thus, the number of stocks in tracking portfolio $p$ with a weight other than zero will be $np$, with $np \leq nb$; therefore, $np$ represents the cardinality of the tracking portfolio. With values $np < nb$ a partial index tracking problem is faced, whereas in the case of $np = nb$ the problem takes the form of full replication.

The return from the tracking portfolio at time $t$ is calculated as the product of the weight vector and the return vector of the stocks at time $t$:

$$r_p(t) = \mathbf{w}^T \mathbf{r}(t) \tag{4}$$

The set of returns from the tracking portfolio and the benchmark for the period $[1 \dots \tau]$ are expressed in (5)-(6) as $\tau \times 1$ vectors:

$$\mathbf{r}_p^T = [r_p(1) \quad \dots \quad r_p(\tau)] \tag{5}$$

$$\mathbf{r}_b^T = [r_b(1) \quad \dots \quad r_b(\tau)] \tag{6}$$

The tracking accuracy during the period $1 \dots \tau$ has been calculated in different ways in the literature. One such method is the calculation of the tracking error ($TE$), which is the difference between the tracking portfolio return and the benchmark return:

$$TE(t) = r_p(t) - r_b(t) \tag{7}$$

This can also be expressed as a $\tau \times 1$ vector (8):

$$\mathbf{TE} = \mathbf{r}_p - \mathbf{r}_b \tag{8}$$

The tracking error variance ($TEV$) is calculated as (9):

$$TEV = Var(\mathbf{r}_p - \mathbf{r}_b) \tag{9}$$

This measure was the first to be employed in the literature, in papers such as Connor and Leland [15], Francks [20], Lobo et al. [30], and Roll [36]. However, the tracking error variance has an important drawback, as noted by Beasley et al. [7]; a portfolio with a constant difference in returns from the benchmark will also have a constant **TE** vector. This will give a value of zero to the tracking error variance and yet it is obvious that the tracking portfolio will not be accurately reproducing the movement of the benchmark.

In order to overcome this disadvantage, Beasley et al. [7] proposed the following as the overall measure (10):

$$\frac{1}{T}\left(\sum_{t=1}^{T}|TE(t)|^{\alpha}\right)^{1/\alpha} \tag{10}$$

where the tracking error is taken as an absolute value. Using this expression, these authors propose using the mean squared error between the tracking portfolio returns and the benchmark returns:

$$MSE\left(\mathbf{r}_p, \mathbf{r}_b\right) = \frac{1}{T}\sum_{t=1}^{T}\left(TE(t)\right)^2 \tag{11}$$

The mean squared error can be expressed as a function of the weighting vector $\mathbf{w}$:

$$MSE\left(\mathbf{r}_p, \mathbf{r}_b\right) = \frac{1}{T}\left(\mathbf{r}_p - \mathbf{r}_b\right)^t\left(\mathbf{r}_p - \mathbf{r}_b\right) = \frac{1}{T}\left(\mathbf{r}_p - \mathbf{r}_b\right)^2 = \frac{1}{T}\left(\mathbf{r}^T\mathbf{w} - \mathbf{r}_b\right)^2 \tag{12}$$

where $\mathbf{r}$ is defined as the $nb \times \tau$ matrix which includes the returns of all the stocks:

$$\mathbf{r}^T = [\mathbf{r}_1 \quad \ldots \quad \mathbf{r}_{nb}] \tag{13}$$

In this way, the problem of partial index tracking can be solved by means of a quadratic optimization model:

$$Min\ \left(\mathbf{r}^T\mathbf{w} - \mathbf{r}_b\right)^2 \tag{14}$$

$$s.t.\ \ \mathbf{1}^T\mathbf{w} = 1 \tag{15}$$

$$\mathbf{1}^T\mathbf{z} = np \tag{16}$$

where $\mathbf{1}$ is a $nb \times 1$ vector and $\mathbf{z}$ a $nb \times 1$ indicator vector, whose component $i$ will have a value of 1 if stock $i$ is present in the tracking portfolio, and 0 if not. In (14)-(16) $\mathbf{w}$ and $\mathbf{z}$ are unknown.

The objective function in (14) minimizes the mean squared tracking error. The constraint in (15) ensures that the investment is entirely distributed across the stocks which make up the tracking portfolio. The constraint in (16) limits the cardinality of the tracking portfolio to $np$ stocks.

Notice that model (14)-(16) does not explicitly consider the risk of the portfolio. In the classical mean-variance optimization model, both return and risk are simultaneously considered. However, the index tracking problem is concerned about replicating a stock market index, hence obtaining the same or very similar return and risk. Although the total risk is not explicitly included in model (14)-(16), the model considers the systematic risk. In fact, the mean squared error defined in expression (11), which is minimized in the objective function of model (14)-(16), is directly related with the systematic component of risk. In the case of perfect tracking the mean squared error will be zero, and hence the risk of the tracking portfolio will be equal to its systematic component. In the case that the tracking is not perfect, differences between the return of the index and the return of the tracking portfolio will be observed, and the mean squared error will be positive. This implies that both the systematic and non-systematic components will become also positive.

