Sparse Portfolios for High-Dimensional Financial Index Tracking

Konstantinos Benidis, Yiyong Feng, and Daniel P. Palomar, *Fellow, IEEE*

***Abstract*—Index tracking is a popular passive portfolio management strategy that aims at constructing a portfolio that replicates or tracks the performance of a financial index. The tracking error can be minimized by purchasing all the assets of the index in appropriate amounts. However, to avoid small and illiquid positions and large transaction costs, it is desired that the tracking portfolio consists of a small number of assets, i.e., a sparse portfolio. The optimal asset selection and capital allocation can be formulated as a combinatorial problem. A commonly used approach is to use mixed-integer programming (MIP) to solve small sized problems. Nevertheless, MIP solvers can fail for high-dimensional problems while the running time can be prohibiting for practical use. In this paper we propose efficient and fast index tracking algorithms that automatically perform asset selection and capital allocation under a set of general convex constraints. A special consideration is given to the case of the non-convex holding constraints and to the downside risk tracking measure. Further, we derive specialized algorithms with closed-form updates for particular sets of constraints. Numerical simulations show that the proposed algorithms match or outperform existing methods in terms of performance, while their running time is lower by many orders of magnitude.**

***Index Terms*—Index tracking, sparsity, majorization-minimization.**

I. Introduction

Fund managers follow two basic investment strategies, namely: active and passive. In active management strategies, the fund managers assume that the markets are not perfectly efficient and through their expertise and superior prediction methods they hope to add value by choosing high performing assets. On the contrary, the passive management strategies are based on the assumption that the market cannot be beaten in the long run. The passive managers have less flexibility and their role is to conform to a closely defined set of criteria.

Analysis of historical data has shown that the majority of the actively managed funds do not outperform the market in the long run [1]. Further, the stock markets have historically risen and therefore reasonable returns can be obtained without the active management's risk. These reasons have prompted the investor's interest into more passive management strategies.

K. Benidis and D. P. Palomar are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology (HKUST), Hong Kong (e-mail: kbenidis@connect.ust.hk, palomar@ust.hk).

Y. Feng is with Three Stones Capital Limited, Hong Kong (e-mail: yiyong@connect.ust.hk). He participated in this work when he was with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology (HKUST), Hong Kong.

This work was supported by the Hong Kong RGC 16208917 research grant.

Index tracking, also known as index replication, is one of the most popular passive portfolio management strategies. It refers to the problem of reproducing the performance of a market index. Although index tracking is driven from the financial industry, it is in fact a pure signal processing problem: an ℓ_2-norm regression of the financial historical data subject to some portfolio constraints. Further, the sparse index tracking problem (that we will introduce shortly) is similar to many sparsity formulations in the signal processing area, such as the "lasso" [2], in the sense that it is a regression problem with some sparsity requirements.

There are two main approaches to the index tracking problem: the static and the dynamic. In the static approach, we construct and hold the tracking portfolio during the time horizon of interest. On the contrary, in the dynamic setting the portfolio is readjusted in a more ad-hoc manner following a trading strategy. For both approaches it is essential the design of an appropriate tracking portfolio regardless of the underlying strategy. In this paper, it is not our scope to provide a trading strategy but rather the tools for constructing efficient tracking portfolios. Therefore, we consider fixed holding periods. Nevertheless, the application of the proposed algorithms in a dynamic approach is straightforward since the only change is the readjustment decision and not the design process.

The most straightforward way to construct an efficient index tracking portfolio is to buy appropriate quantities of all the assets that compose the index, a technique known as full replication, given that the true index construction weights are available. Following this approach, a perfect tracking can be achieved. However, there are some important drawbacks. First, a portfolio consisting of all the assets incorporates too many small and illiquid stocks. This translates into higher risk to investors since an illiquid stock is hard to sell if we are looking to exit and we might have to sell it for less than the current market price. Further, allocating capital to all the assets increases significantly the transaction costs.

Another popular way to engage in index tracking is to purchase an exchange traded fund (ETF). An ETF is like a stock but its value tracks closely a given index. It is constructed either by using derivative products (synthetic ETF) or the underlying components of the index (physical ETF). Many physical ETFs use full replication of the index they are tracking, e.g. the Standard and Poor's Depositary Receipts (Bloomberg ticker SPY) based on the S&P 500 and the Nasdaq 100 Trust Shares (Bloomberg ticker QQQ) based on the Nasdaq 100. However, there are also many ETFs using a sparse construction, where representative sampling, with 80-95% of the underlying securities being used, or aggressive

sampling, with only a tiny percentage being used [3]–[6].

There are several challenges that we face when we engage in index tracking. First, due to the price changes of the assets, the relative quantities of the holding portfolio will change daily and, after a period of time, they may diverge significantly from their designed value. In order to compensate for these changes we need to *rebalance* our portfolio frequently in order to recover the initial design. Further, when we construct a portfolio with the goal of tracking a financial index, we use the historical data of the assets that compose the index in order to find a portfolio that meets some criteria. However, since the dynamics of the market and the composition of an index constantly change, it is not wise to use data from the distant past but rather only from a recent period. This signifies the importance of holding a tracking portfolio for a limited amount of time since it can become obsolete if it does not take into account all of these changes. These reasons make necessary the *redesign* of a tracking portfolio after a period of time.

The above challenges lead to a natural trade-off between the tracking efficiency and the transaction costs. By rebalancing or redesigning frequently our portfolio we can achieve a better tracking but we increase the transaction costs, especially when the portfolio consists of a large number of assets as in full replication. A natural way to deal with these problem is to use a small number of assets to (approximately) replicate an index. This leads to the construction of a sparse index tracking portfolio [7], [8], which is the main focus of this paper.

The rest of the paper is organized as follows. Section II reviews the related work and the different problem formulations. Section III presents the solution of the sparse index tracking problem considering a set of general convex constraints. Closed-form update algorithms are further proposed for particular constraint sets. Section IV deals with the non-convex holding constraints, proposing general and particular algorithms. Section V presents possible constraints and different tracking error functions. Section VI provides a convergence analysis and an acceleration scheme that increases the convergence rate of the proposed algorithms. Section VII presents numerical experiments on real and synthetic data. Finally, the paper is concluded in Section VIII.