Minimizing the mean squared error between the tracking portfolio returns and the benchmark returns is tantamount to maximize the systematic component of risk. Hence, the models considered in the index tracking problems are focused on the maximization of the systematic risk (minimization of the unsystematic risk), regardless of the total risk of the portfolio. Indeed, the risk of the tracking portfolio will be constrained to be as close as possible to the risk of the replicated index. Further details on these components of risk ad how beta is related with the index tracking problem can be found in Roll [36].

This model would efficiently obtain an optimal and unique solution if no constraint on cardinality existed (16). Given this constraint, one way to solve the problem is to include this constraint in the objective function, with a consistent penalty, $d$, linked to non-compliance with the constraint on cardinality. In this case, the objective function is the following:

$$Min\ (\mathbf{r}^T\mathbf{w} - \mathbf{r_b})^2 - d|\mathbf{1}^T\mathbf{z} - np| \tag{17}$$

and the only constraint is (15). Note how the problem ceases to be quadratic and becomes non-linear, with cardinality deviations being penalized for being too high or too low.

Unfortunately, (17) cannot be efficiently solved by means of optimization algorithms and the computational costs of the problem becomes unsustainable as cardinality increases as stated by Ruiz-Torrubiano and Suárez [37]. Including the cardinality constraint as a penalty within the objective function must therefore be rejected. It is more efficient to use heuristics which only evaluate feasible solutions.

In contrast with the overall problem in (14)-(16), the model (14)-(15) can be efficiently solved – without a cardinality constraint – by using quadratic programming algorithms. To this end, it is necessary to further detail the objective function expression in (14):

$$(\mathbf{r}^T\mathbf{w} - \mathbf{r}_b)^2 = \mathbf{w}^T\mathbf{r}\mathbf{r}^T\mathbf{w} - 2\mathbf{w}^T\mathbf{r}\mathbf{r}_b + \mathbf{r}_b^T\mathbf{r}_b$$

As the third term is a constant, the expression is reduced as follows:

$$\frac{1}{2}\mathbf{w}^T\mathbf{r}\mathbf{r}^T\mathbf{w} - \mathbf{w}^T\mathbf{r}\mathbf{r}_b$$

Doing this, the model (14)-(15) is transformed into an equivalent model (18)-(21). This new model can be efficiently solved by applying quadratic optimization algorithms:

$$Min\ \left(\tfrac{1}{2}\mathbf{w}^T\mathbf{H}\mathbf{w} - \mathbf{w}^T\mathbf{g}\right) \tag{18}$$

$$s.t.\ \ \mathbf{w}^T\mathbf{1} = 1 \tag{19}$$

$$\mathbf{H} = \mathbf{r}\mathbf{r}^T \tag{20}$$

$$\mathbf{g} = \mathbf{r}\mathbf{r}_b \tag{21}$$

In this way, the problem of partial index tracking can be tackled by dividing the problem into two sub-problems: 1) the first concerns the determination of which stocks to include in the tracking portfolio, while meeting the cardinality constraint, and will be solved using heuristic

algorithms; 2) the second involves efficiently determining the weights of the tracking portfolio using a quadratic programming model (18)-(21).

Finally, the former model implies a constant weight $\mathbf{w}$ of the stocks in the portfolio. This is opposite to the popular value-weighted method, which has already been mentioned in the first section of this paper. The weight of the stocks in a value-weighted index varies constantly. Keeping a constant weight would mean to rebalance the portfolio constantly, as well. Surprisingly, this theoretical difference does not imply a big difference in practice. As an example, Figure 1 compares the evolution of two portfolios composed by randomly selecting 5 stocks from Hang-Seng index. One portfolio has been calculated using constant weights and the other one with variable weights, using the public data available from the OR-Library designed by Beasley [6]. This database is employed again later in this paper, in the results section. Portfolios have been generated taking as base 100. At the start, the first 5 stocks of the Hang-Seng are equally weighted. It can be observed that the two portfolios have a similar development over time, although their weighting design is different.

[Here Figure 1]

## 3. Recent literature on partial index tracking

Solving the problem of partial index tracking is computationally infeasible for medium to large-scale problems. For example, let us track an index composed of $nb = 100$ stocks using a tracking portfolio of $np = 10$. If we search for the optimal solution for a portfolio composed by 10 stocks, then around 2E+14 tracking portfolios need to be evaluated. Given a mean evaluation time of 0.0001 second, the search for all of the portfolios would take approximately 55 years.