Notation: The sequence of nonnegative integers is denoted by $\mathbb{N}$. The real field is denoted by $\mathbb{R}$, while $\mathbb{R}^m$ ($\mathbb{R}^m_+$) denotes the set of (nonnegative) real vectors of size m, and $\mathbb{R}^{n\times m}$ ($\mathbb{R}^{n\times m}_+$) the set of (nonnegative) real matrices of size $n\times m$. Vectors are denoted by bold lower case letters and matrices by bold capital letters i.e., $\mathbf{x}$ and $\mathbf{X}$, respectively. A general entry of a vector $\mathbf{x}$ is denoted by x, its i-th entry by x_i, the i-th column of matrix $\mathbf{X}$ by $\mathbf{x}_i$, and the (i-th,j-th) element of a matrix by x_{ij}. A vector of zeros is denoted by $\mathbf{0}$ and a vector of ones by $\mathbf{1}$, while $\mathbf{I}$ denotes the identity matrix. Their dimension will be implicit from the context. The superscript $(\cdot)^\top$ denotes the transpose of a matrix. $(\mathbf{x})^+ = \max(\mathbf{x},\mathbf{0})$ where the max operator is performed elementwise. $\mathbf{x}\odot\mathbf{y}$ denotes the Hadamard product of $\mathbf{x}$ and $\mathbf{y}$. $\mathrm{Diag}(\mathbf{x})$ is a diagonal matrix formed with $\mathbf{x}$ at its principal diagonal, and $\mathrm{diag}(\mathbf{X})$ is a column vector consisting of all the diagonal elements of $\mathbf{X}$. $\|\mathbf{x}\|_0$ denotes the number of nonzero elements of a vector $\mathbf{x}\in\mathbb{R}^m$. $\mathbf{S}\succcurlyeq 0$ means that the symmetric matrix $\mathbf{S}$ is positive semidefinite, while $\lambda^{(\mathbf{S})}_{\max}$ is its maximum eigenvalue. $\mathrm{card}(\mathcal{A})$ denotes the cardinality of the set $\mathcal{A}$. $\mathbf{x}_{[i:j]} = [x_i, x_{i+1}, \ldots, x_j]^\top$, with $i\leq j$ integers.

II. Related Work

We first provide some basic definitions that we will use throughout the paper. Assume that an index is composed of N assets. We denote by $\mathbf{r}^b = [r^b_1,\ldots,r^b_T]^\top\in\mathbb{R}^T$ and $\mathbf{X} = [\mathbf{r}_1,\ldots,\mathbf{r}_T]^\top\in\mathbb{R}^{T\times N}$ the returns of the index and the N assets in the past T days, respectively, with $\mathbf{r}_t\in\mathbb{R}^N$ denoting the returns of the N assets at the t-th day[1]. Further, $\mathbf{b}\in\mathbb{R}^N_+$ denotes the normalized benchmark index weights such that $\mathbf{b}>\mathbf{0}$, $\mathbf{b}^\top\mathbf{1}=1$ and $\mathbf{X}\mathbf{b}=\mathbf{r}^b$.

A commonly used tracking measure is the empirical tracking error (ETE) which measures how closely the tracking portfolio replicates the index [7]–[10]. It is defined as:

$$\mathrm{ETE}(\mathbf{w}) = \frac{1}{T}\left\|\mathbf{X}\mathbf{w}-\mathbf{r}^b\right\|_2^2, \tag{1}$$

where $\mathbf{w}$ denotes the portfolio we wish to design that must satisfy $\mathbf{w}\geq\mathbf{0}$ and $\mathbf{w}^\top\mathbf{1}=1$, among other possible constraints.

Note that (1) assumes a constant $\mathbf{w}$ in the corresponding T days, which implicitly means that the portfolio is rebalanced on a daily basis. During the design process, this is a necessary approximation (that we will make in all the following formulations) in order to have a tractable problem. However, in the out-of-sample evaluation of the derived portfolios we should always take into account the portfolio changes for a proper performance assessment (see Section VII).

The first approach of sparse index tracking is to decompose the task into two steps, namely stock selection, where we select a subset of the assets, and capital allocation, where we distribute the capital among the selected assets. Various stock selection methods have been proposed. A widely used method, especially for a market capitalization weighted index, is to select the largest $K<N$ assets according to their market capitalization [11]. Another approach is to select the assets that have similar return performances as the index, i.e., the K most correlated assets to the index [12], [13]. Finally, a selection based on cointegration has been proposed where the idea is to select K assets so that there exists a linear combination of their log-prices cointegrated well with the value of the index [13], [14].

Once a subset of K assets has been selected we need to assign the capital in a proper manner. A naive allocation is to distribute the capital among the selected assets proportional to the original weights with their summation equal to 1. This requires that the benchmark portfolio weight vector $\mathbf{b}$ and all its changes (the benchmark weight vector is consistently rebalanced by the indices providers) are known exactly, which could be very expensive. For example, in 2006 the index sponsors S&P, Dow Jones, MSCI, and FTSE earned total revenues of $1.66 billion from the ETF providers and therefore the ETF provides were even thinking of cutting these costs by setting up their own market indices[2]. Although this naive

[1]The return r_t of an index or an asset at time t is defined as $r_t = \frac{p_t - p_{t-1}}{p_{t-1}}$, where p_t and p_{t-1} denote the price at time t and $t-1$, respectively.

[2]See "ETF providers float idea of setting up their own market indices", published in Financial Times on 2017-05-24.

allocation is easy and fast, it is not optimized. Further, in many cases the index weight vector $\mathbf{b}$ is not available. To this end, we can use the following optimized allocation that does not make any use of the index weight vector $\mathbf{b}$ [11]:

$$\begin{array}{ll}\underset{\mathbf{w}}{\text{minimize}} & \frac{1}{T}\left\|\mathbf{X}(\mathbf{w}\odot\mathbf{s})-\mathbf{r}^b\right\|_2^2 \\ \text{subject to} & (\mathbf{w}\odot\mathbf{s})^\top\mathbf{1}=1, \\ & \mathbf{w}\geq\mathbf{0},\end{array} \tag{2}$$

where s_i is 1 if the i-th stock is selected and 0 otherwise, with $\mathbf{s}^\top\mathbf{1}=K$. This is effectively the minimization of the tracking error using only the selected assets. To make the solution more robust, one can remove a few assets each time and apply this idea iteratively several times to achieve enough sparsity [7].