Figure 2 shows the number of portfolios to evaluate (y-axis) for different cardinalities (x-axis), given an index composed of 100 stocks. Panel A shows how the growth in the number of possible portfolios is exponential with respect to the cardinality of the tracking portfolio until it reaches its maximum when $np = 50$. In panel B, the scale has been changed from linear to logarithmic in order to show the distribution more clearly.

[Here Figure 2]

Due to the computational cost of such a search, numerous studies have sought different approaches to the problem. A description of a number of relatively recent contributions is presented below.

The study by Beasley et al. [7] is one of most widely cited in the academic literature. Their formulation explicitly considered transaction costs and the revision of an existing tracking portfolio (portfolio rebalancing), and explicitly limited the number of stocks to be included in the tracking portfolio. They addressed the partial index tracking problem using evolutionary heuristics with real-value chromosome representations. In addition, computational results were presented for five data sets drawn from major world markets. The authors made the databases they used with their heuristic available to the academic community, enabling other researchers to compare new methods and approaches for partial index tracking.

Further research has also focused on using evolutionary algorithms. One noteworthy study was carried out by Chiam et al. [14], who proposed a multi-objective evolutionary index tracking platform that could simultaneously optimize both tracking performance and transaction costs throughout the investment horizon. To make their model comparable with others, the authors used the database provided by Beasley et al. [7].

Ni & Wang [35] also made use of genetic algorithms. The mathematical model they proposed is based on a hybrid genetic algorithm with a self-adaptive evolving mechanism. In order to enhance the model's efficiency, the authors optimized the original genetic algorithm by applying Pareto efficiency as a measure of utility and goal programming for the inevitable conflicts between multiple objectives or interests.

The study by Andriosopoulos et al. [3] generated portfolios using a subset of shipping stocks selected from the Dow Jones Composite Average indices. The index tracking problem was addressed using a differential evolution algorithm and a genetic algorithm. To test the performance of the heuristics three different rebalancing scenarios were examined: annually, quarterly and monthly. Transaction costs were also considered.

Li et al. [27] proposed a multi-objective optimization scheme for the enhanced index tracking problem, which maximized the extent to which the tracking index outperforms the benchmark, while minimizing the tracking error when underperforming the benchmark. The authors put forward an immunity-based multi-objective optimization algorithm to search for the solution of the enhanced index tracking problem. The problem of not only reproducing the behavior of the index, but also outperforming it in terms of returns, was also previously addressed by Roll [36] for full replication and by Canakgoz and Beasley [9] for partial tracking.

Chavez-Bedoya and Birge [12] also examined the issue of enhanced index tracking and proposed a model with a parametric approach in which the portfolio weights were modeled as functions of asset characteristics. This approach applies nonlinear and nonconvex objective functions that are difficult to incorporate into existing index tracking and enhanced indexation models. Chen and Kwon [13] put forward a 0–1 integer model in order to maximize similarity between the selected stocks and the stocks of the index. Uncertainty is introduced in the objective function by using a computationally tractable robust framework that can control for the conservativeness of the solution. This protects against worst-case realizations of potential estimation errors and other deviations.

Wang et al. [42] also used a linear 0-1 model, introducing CVaR for the measurement of risk. When a CVaR constraint is added into the general index tracking model, the resulting index tracking problem can be transformed into a mixed 0–1 linear programming problem. When the number of 0–1 variables is relatively small, the authors show that the resulting mixed 0–1 linear program can be efficiently solved using standard optimization software.

The study by García et al. [22] also considered a new element of the partial tracking problem: the curvature of the tracking frontier. This criterion is not defined by a particular portfolio, but by all the portfolios that define the tracking frontier. The main implication is that a manager can satisfy different investment profiles with the same subset of stocks, and therefore reduce transaction costs, as all the portfolios on the frontier contain the same stocks.

A different approach was taken by Guastaroba and Speranza [24], who put forward an improved Kernel Search for the partial index tracking problem. The improved variant was based on the detection of a set of the most promising items that were likely to be selected in the optimal solution of the original problem. This detection was carried out by exploiting the information collected from the application of the basic version of the algorithm. Once the set of the most promising items was determined, the original problem was solved after fixing the corresponding variables to a specific value.

Two other papers have recently been published by Aguilar-Rivera et al. [1] and Berutich et al. [8] dealing with evolutionary algorithms, stock markets and portfolio selection.

## 4. Proposed heuristics for partial index tracking

This study compares the computational cost of two heuristics addressing the index tracking problem with cardinality constraint: a genetic algorithm (GA) and tabu search (TS). Similar techniques have been used in the past to solve financial problems such as predicting price

movements, as in Barak et al. [5], Dunis et al. [17], Hsu [26], Li et al. [28], Lindemann et al. [29] and Thenmozhi and Sarath Chand [41].