The previous methods allocate the capital in two steps (i.e., asset selection and capital optimization) and it is not clear how optimal the resulting tracking portfolio is. Another approach that unifies these two steps is to directly penalize the cardinality of the tracking portfolio [7], [15]:

$$\begin{array}{ll}\underset{\mathbf{w}}{\text{minimize}} & \frac{1}{T}\left\|\mathbf{X}\mathbf{w}-\mathbf{r}^b\right\|_2^2+\lambda\|\mathbf{w}\|_0 \\ \text{subject to} & \mathbf{w}^\top\mathbf{1}=1, \\ & \mathbf{w}\geq\mathbf{0},\end{array} \tag{3}$$

where $\lambda\geq 0$ is a parameter that controls the sparsity of the portfolio, i.e., we get sparser solutions for larger values of λ. With this formulation, we jointly perform an asset selection and allocation of the capital to the selected assets. Notice that this problem is highly non-convex due to the ℓ_0-"norm" term[3]. We revisit this problem in Section III, where we will rely on the MM method to iteratively upperbound (3), while the major difficulty will be to find the right surrogate function.

All the constraints we have imposed in the problem formulation (3) are convex. In practice, the constraints that are usually considered in the index tracking problem can be written in a convex form. However, there is one exception. It is common for the fund managers to impose some holding constraints to avoid extreme positions or brokerage fees for very small orders, which translates into non-convex constraints. In that case the optimization problem takes the following form:

$$\begin{array}{ll}\underset{\mathbf{w},\mathbf{s}}{\text{minimize}} & \frac{1}{T}\left\|\mathbf{X}\mathbf{w}-\mathbf{r}^b\right\|_2^2+\lambda\mathbf{s}^\top\mathbf{1} \\ \text{subject to} & \mathbf{w}^\top\mathbf{1}=1, \\ & \mathbf{l}\odot\mathbf{s}\leq\mathbf{w}\leq\mathbf{u}\odot\mathbf{s}, \\ & \mathbf{s}\in\{0,1\}^N,\end{array} \tag{4}$$

where $s_i=\mathcal{I}_{\{w_i>0\}}$ plays the role of the indicator function, but here $\mathbf{s}$ is a variable and not fixed as in (2), and $\mathbf{l},\mathbf{u}\in\mathbb{R}_+^N$, with $\mathbf{0}\leq\mathbf{l}\leq\mathbf{u}$, are the lower and upper holding constraints, respectively, for the selected stocks. For clarity of presentation, we will consider the simple case $\mathbf{l}=l\cdot\mathbf{1}$ and $\mathbf{u}=u\cdot\mathbf{1}$, with $0\leq l\leq u$, however, our approach holds for the general case.

Many approaches have been proposed to deal with problem (4). A practical heuristic is to solve the problem without the binary variable $\mathbf{s}$ (as in (1)), then select the assets with weights larger than a certain threshold, and finally reoptimize the weights with $\mathbf{s}$ fixed (as in (2)). Another approach is to use mixed-integer programming (MIP). Problem (4) is indeed an MIP and commercial solvers (e.g., Gurobi, CPLEX) can be used [10]. However, they are appropriate only for small or medium sized problems since it is hard to find an optimal solution within practical time limits when the dimension becomes large. We revisit problem (4) in Section IV.

In other works, a common approach is to approximate the ℓ_0-"norm" with the non-convex ℓ_p-"norm", where $p<1$, and use a heuristic algorithm to solve the optimization problem [7], [16]. In [17], the authors approximate the ℓ_0-"norm" with a non-convex function that becomes convex in the limit. Thus, they propose a gradual non-convexity method. In [18], the authors consider a general formulation and propose an algorithm that determines whether or not to rebalance a given portfolio. In [19], a mean-variance analysis or the index tracking portfolios is provided based on the classical Markowitz mean-variance framework [20], while in [21], Brodie et al. consider the problem of portfolio selection within the Markowitz framework including an ℓ_1-penalty. In [22], Takeda et al. formulate an MIP problem. Since it is hard to solve it in practice due to the prohibiting running time, they propose a greedy algorithm. Finally, genetic algorithms [8], [10], [11], [23], [24] and differential evolution [9], [23], [25] heuristics have been proposed. However, these methods are not able to guarantee optimality of the solution and, in general, they have inferior performance compared to an MIP solver [10].

In all the problem formulations presented in this section we have included only a minimum set of constraints for illustration purposes. In the rest of the paper we will use a general constraint set that will include all the possible convex constraints, i.e., $\mathbf{w}\in\mathcal{W}$, where $\mathcal{W}$ is convex. We will further assume that $\{\mathbf{w}|\mathbf{w}\geq\mathbf{0},\mathbf{w}^\top\mathbf{1}=1\}\subseteq\mathcal{W}$. When we need to include a non-convex constraint we will indicate it separately.

III. Sparse Index Tracking

In this section we focus on problem (3), i.e., the joint asset selection and capital allocation problem. First, we approximate the ℓ_0-"norm" by a continuous and differentiable function, albeit still non-convex. Then, we use the majorization-minimization framework to derive an algorithm based on the first-order Taylor expansion. Finally, we propose a specialized iterative closed-form update algorithm for a particular set of constraints.

A. ℓ_0 Approximate Function

A popular convex approximation of the ℓ_0-"norm" that promotes sparsity is the ℓ_1-norm (as indicated by the blue dashed line in Figure 1), i.e., the least absolute shrinkage and selection operator (LASSO) technique [2]. Unfortunately, this approximation does not work in index tracking since we require (among other constraints) the weights to be nonnegative and their summation to be equal to one. Thus, the ℓ_1-norm

[3]Although ℓ_p with $p<1$ is not a norm, it is a common abuse of notation to call it as such. To highlight this difference we will use quotation marks when we are dealing with an ℓ_p with $p<1$, i.e., ℓ_p-"norm".

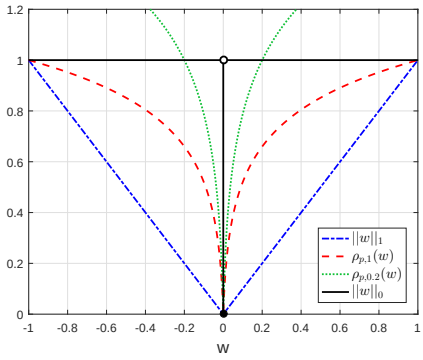


Fig. 1: Approximation of the indicator function with $\rho_{p,1}(w)$ and $\rho_{p,0.2}(w)$ with $p = 10^{-3}$.

trivially becomes: $\|\mathbf{w}\|_1 = |\mathbf{w}|^\top \mathbf{1} = \mathbf{w}^\top \mathbf{1} = 1$, which is a constant.