### 4.1 Genetic algorithms

GAs are among the most widely studied and employed approaches in the evolutionary algorithm literature. GAs can be defined as an optimization technique based on the heuristic search for solutions. Holland [25] was inspired to design GAs by evolution in nature, which enables species to adapt to their environment. The concept was then applied to the creation of an artificial system. Unlike other classic optimization systems, GAs simultaneously examine a set of possible solutions. The candidate solutions are encoded as strings or chromosomes, in what is known as a *population within a search space*. The chromosomes compete with each other for survival, whereby only the strongest survive. Directed by selection pressures and information inheritance, after a number of generations of reproduction involving crossovers and mutation, the population finally reaches convergence. This occurs when all of the individuals in the population are essentially the same or are very similar to each other. Sometimes, a problem emerges when there is a premature convergence towards local optima, hindering the emergence of solutions closer to the global optimum.

A fundamental concept of GAs is their fitness function, which enables the fitness of each individual for the objective function to be calculated. Fitter individuals have a greater probability of reproducing by exchanging their genetic information (known as *crossover*) with other highly fit individuals. This exchange produces "children" which inherit the genetic information from their "parents", generating new individuals within the population. Mutations also frequently occur after a crossover, when one of the genes in the individuals' strings changes. In this way new solutions are generated which add information to that of the individuals that gave rise to the children. These children may replace the whole population or only the unfit individuals. The process of crossover plus mutation is iterative, as it is repeated until a satisfactory solution is found.

In our research the population size is 50. A uniform crossover operator is used in which two parents produce a single child. If a stock is present in the chromosomes of both parents, then it will appear as well in the chromosomes of the child. When a stock is not present in the chromosomes of none of the parents, the stock will not be present in the chromosomes of the child. If a stock is only present in the chromosomes of one of the parents, the probability of being present in the chromosomes of the child is 0.8.

When the number of stocks in a child is less than the cardinality $np$, then the number of stocks is completed selecting them randomly among the stocks remaining. Similarly, if the number of stocks exceeds $np$, stocks are randomly removed until the desired cardinality is achieved. Mutation probability of the chromosomes has been set at 0.1. Figure 3 shows an example of the codification of individuals in the sample and how crossover and mutation have been generated.

[Here Figure 3]

In those experiments with no time execution limit, the maximum number of iterations has been set at 5,000. Finally, the Tournament method has been used to make the selection.

### 4.2 Tabu Search

TS is an algorithmic heuristic search method that employs a memory structure to undertake moves which enable the algorithm to escape from local optima, as stated by Glover [23]. The memory structure is known as a *tabu search* and stores the most recent moves in the search.

The initial search space of TS is the space of all possible solutions that can be included in the problem. As in the GA heuristic, we have codified each potential solution as a binary vector (Figure 3).

At each iteration, the algorithm examines the neighbors of the current solution, and selects the best of those which are not on the tabu, or forbidden, list. This process consists of local transformations that can be applied to the current solution. Thus a set of neighboring solutions in the search space is associated to the current solution. In our configuration the number of neighbors to check at each iteration coincides with the cardinality of the index, and this enables to analyze more potential solutions when the index is composed from a large number of stocks.

Recent moves are stored in a tabu list to prevent cycling when moving away from local optima through non-improving moves. The tabu list most commonly used involves recording the last movements performed on the current solution, and then prohibiting reverse transformations. We have used a tabu list with the last 9 transformations of the current solution.

We have also used the intensification and diversification phases to search for the solution. The intensification phase consists of exploring the portions of the search space that seem promising, so assuring that the best solutions in these areas are found. This process is based on recency memory, which records the number of consecutive iterations where some solution components have been present in the current solution. In our case, the intensification phase is restricted to 100 iterations.

The diversification phase is a mechanism that tries to mitigate the problem of local optima, when the heuristic tends to spend most of their time in a constrained portion of the search space. This is done by forcing the search into previously unexplored areas. This process is based on frequency memory, which records the total number of iterations that some solution components have been present in the current solution or have been involved in some move. In our approach we have followed the restart diversification, which involves forcing a rarely used component in the current solution and restarting the search from this new point.

The process is repeated until the maximum number of iterations is reached, or until the optimal solution is found, if this is known. The nature of the algorithm depends on the starting point at which the moves begin. The closer this point is to the optimum, the quicker the algorithm will converge with the optimal solution. As with GAs, the selection of the best neighbor takes place by means of evaluating a fitness function. In this study, the same fitness function was used for both algorithms, so the performance of GA and TS can properly be compared.

## 5. Computational results

### 5.1 Data sets

Data from the OR-Library designed by Beasley [6] is used to compare the performance of GA and TS heuristics solving the partial index tracking problem. The OR-Library consists of an extensive collection of data available to the public for testing different statistical and optimization models. For index tracking, the OR-Library contains a set of weekly prices from 1992 to 1997 of a number of indices from around the world. In order to obtain a group of indices with heterogeneous cardinality, we used the following indices for comparison: Hang Seng 32 (Hong Kong), DAX 100 (Germany), FTSE 100 (UK), S&P 100 (USA), Nikkei 225 (Japan), S&P 500 (USA) and Russell 2000 (USA).