Instead of the ℓ_1-norm, we can approximate the ℓ_0-"norm" (to be exact the indicator function) by the following function[4]:

$$\rho_p(w) = \frac{\log(1 + |w|/p)}{\log(1 + 1/p)}, \tag{5}$$

where $0 < p \ll 1$ is a parameter and $\rho_p(w) \to \mathcal{I}_{\{w \neq 0\}}$ as $p \to 0$. This function was also used in [26] to replace the ℓ_1-norm that led to the well known iteratively reweighted ℓ_1-norm minimization algorithm, and in [27], [28] for sparse eigenvector extraction.

In practice, $\rho_p(w)$ is a good approximation of the ℓ_0-"norm" when $|w| \in [0, 1]$. In many cases we are interested in approximating the ℓ_0-"norm" in other intervals, e.g., in the interval $[0, u]$ where $u \leq 1$ is an upper bound of the index weights, or in the interval $[0, l]$, where l is a lower bound of the index weights. The use of the latter will become clear in Section IV where we introduce the non-convex lower holding constraints. To this end, we use a modified version of the function $\rho_p(w)$ defined as:

$$\rho_{p,\gamma}(w) = \frac{\log(1 + |w|/p)}{\log(1 + \gamma/p)}, \tag{6}$$

where $\gamma > 0$. Notice that $\rho_p(w) = \rho_{p,1}(w)$. The function $\rho_{p,\gamma}(w)$ is a good approximation of the indicator function in the interval $[0, \gamma]$ as shown in Figure 1.

For the multivariate case, it is convenient to define:

$$\boldsymbol{\rho}_{p,\gamma}(\mathbf{w}) = [\rho_{p,\gamma}(w_1), \ldots, \rho_{p,\gamma}(w_N)]^\top.$$

Now, we can approximate problem (3) as follows:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{T}\left\|\mathbf{X}\mathbf{w} - \mathbf{r}^b\right\|_2^2 + \lambda \mathbf{1}^\top \boldsymbol{\rho}_{p,u}(\mathbf{w}) \\ \text{subject to} \quad & \mathbf{w} \in \mathcal{W}. \end{aligned} \tag{7}$$

Here, we have set $\gamma = u$, where u is an upper bound of the weights specified in $\mathcal{W}$. If there is not an upper bound constraint then implicitly $u = 1$.

This problem is not convex since the function $\rho_{p,u}(w)$ is concave (for $w \geq 0$). To deal with the non-convexity we will use the majorization-minorization (MM) framework that is briefly introduced for completeness.

[4]More precisely, (5) approximates the indicator function and then we can use $\|\mathbf{w}\|_0 = \sum_{i=1}^N \mathcal{I}_{\{w_i \neq 0\}}$.

B. Majorization-Minimization

The majorization-minimization algorithm is a way to handle optimization problems that are too difficult to face directly [29], [30]. Consider a general optimization problem

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X}, \end{aligned}$$

where $\mathcal{X}$ is a closed set. We say that the function $f(\mathbf{x})$ is majorized at a given point $\mathbf{x}^{(k)}$ (with k denoting iteration) by the surrogate function $g\left(\mathbf{x}|\mathbf{x}^{(k)}\right)$ if the following properties are satisfied (in words if it is a tight upper bound):

- $f\left(\mathbf{x}^{(k)}\right) = g\left(\mathbf{x}^{(k)}|\mathbf{x}^{(k)}\right)$,
- $f(\mathbf{x}) \leq g\left(\mathbf{x}|\mathbf{x}^{(k)}\right), \forall \mathbf{x} \in \mathcal{X}$,
- $\nabla f\left(\mathbf{x}^{(k)}\right) = \nabla g\left(\mathbf{x}^{(k)}|\mathbf{x}^{(k)}\right)$.

Then, $\mathbf{x}$ is iteratively updated as:

$$\mathbf{x}^{(k+1)} = \arg\min_{\mathbf{x} \in \mathcal{X}} g\left(\mathbf{x}|\mathbf{x}^{(k)}\right). \tag{8}$$

With this scheme, it is easy to prove that the objective value is decreased monotonically at each iteration, i.e., $f\left(\mathbf{x}^{(k+1)}\right) \leq g\left(\mathbf{x}^{(k+1)}|\mathbf{x}^{(k)}\right) \leq g\left(\mathbf{x}^{(k)}|\mathbf{x}^{(k)}\right) \leq f\left(\mathbf{x}^{(k)}\right)$.

In practice, the main difficulty in the derivation of an MM algorithms is to find an appropriate surrogate function such that the minimizer of $g\left(\mathbf{x}|\mathbf{x}^{(k)}\right)$ can be easily found or even have a closed-form solution. This is not a trivial task since there is no systematic way of constructing surrogate functions and the derivation depends highly on the structure and the special characteristics of each problem [30].

C. First-order Taylor Approximation

Since $\boldsymbol{\rho}_{p,\gamma}(\mathbf{w})$ is separable[5] we just need to find a majorization function for the univariate case, i.e., $\rho_{p,\gamma}(w)$. Here, following [26], an upper bound is provided for $\rho_{p,\gamma}(w)$ at each iteration point $w^{(k)}$ by its first-order Taylor expansion.

Lemma 1. *The function $\rho_{p,\gamma}(w)$, with $w \geq 0$, is majorized at $w^{(k)}$ by $g_{p,\gamma}(w, w^{(k)}) = d_{p,\gamma}(w^{(k)})w + c_{p,\gamma}(w^{(k)})$, where*

$$d_{p,\gamma}(w^{(k)}) = \frac{1}{\kappa_1(p + w^{(k)})} \tag{9}$$

and

$$c_{p,\gamma}(w^{(k)}) = \frac{\log\left(1 + w^{(k)}/p\right)}{\kappa_1} - \frac{w^{(k)}}{\kappa_1(p + w^{(k)})}, \tag{10}$$

with $\kappa_1 = \log(1 + \gamma/p)$.

Proof. The function $\rho_{p,\gamma}(w)$ is concave for $w \geq 0$. Thus, an upper bound of the function is its first-order Taylor approximation at any point $w_0 \geq 0$ [15], [26]:

$$\rho_{p,\gamma}(w) = \frac{\log(1 + w/p)}{\log(1 + \gamma/p)}$$

[5]Here, separable means that each entry $\rho_{p,\gamma}(w_i)$ of $\boldsymbol{\rho}_{p,\gamma}(\mathbf{w})$, as defined in (III-A), is a function only of w_i.