Using this data, with the aim of comparing the performance of GA and TS in different scenarios, partial tracking of both artificial and real indices is undertaken following the procedure of Beasley et al. [7] and Ruiz-Torrubiano and Suárez [37].

In the first case, the global optimum is known, so the comparison focuses on the time required by each heuristic to find the global optimum. In the second case the global optimum is not known and so the analysis concerns 1) establishing the accuracy of the tracking error as a

function of the CPU time of each heuristic, and 2) calculating the tracking error of the solutions found in in-sample training and comparing it with the solutions in an out-of-sample test. The aim is to determine whether the tracking portfolios designed for a particular period maintain their tracking ability in later periods. The results detailed in the following sections are obtained on a 2.5 GHz Intel Core i5 processor with 4 GB RAM.

### 5.2 Artificial indices

The exact composition of the stock market indices is not shown in the OR-Library, making it impossible to know the optimal solution to the partial tracking problem. For that reason, some artificial indices are generated whose composition is known, for which the optimal solution to the problem is therefore also known. This makes it possible to precisely determine the computation time used by each heuristic to find the optimal solution.

The artificial indices are created as follows: given an index cardinality $nb$ and a tracking cardinality $np$, $np$ stocks are randomly selected from the index. The weights of each stock in the artificial index are also randomly generated using a uniform distribution (0, 1), normalizing the weight vector so the sum is 1. As a result, the precise composition of the index is known and therefore also the optimal solution which the heuristics have to find. This process is repeated 500 times, in order to determine the distribution of the CPU time employed by each heuristic for this task.

The simulation is undertaken for the cardinalities $np \in \{5..10\}$ and for the indices referred to in the previous section. Both heuristics use the same fitness function, consisting of the model in (18)-(21), which minimizes the mean square tracking error with the only constraint that the weights of the stocks in the tracking portfolio must add up to 1.

The pseudo-code used is shown in Figure 4. The algorithm is coded in R version 3.2.1, transforming the *GA* and *tabuSearch* libraries to adapt the heuristics to the problem in question.

The code consists of three loops. The external loop enables the code to be repeated for each of the real-world indices mentioned in the previous section. The intermediate loop implements the code for the six cardinalities considered for the tracking portfolio. The internal loop undertakes the 500 simulations necessary to determine the distribution of the CPU time of each heuristic.

As seven indices and six cardinalities are used for the tracking portfolio, the total number of simulations is 21,000. For each of these simulations an artificial index is created in which both the stocks and their weights are randomly determined.

GA and TS heuristics use the artificial index as the benchmark in each simulation and search for the optimal solution using the same fitness function to minimize the mean square tracking error. The general rule for the implementation of these heuristics is to stop after a maximum execution time, or when the best solution found is not modified after a certain number of iterations. This latter is the case of the functions implemented for the GA and tabu search libraries in R. In this study, we modified this stopping rule, so that the algorithms are only stopped when the tracking error is zero, i.e. when the tracking portfolios coincide with the benchmark. So it was possible to determine the distribution of the CPU time for each heuristic across the 21,000 simulations performed. Thus the comparison of the heuristics is more precise than if it only considered the number of iterations undertaken by each one of them, given that the CPU time of each iteration can vary from one heuristic to another. The *timeGA* and *timeTS* variables reflect the CPU times employed by each heuristic.

[Here Figure 4]

The results of the use of the pseudo-code shown in Figure 4 on seven artificial indices are summarized in Figure 5. Box-and-whisker plots are used to represent the CPU times of the two heuristics for the cardinalities detailed above. The OR-Library does not record all of the stocks for each of the indices and so the tracking tests are carried out only with those that are available. For example, in the case of the Hang Seng 32 Index, the data available concern the returns of 31 of the 32 stocks included in the index.

The results show that TS is more efficient than the GA heuristic, since in all cases TS clearly solves the problem in shorter mean CPU times per execution. TS median time is always less than GA. The CPU times required by GA to find the optimal tracking portfolio displays greater dispersion. This indicates that TS is much more consistent in the search, while GA often requires a longer search time when the algorithm becomes stuck on a local solution and the mutations are unable to re-start the search for new solutions.

As expected, CPU time per execution increases for both heuristics as the cardinality of the tracking portfolio does. Nevertheless, in both cases it is found that the growth is not exponential, as Figure 2 shows. Therefore, the use of either heuristic finds optimal solutions with low cardinalities in a reasonable timeframe.