Algorithm 1 LAIT - Linear Approximation for Index Tracking problem (7)

1: Set $k=0$, choose $\mathbf{w}^{(0)} \in \mathcal{W}$
2: **repeat**:
3: Compute $\mathbf{d}_{p,u}^{(k)}$ according to (11) and (9)
4: Solve (13) and set the optimal solution as $\mathbf{w}^{(k+1)}$
5: $k \leftarrow k+1$
6: **until** convergence
7: **return** $\mathbf{w}^{(k)}$

$$\begin{aligned}&\leq \frac{1}{\log(1+\gamma/p)}\left[\log\left(1+w_0/p\right)+\frac{1}{p+w_0}(w-w_0)\right]\\&= d_{p,\gamma}w + b_{p,\gamma},\end{aligned}$$

where $d_{p,\gamma}$ and $b_{p,\gamma}$ are given by (9) and (10), respectively. □

In the following, it is convenient to define:

$$\mathbf{d}_{p,\gamma}^{(k)} = \left[d_{p,\gamma}(w_1^{(k)}),\ldots,d_{p,\gamma}(w_N^{(k)})\right]^\top, \tag{11}$$

$$\mathbf{c}_{p,\gamma}^{(k)} = \left[c_{p,\gamma}(w_1^{(k)}),\ldots,c_{p,\gamma}(w_N^{(k)})\right]^\top. \tag{12}$$

Now, discarding the constant terms, at the $(k+1)$-th iteration we can solve the following convex optimization problem:

$$\begin{aligned}\underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{T}\left\|\mathbf{X}\mathbf{w}-\mathbf{r}^b\right\|_2^2 + \lambda \mathbf{d}_{p,u}^{(k)\top}\mathbf{w}\\ \text{subject to} \quad & \mathbf{w}\in\mathcal{W},\end{aligned} \tag{13}$$

where we have again set $\gamma = u$.

Algorithm 1 summarizes the above iterative procedure. We refer to it as LAIT.

D. Specialized Algorithms

In the previous section we derived an iterative algorithm that works for a general convex constraint set $\mathcal{W}$. One step of the algorithm is to solve (13) which is convex and therefore it can be done by any standard solver. Nevertheless, since the problem has to be solved several times during the MM procedure, the computational cost can be significant.

Interestingly, for specific constraint sets we can derive algorithms that have a closed-form update and therefore do not require a solver. In particular, we consider the following convex set parametrized by u:

$$\mathcal{W}_u = \left\{\mathbf{w}\middle|\mathbf{w}^\top\mathbf{1}=1, \mathbf{0}\leq\mathbf{w}\leq u\mathbf{1}\right\}, \tag{14}$$

that is, we require the weights to be nonnegative, to have an upper bounds u, and their summation to be equal to one.

First, we state two results that will be useful in the derivation of the closed-form update algorithms. Consider an optimization problem of the following form:

$$\begin{aligned}\underset{\mathbf{w}}{\text{minimize}} \quad & \mathbf{w}^\top\mathbf{w} + \mathbf{q}^\top\mathbf{w}\\ \text{subject to} \quad & \mathbf{w}\in\mathcal{W}_u,\end{aligned} \tag{15}$$

where $\mathbf{q}\in\mathbb{R}^N$. The following propositions provide a waterfilling structured solution of (15), considering two special cases i.e., $u=1$ and $u<1$, respectively [31].

Proposition 1. [$\text{AS}_1(\mathbf{q})$] *The optimal solution of the optimization problem* (15) *with* $u=1$ *is*

$$\mathbf{w}^\star = \left(-\frac{1}{2}(\mu\mathbf{1}+\mathbf{q})\right)^+, \tag{16}$$

with

$$\mu = -\frac{\sum_{i\in\mathcal{A}} q_i + 2}{card(\mathcal{A})}, \tag{17}$$

and

$$\mathcal{A} = \left\{i\middle|\mu+q_i<0\right\}, \tag{18}$$

where $\mathcal{A}$ *can be determined in* $O(\log(N))$ *steps.*

Proof. See Appendix A. □

We refer to the iterative procedure of Proposition 1 as $\text{AS}_1(\mathbf{q})$ (Active-Set for $u=1$).

Proposition 2. [$\text{AS}_u(\mathbf{q})$] *The optimal solution of the optimization problem* (15) *with* $u<1$ *is*

$$\mathbf{w}^\star = \left(\min\left(-\frac{1}{2}(\mu\mathbf{1}+\mathbf{q}), u\mathbf{1}\right)\right)^+, \tag{19}$$

with

$$\mu = -\frac{\sum_{i\in\mathcal{B}_2} q_i + 2 - card(\mathcal{B}_1)2u}{card(\mathcal{B}_2)}, \tag{20}$$

and

$$\mathcal{B}_1 = \left\{i\middle|\mu+q_i\leq -2u\right\}, \tag{21}$$

$$\mathcal{B}_2 = \left\{i\middle|-2u<\mu+q_i<0\right\}, \tag{22}$$

where $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *can be determined in* $O(N\log(N))$ *steps.*

Proof. See Appendix B. □

We refer to the iterative procedure of Proposition 2 as $\text{AS}_u(\mathbf{q})$ (Active-Set for general $u<1$). For simplicity we have considered the simple case where $\mathbf{u}=u\mathbf{1}$, however the extension to a general $\mathbf{u}$ is trivial and the differences are mentioned in Appendix B.

Notice that if we set $u=1$, AS_1 and AS_u do not become the same algorithm although they will return the same optimal solution. Further, a good practice would be to use a smart initialization of the sets $\mathcal{B}_1$ and $\mathcal{B}_2$ based on the solution of AS_1. A more extensive discussion about the iterative algorithms of Propositions 1 and 2 can be found in Appendix A and B, respectively. We will illustrate the benefit of AS_1 and AS_u (with and without initialization) for solving (15) in the numerical results of Section VII.

Now, let us return to the optimization problem (13). By expanding the norm of the objective function and dropping the constants we can rewrite the optimization problem as follows:

$$\begin{aligned}\underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{T}\mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w} + \left(\lambda\mathbf{d}_{p,u}^{(k)} - \frac{2}{T}\mathbf{X}^\top\mathbf{r}^b\right)^\top\mathbf{w}\\ \text{subject to} \quad & \mathbf{w}\in\mathcal{W}_u.\end{aligned} \tag{23}$$

In order to get a closed-form update algorithm as in (15), we need to majorize the quadratic term and decouple the variables.