[Here Figure 5]

The possibility that the CPU times for both heuristics might be linked to the complexity of the artificial index constructed is also explored. In fact, it is possible that some instances of the problem could be particularly difficult to solve and that this difficulty influences the computational time needed by both heuristics. When calculating the correlations between the CPU times, it is found out that none of these correlations is statistically significant. As an example, Figure 6 shows the CPU times for the GA and TS heuristics on the Russell 2000 Index with $np = 6$. Each dot represents the CPU time needed by GA and TS to find the tracking portfolio using the pseudo-code shown in Figure 4.

[Here Figure 6]

5.3 Real-world indices

The previous section concludes that TS performs best in the search for the global optimum with artificial indices. This section compares the performance of both heuristics with real-world indices. In this case, neither the composition of the index nor that of the optimal tracking index is known. Therefore, the comparison of the two cannot be approached as in the previous section. Now, the CPU time $t$ for the two heuristics is limited in order to determine the level of accuracy demonstrated by the tracking error for different values of $t$. For example, both heuristics seek solutions with the CPU time limited to $t = 0.025$ seconds and their performance is then compared. Further solutions are then sought, limiting the CPU time to different values of $t$, and the tracking errors for the two heuristics are then compared.

For each real-world index and each tracking portfolio cardinality $np$, a total of 10 values for the CPU time $t$ are tested. Based on the results obtained from the artificial indices, CPU times proportional to the $np$ cardinality are considered. The 10 CPU time values are not the same for all instances of the problem, but change according to the index tracked and the $np$ cardinality. They are determined by using the following expression (22):

$$t \in \left[0.025,\ median\left(CPU\ time_{index,np}\right)/2\right] \tag{22}$$

where $CPU\ time_{index,np}$ represents the time vector for the artificial index and the cardinality of the tracking portfolio $np$. The minimum CPU time considered for all instances of the problem is always the same: 0.025 seconds. As the maximum computation time available, we have considered half of the median CPU time for the simulations undertaken on a tracking portfolio with a cardinality of 10. As the mean CPU time needed by GA is higher than that of TS, only those of GA are used to calculate this median. Once the minimum and maximum interval values are established (22), the interval is divided into 10 regularly spaced values for $t$. Doing this it is possible to analyze how the solutions obtained improve as more computing time is available.

For each of the 10 values of $t$, 30 random simulations are carried out; and for each $t$ the best solution is identified by means of MSE. Doing this, the best solution found by the heuristics for each index, CPU time $t$ and cardinality $np$ is determined. However, determining the accuracy of the heuristics regarding these parameters is not the only important issue to explore. Until this point, only the problem of how generating the tracking portfolio for the complete period $[1 \dots \tau]$ has been considered, yet the final objective of index tracking is to construct portfolios whose performance is optimal, not only with regard to the past $[1 \dots \tau]$, but, much more importantly, with regard to the future $[\tau + 1 \dots \tau + L]$. Of course, the future behavior of assets is unknown, and so it is necessary to assume that the optimal solution obtained for $[1 \dots \tau]$ will continue to perform well for the period $[\tau + 1 \dots \tau + L]$. As $\tau$ increases, it is reasonable to expect that this hypothesis is confirmed. It must be noted that this would depend on the assumption of stationarity, which may or may not hold for all index series and series components. In order to test this, we divide the database into two sub-periods of equal size. The first set of weeks $[1 \dots 145]$ is taken as the training period, with the second set $[146 \dots 290]$ being the test period. In this way it is possible to compare the performance of the first period (in-sample performance) with the second (out-of-sample performance).

Due to space constraint, only the results for Hang Seng, Nikkei and Russell indices are shown (Figure 7). In this way, we present results for low, medium and high cardinality indices. First, we have computed the best (i.e. minimum) tracking error $MSE$ obtained for 30 random repetitions of each heuristic in the in-sample period, for different values of $np$ ($np \in \{5..10\}$) and for each CPU time $t$ obtained through equation (22). Plots in Figure 7 do not include actual values of $t$, which vary according to the both cardinality of the portfolio and the index. We have simplified these plots by ordering computation time in the x axis from 1 to 10, instead of using actual computation time – different for each np –. Then, these optimal portfolios have been used in the out-of-sample period and its tracking error $MSE$ has been calculated.

Regarding the in-sample period it can be observed that GA obtains better results than TS for lower values of $t$. This holds in general for all values of cardinality $n$. But as $t$ increases, TS is able to find better solutions than GA, regardless of cardinality. So we can conclude that GA can find good local optima solutions with short computation time, while TS needs more initialization time to find these solutions. However, once GA has achieved a local optimum it is difficult to pass this point and obtain new solutions that are closer to the global optimum. The diversification phase of TS gets over these local optima and finds new solutions closer to the global optimum, so improving the tracking error $MSE$ by exploring new areas in the search space.