Lemma 2. *[32, Lemma 1] Let* $\mathbf{L}$ *be a real symmetric matrix and* $\mathbf{M}$ *another real symmetric matrix such that* $\mathbf{M}\succeq\mathbf{L}$*. Then,*

Algorithm 2 SLAIT - Specialized LAIT for (7) with $\mathcal{W} = \mathcal{W}_u$

1: Set $k = 0$, choose $\mathbf{w}^{(0)} \in \mathcal{W}_u$
2: Compute $\lambda_{\max}^{(\mathbf{L}_1)}$
3: **repeat**:
4: Compute $\mathbf{q}_1^{(k)}$ according to (25), (11) and (9)
5: Solve (24) with $\text{AS}_{1|u}(\mathbf{q}_1)$ from Propositions 1,2 and set the optimal solution as $\mathbf{w}^{(k+1)}$
6: $k \leftarrow k + 1$
7: **until** convergence
8: **return** $\mathbf{w}^{(k)}$

for any point $\mathbf{w}^{(k)} \in \mathbb{R}^N$*, the quadratic function* $\mathbf{w}^\top \mathbf{L}\mathbf{w}$ *is majorized at* $\mathbf{w}^{(k)}$ *by* $\mathbf{w}^\top \mathbf{M}\mathbf{w} + 2\mathbf{w}^\top (\mathbf{L} - \mathbf{M})\mathbf{w}^{(k)} + {\mathbf{w}^{(k)}}^\top (\mathbf{M} - \mathbf{L})\mathbf{w}^{(k)}$.

Based on Lemma 2, if we set $\mathbf{L}_1 = \frac{1}{T}\mathbf{X}^\top\mathbf{X}$, and $\mathbf{M}_1 = \lambda_{\max}^{(\mathbf{L}_1)}\mathbf{I}$, then $\mathbf{M}_1 \succeq \mathbf{L}_1$ holds and the quadratic term of (23) can be majorized at $\mathbf{w}^{(k)}$ by:

$$\begin{aligned}\mathbf{w}^\top \mathbf{L}_1 \mathbf{w} \leq & \mathbf{w}^\top \mathbf{M}_1 \mathbf{w} + 2\mathbf{w}^\top (\mathbf{L}_1 - \mathbf{M}_1)\mathbf{w}^{(k)} + const \\ = & \lambda_{\max}^{(\mathbf{L}_1)} \mathbf{w}^\top \mathbf{w} + 2\mathbf{w}^\top \left(\mathbf{L}_1 - \lambda_{\max}^{(\mathbf{L}_1)}\mathbf{I}\right)\mathbf{w}^{(k)} + const\end{aligned}$$

Now, after dropping the constant terms, the new optimization problem at the $(k+1)$-th iteration becomes:

$$\begin{array}{ll}\underset{\mathbf{w}}{\text{minimize}} & \mathbf{w}^\top \mathbf{w} + {\mathbf{q}_1^{(k)}}^\top \mathbf{w} \\ \text{subject to} & \mathbf{w} \in \mathcal{W}_u,\end{array} \tag{24}$$

where

$$\mathbf{q}_1^{(k)} = \frac{1}{\lambda_{\max}^{(\mathbf{L}_1)}}\left(2\left(\mathbf{L}_1 - \lambda_{\max}^{(\mathbf{L}_1)}\mathbf{I}\right)\mathbf{w}^{(k)} + \lambda \mathbf{d}_{p,u}^{(k)} - \frac{2}{T}\mathbf{X}^\top \mathbf{r}^b\right) \tag{25}$$

is a constant depending on $\mathbf{w}^{(k)}$. The optimization problem (24) is now in the form of (15) and can be solved via $\text{AS}_{1|u}(\mathbf{q}_1)$, where $\text{AS}_{1|u}(\cdot)$ means $\text{AS}_1(\cdot)$ or $\text{AS}_u(\cdot)$.

Algorithm 2 summarizes the overall iterative procedure to solve (7) with $\mathcal{W} = \mathcal{W}_u$. We refer to it as SLAIT.

E. Computational Complexity

In this section we study the computational complexity of the proposed algorithms LAIT and SLAIT.

First we consider the LAIT algorithm. In every iteration we need to compute the vector $\mathbf{d}_{p,u}^{(k)}$ which can be done in $O(N)$ operations. Then, in the general case, we need to solve a quadratically constrained quadratic program (QCQP) (depending on the constraints it may reduce to a quadratic program (QP)). These problems can be reformulated as second order cone programs (SOCP) with complexity $O(N^{3.5}\log(1/\delta))$ per iteration, using the NT direction [33], where δ is the accepted duality gap. Thus, keeping only the higher order terms, LAIT has an overall complexity of $O\left(N_{\text{iter}} N^{3.5}\log(1/\delta)\right)$, where N_{iter} is the number of iterations.

In the case of the SLAIT algorithm we do not need to call a solver. First, we need to compute $\mathbf{L}_1$ and $\lambda_{\max}^{(\mathbf{L}_1)}$ in $O(N^2T)$ and $O(N^2)$ operations, respectively. Then, in every iteration we need to compute $\mathbf{d}_{p,u}^{(k)}$ $(O(N))$, perform a matrix-vector multiplication $(O(N^2))$ and finally some vector additions $(O(N))$ in order to obtain $\mathbf{q}_1^{(k)}$. The last step is to find the next iterate point which can be efficiently computed by the proposed algorithms AS_1 or AS_u in $O(\log(N))$ or $O(N\log(N))$ steps, where the complexity of each step is $O(N)$. Thus, again keeping the higher order terms, the overall complexity of the algorithm is $O\left(N^2T + N_{\text{iter}}N^2\log(N)\right)$.

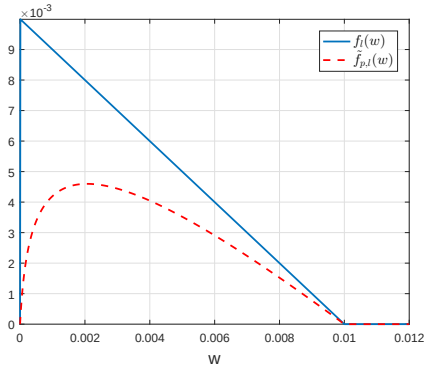


Fig. 2: Penalty functions $f_l(w)$ and $\tilde{f}_{p,l}(w)$ for $l = 0.01$ and $p = 10^{-4}$.