Despite the diversification phase makes TS solutions become superior to GA, it is also found that the differences in $MSE$ between the two heuristics are not as large as could be expected after the results obtained from the artificial indices. In fact, in the previous section TS proved to be notably superior to GA in terms of computation time; however, this did not translate into considerably better solutions when working with real-world indices.

[Here Figure 7]

A particular strength of both heuristics is their performance in the out-of-sample period. The best tracking portfolios obtained for the in-sample period also perform well in the out-of-sample period with $MSE$ values of the same magnitude. Therefore, it can be concluded that both TS and GA are capable of finding tracking portfolios which maintain a stable relationship over time with the benchmark.

For example, Figure 8 compares the performance of S&P100 with the best tracking portfolios obtained by GA and TS for $np = 10$ and $t = 0.8242$. It can be observed how the tracking portfolios closely follow the performance of the index in both the in-sample and out-of-sample periods.

[Here Figure 8]

## 6. Conclusions

The aim of this study is to compare the performance of two well-known heuristics, genetic algorithms and tabu search, on the financial problem of partial tracking of stock market indices. The database used is composed of 290 observations with the weekly performance of seven real-world indices in the OR-Library records.

The partial tracking problem is approached by dividing it into two sub-problems: The first problem is determining which stocks should be included in the tracking portfolio. Two heuristics are employed for this purpose, TS and GA. The second problem concerns calculating the weight of each stock in the tracking portfolio, which can be efficiently solved by means of quadratic programming.

The comparison of the two heuristics is implemented in two steps. First, artificial indices are constructed for which the optimal tracking portfolios are known. In this step, the CPU time needed by each heuristic to find a solution is compared. The second step applies real-world indices and the mean square error of the tracking portfolios is compared.

After this analysis, it can be concluded that TS is more efficient in the search for solutions to the partial tracking problem than GA. It is able to find the optimal solution of the artificial indices with a much lower mean CPU time than GA. The mean performance of TS is also superior in the case of real-world indices. Indeed, when CPU time for both heuristics is limited, TS is able to find better solutions (having a lower mean square error) when considering realistic cardinalities for the tracking portfolios. However, both heuristics have a positive outcome, as the tracking portfolios generated during the in-sample period also perform well in the out-of-sample period for TS and GA.

Nevertheless, computing time might be a problem when tracking indices like Russell 2000, specially as the cardinality of the tracking portfolio increases. In our example, the highest cardinality in the tracking portfolio is just 10, but some portfolios may require higher cardinalities to optimally track the indices. If this is the case, the computing time will increase considerably for large indices especially with high cardinalities of the tracking portfolio. The search space does grow more than linearly. In such situation time computation can become an important issue to control for.

Finally, we would like to emphasize that searching for heuristics that minimize computing time is itself an important task from the academic point of view. In our study, the computing time required for both heuristics to find optimal or quasi-optimal solutions is very short. Therefore, both heuristics can be used to solve the index tracking problem with cardinality constraint in the case of stock market indices with a small or medium number of stocks.

## Conflict of interest

The authors declare that they have no conflict of interest.

## References


1. Aguilar-Rivera R, Valenzuela-Rendón M, Rodríguez-Ortiz JJ (2015) Genetic algorithms and Darwinian approaches in financial applications: A survey. Expert Syst Appl 42:7684–7697

2. Alexander SS (1961) Price Movements in speculative Markets: Trends or Random Walks. Ind Manag Rev 2:7–26

3. Andriosopoulos K, Doumpos M, Papapostolou NC, Pouliasis PK (2013) Portfolio optimization and index tracking for the shipping stock and freight markets using evolutionary algorithms. Transp Res Part E Logist Transp Rev 52:16–34

4. Aouni B, Colapinto C, La Torre D (2013) A cardinality constrained stochastic goal programming model with satisfaction functions for venture capital investment decision making. Ann Oper Res 205: 77–88.

5. Barak S, Dahooie JH, Tichý T (2015) Wrapper ANFIS-ICA method to do stock market timing and feature selection on the basis of Japanese Candlestick. Expert Syst Appl 42:9221–9235

6. Beasley JE (1990) OR-library : distributing test problems by electronic mail. J Oper Res Soc 41:1069–1072

7. Beasley JE, Meade N, Chang TJ (2003) An evolutionary heuristic for the index tracking problem. Eur J Oper Res 148:621–643

8. Berutich JM, López F, Luna F, Quintana D (2016) Robust technical trading strategies using GP for algorithmic portfolio selection. Expert Syst Appl 46:307–315

9. Canakgoz NA, Beasley JE (2008) Mixed-integer programming approaches for index tracking and enhanced indexation. Eur J Oper Res 196:384–399

10. Cesarone F, Scozzari A, Tardella F (2013) A new method for mean-variance portfolio optimization with cardinality constraints. Ann Oper Res 205: 213–234

11. Chang TJ, Meade N, Beasley JE, Sharaiha YM (2000) Heuristics for cardinality constrained portfolio optimisation. Comput Oper Res 27:1271–1302

12. Chavez-Bedoya L, Birge JR (2014) Index tracking and enhanced indexation using a parametric approach. J Econ Financ Adm Sci 19:19–44

13. Chen C, Kwon RH (2012) Robust portfolio selection for index tracking. Comput Oper Res 39:829–837

14. Chiam SC, Tan KC, Al Mamun A (2013) Dynamic index tracking via multi-objective evolutionary algorithm. Appl Soft Comput J 13:3392–3408

15. Connor G, Leland H (1995) Cash Management for Index Tracking. Financ Anal J 51:75–80

16. Corielli F, Marcellino M (2006) Factor based index tracking. J Bank Financ 30:2215–2233

17. Dunis CL, Rosillo R, de la Fuente D, Pino R (2013) Forecasting IBEX-35 moves using support vector machines. Neural Comput Appl 23:229–236

18. Fama EF (1970) Efficient Capital Markets: A Review of Theory and Empirical Work. J Finance 25:383–417

19. Filippi C, Guastaroba G, Speranza MG (2016) A heuristic framework for the bi-objective enhanced index tracking problem. Omega, in press

20. Franks EC (1992) Targeting excess-of-benchmark returns. J Portf Manag 18:6–12

21. Frino A, Gallagher DR, Oetomo TN (2005) The Index Tracking Strategies of Passive and Enhanced Index Equity Funds. Aust J Manag 30:23–55

22. García F, Guijarro F, Moya I (2011) The curvature of the tracking frontier: A new criterion for the partial index tracking problem. Math Comput Model 54:1781–1784

23. Glover F (1986) Future paths for integer programming an links to artificial intelligence. Comput Oper Res 13:533–549

24. Guastaroba G, Speranza MG (2012) Kernel Search: An application to the index tracking problem. Eur J Oper Res 217:54–68

25. Holland JH (1975) Adaptation in natural and artificial systems. an introductory analysis with applications to biology, control and artificial intelligence. University of Michigan Press

26. Hsu C (2013) A hybrid procedure with feature selection for resolving stock / futures price forecasting problems. Neural Comput Appl 22:651–671

27. Li Q, Sun L, Bao L (2011) Enhanced index tracking based on multi-objective immune algorithm. Expert Syst Appl 38:6101–6106

28. Li X, Xie H, Wang R, Cai Y, Cao J, Wang F, Min H, Deng X (2014) Empirical analysis: stock market prediction via extreme learning machine. Neural Comput Appl 27:67–78

29. Lindemann A, Dunis CL, Lisboa P (2005) Level estimation, classification and probability distribution architectures for trading the EUR/USD exchange rate. Neural Comput Appl 14:256–271

30. Lobo MS, Fazel M, Boyd S (2007) Portfolio optimization with linear and fixed transaction costs. Ann Oper Res 152:341–365

31. Lwin K, Qu R (2013) A hybrid algorithm for constrained portfolio selection problems. Appl Intell 39: 251–266

32. Maringer D, Kellerer H (2003) Optimization of cardinality constrained portfolios with a hybrid local search algorithm. OR Spectrum 25: 481–495

33. Markowitz H (1952) Portfolio selection. J Financ 7: 77–91

34. Mezali H, Beasley JE (2014) Index tracking with fixed and variable transaction costs. Optim Lett 8: 61–80

35. Ni H, Wang Y (2013) Stock index tracking by Pareto efficient genetic algorithm. Appl Soft Comput J 13:4519–4535

36. Roll R (1992) A mean/variance analysis of tracking error. J Portf Manag 18:13–22

37. Ruiz-Torrubiano R, Suárez A (2009) A hybrid optimization approach to index tracking. Ann Oper Res 166:57–71

38. Sant'Anna LR, Filomena RP, Guedes PC, Borenstein D (2016) Index tracking with controlled number of assets using a hybrid heuristic combining genetic algorithm and non-linear programming. Ann Oper Res, in press

39. Shoven JB, Sialm C (2000) The Dow Jones Industrial Average: the impact of fixing its flaws. J Wealth Manag 3:9–18

40. Streichert F, Ulmer H, Zell A (2004) Evolutionary algorithms and the cardinality constrained portfolio optimization problem. Operations Research Proceedings 2003, 253–260. Springer Berlin Heidelberg

41. Thenmozhi M, Sarath Chand G (2016) Forecasting stock returns based on information transmission across global markets using support vector machines. Neural Comput Appl 27:805–824

42. Wang M, Xu C, Xu F, Xue H (2012) A mixed 0-1 LP for index tracking problem with CVaR risk constraints. Ann Oper Res 196:591–609

43. Woodside-Oriakhi M, Lucas C, Beasley JE (2011) Heuristic algorithms for the cardinality constrained efficient frontier. Eur J Oper Res 213: 538–550