IV. Sparse Index Tracking with Holding Constraints

In this section we revisit problem (4) assuming a general set of convex constraints $\mathcal{W}$ and the non-convex holding constraints that we indicate separately. Again, we approximate the ℓ_0-"norm" by the continuous differentiable function $\rho_{p,\gamma}$. Now, the problem can be reformulated as follows:

$$\begin{array}{ll}\underset{\mathbf{w}}{\text{minimize}} & \frac{1}{T}\left\|\mathbf{X}\mathbf{w} - \mathbf{r}^b\right\|_2^2 + \lambda \mathbf{1}^\top \boldsymbol{\rho}_{p,u}(\mathbf{w}) \\ \text{subject to} & \mathbf{w} \in \mathcal{W}, \\ & \mathbf{l} \odot \boldsymbol{\mathcal{I}}_{\{\mathbf{w}>\mathbf{0}\}} \leq \mathbf{w} \leq \mathbf{u} \odot \boldsymbol{\mathcal{I}}_{\{\mathbf{w}>\mathbf{0}\}}.\end{array} \tag{26}$$

Here, we have used the notation $\boldsymbol{\mathcal{I}}_{\{\mathbf{w}>\mathbf{0}\}} = [\mathcal{I}_{\{w_1>0\}}, \ldots, \mathcal{I}_{\{w_N>0\}}]^\top$. Notice that the upper bound constraint $\mathbf{w} \leq \mathbf{u} \odot \boldsymbol{\mathcal{I}}_{\{\mathbf{w}>\mathbf{0}\}}$ is equivalent to $\mathbf{w} \leq \mathbf{u}$ and therefore it can be included in $\mathcal{W}$.

In the special case where $l = 0$, this problem becomes equivalent to (7) and the algorithms proposed in Section III can be used to solve it. Thus, we assume that $l > 0$.

A. Penalization of Constraint Violations

The lower bound constraint of (26) is non-convex and hard to deal with directly. Thus, instead of this constraint we can include an additional term in the objective that penalizes all the non-zero w_i's that are less than l. Since the lower bound constraint is separable for each w_i we can use a penalty function that penalizes each w_i independently. A suitable penalty function for a general entry w is the following:

$$f_l(w) = \left(\mathcal{I}_{\{0<w<l\}} \cdot l - w\right)^+. \tag{27}$$

Again, we can approximate the indicator function with $\rho_{p,\gamma}(w)$, given in (6). Since we are interested for the interval $[0, l]$ we select $\gamma = l$. We define the approximate penalty function as:

$$\tilde{f}_{p,l}(w) = \left(\rho_{p,l}(w) \cdot l - w\right)^+. \tag{28}$$

Algorithm 3 LAITH - Linear Approximation for the Index Tracking problem with Holding constraints (29)

1: Set $k = 0$, choose $\mathbf{w}^{(0)} \in \mathcal{W}$
2: **repeat**:
3: Compute $\mathbf{d}_{p,l}^{(k)}, \mathbf{d}_{p,u}^{(k)}$ according to (11) and (9)
4: Compute $\mathbf{c}_{p,l}^{(k)}$ according to (12) and (10)
5: Solve (31) and set the optimal solution as $\mathbf{w}^{(k+1)}$
6: $k \leftarrow k + 1$
7: **until** convergence
8: **return** $\mathbf{w}^{(k)}$

In Figure 2 we illustrate $f_l(w)$ and $\tilde{f}_{p,l}(w)$ for $l = 0.01$.

Now, we include the additional penalty term in the objective and the new optimization problem becomes:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{T}\left\|\mathbf{X}\mathbf{w} - \mathbf{r}^b\right\|_2^2 + \lambda \mathbf{1}^\top \boldsymbol{\rho}_{p,u}(\mathbf{w}) + \boldsymbol{\nu}^\top \tilde{\mathbf{f}}_{p,l}(\mathbf{w}) \\ \text{subject to} \quad & \mathbf{w} \in \mathcal{W}, \end{aligned} \tag{29}$$

where $\boldsymbol{\nu}$ is a parameter vector that controls the penalization of the weights that violate the lower bound, and $\tilde{\mathbf{f}}_{p,l}(\mathbf{w}) = [\tilde{f}_{p,l}(w_1), \ldots, \tilde{f}_{p,l}(w_N)]^\top$. This problem is not convex since $\rho_{p,u}(w)$ is concave and $\tilde{f}_{p,l}(w)$ is neither convex nor concave as shown in Figure 2.

Let us first focus on the third term of the objective, i.e., the function $\tilde{f}_{p,l}(\mathbf{w})$. Again, since the function is separable we need to deal with the univariate case only.

Lemma 3. *The function $\tilde{f}_{p,l}(w)$ is majorized at $w^{(k)} \in [0, u]$ by the convex function*

$$h_{p,l}(w, w^{(k)}) = \left(\left(d_{p,l}(w^{(k)}) \cdot l - 1\right) w + c_{p,l}(w^{(k)}) \cdot l\right)^+, \tag{30}$$

where $d_{p,l}(w^{(k)})$ and $c_{p,l}(w^{(k)})$ are given by (9) *and* (10)*, respectively.*

Proof. From Lemma 1 we have that $\rho_{p,l}(w) \leq d_{p,l}(w^{(k)})w + c_{p,l}(w^{(k)})$ for $w \geq 0$. Thus, for $\tilde{f}_{p,l}(w)$ we get:

$$\begin{aligned} \tilde{f}_{p,l}(w) &= \max\left(\rho_{p,l}(w) \cdot l - w, 0\right) \\ &\leq \max\left((d_{p,l}(w^{(k)})w + c_{p,l}(w^{(k)})) \cdot l - w, 0\right) \\ &= \max\left(\left(d_{p,l}(w^{(k)}) \cdot l - 1\right) w + c_{p,l}(w^{(k)}) \cdot l, 0\right). \end{aligned}$$

This function is convex since it is the maximum of two convex (actually affine) functions, i.e., $f_1 = \left(d_{p,l}(w^{(k)}) \cdot l - 1\right) w + c_{p,l}(w^{(k)}) \cdot l$ and $f_2 = 0$. □

The second term of (29) can be majorized with the linear majorization function presented in Lemma 1. The optimization problem at the $(k+1)$-th iteration becomes:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{T}\left\|\mathbf{X}\mathbf{w} - \mathbf{r}^b\right\|_2^2 + \lambda {\mathbf{d}_{p,u}^{(k)}}^\top \mathbf{w} \\ & + \boldsymbol{\nu}^\top \left(\text{Diag}\left(\mathbf{d}_{p,l}^{(k)} \odot \mathbf{l} - \mathbf{1}\right) \mathbf{w} + \mathbf{c}_{p,l}^{(k)} \odot \mathbf{l}\right)^+ \\ \text{subject to} \quad & \mathbf{w} \in \mathcal{W}, \end{aligned} \tag{31}$$

where $\mathbf{d}_{p,u}^{(k)}$, $\mathbf{d}_{p,l}^{(k)}$ are given by (11), and $\mathbf{c}_{p,l}^{(k)}$ by (12).

Algorithm 3 summarizes the above iterative procedure. We refer to it as LAITH.

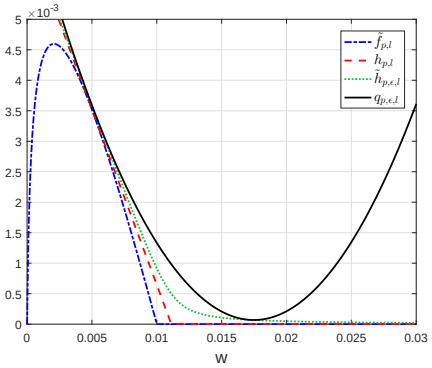


Fig. 3: Penalty function $\tilde{f}_{p,l}(w)$, linear ($h_{p,l}(w)$) and smooth linear ($\tilde{h}_{p,\epsilon,l}(w)$) upper bounds of $\tilde{f}_{p,l}(w)$, and quadratic upper bound $q_{p,\epsilon,l}(w)$, for $l = 0.01$, $\epsilon = 10^{-3}$ and $p = 10^{-4}$.

B. Specialized Algorithms

As in the case presented in Section III-D, if we restrict the constraint set $\mathcal{W}$ for problem (31) we can derive algorithms that have a closed-form update and therefore do not require a solver. Here, we consider the same set $\mathcal{W}_u$ given by (14).

To get a closed-form update algorithm we need to majorize the objective of (31) and transform it into an appropriate form. Let us begin with the majorization of the third term of the objective. This term is separable and therefore we need to focus only in the univariate case, i.e., in the function $h_{p,l}(w, w^{(k)})$ as defined in Lemma 3. However, $h_{p,l}(w, w^{(k)})$ is not smooth due to the max operator. Thus, a smooth majorization function cannot be defined at the point where $h_{p,l}(w, w^{(k)})$ is not smooth due to the non-differentiability of $h_{p,l}(w, w^{(k)})$ [34]. To this end, we will use a smooth approximation of the function $(x)^+$ defined as:

$$\frac{x + \sqrt{x^2 + \epsilon^2}}{2},$$

where $0 < \epsilon \ll 1$ controls the approximation. Applying this result in $h_{p,l}(w, w^{(k)})$, we can define its smooth version as:

$$\tilde{h}_{p,\epsilon,l}(w, w^{(k)}) = \frac{\alpha^{(k)} w + \beta^{(k)} + \sqrt{(\alpha^{(k)} w + \beta^{(k)})^2 + \epsilon^2}}{2}, \tag{32}$$

where $\alpha^{(k)} = d_{p,l}(w^{(k)}) \cdot l - 1$, and $\beta^{(k)} = c_{p,l}(w^{(k)}) \cdot l$.

Lemma 4. *The function $\tilde{h}_{p,\epsilon,l}(w, w^{(k)})$ is majorized at $w^{(k)}$ by the quadratic convex function $q_{p,\epsilon,l}(w, w^{(k)}) = a_{p,\epsilon,l}(w^{(k)})w^2 + b_{p,\epsilon,l}(w^{(k)})w + c_{p,\epsilon,l}(w^{(k)})$, where*

$$a_{p,\epsilon,l}(w^{(k)}) = \frac{(\alpha^{(k)})^2}{2\kappa_2}, \tag{33}$$

$$b_{p,\epsilon,l}(w^{(k)}) = \frac{\alpha^{(k)}\beta^{(k)}}{\kappa_2} + \frac{\alpha^{(k)}}{2}, \tag{34}$$

and $c_{p,\epsilon,l}(w^{(k)}) = \frac{(\alpha^{(k)}w^{(k)})(\alpha^{(k)}w^{(k)} + 2\beta^{(k)}) + 2({\beta^{(k)}}^2 + \epsilon^2)}{2\kappa_2} + \frac{\beta^{(k)}}{2}$ is an optimization irrelevant constant, with $\kappa_2 = 2\sqrt{(\alpha^{(k)}w^{(k)} + \beta^{(k)})^2 + \epsilon^2}$.

Proof. See Appendix C. □

In Figure 3 we illustrate this majorization procedure with all the intermediate steps.

Algorithm 4 SLAITH - Specialized LAITH for (29) with $\mathcal{W} = \mathcal{W}_u$

Set $k = 0$, choose $\mathbf{w}^{(0)} \in \mathcal{W}_u$
Compute $\lambda_{\max}^{(\mathbf{L}_2)}$
repeat:
  Compute $\mathbf{q}_2^{(k)}$ according to (37), (11) and (9)
  Solve (36) with $\text{AS}_{1|u}(\mathbf{q}_2)$ from Propositions 1,2 and set the optimal solution as $\mathbf{w}^{(k+1)}$
  $k \leftarrow k + 1$
until convergence
return $\mathbf{w}^{(k)}$

For notational convenience we define:

$$\mathbf{a}_{p,\epsilon,l}^{(k)} = [a_{p,\epsilon,l}(w_1^{(k)}), \ldots, a_{p,\epsilon,l}(w_N^{(k)})]^\top,$$
$$\mathbf{b}_{p,\epsilon,l}^{(k)} = [b_{p,\epsilon,l}(w_1^{(k)}), \ldots, b_{p,\epsilon,l}(w_N^{(k)})]^\top.$$

Now, using the quadratic majorizer of Lemma 4 and expanding the norm, we can rewrite the optimization problem as follows:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \mathbf{w}^\top \left(\frac{1}{T}\mathbf{X}^\top\mathbf{X} + \text{Diag}\left(\mathbf{a}_{p,\epsilon,l}^{(k)} \odot \boldsymbol{\nu}\right)\right)\mathbf{w} \\ & + \left(\lambda \mathbf{d}_{p,u}^{(k)} - \frac{2}{T}\mathbf{X}^\top\mathbf{r}^b + \mathbf{b}_{p,\epsilon,l}^{(k)} \odot \boldsymbol{\nu}\right)^\top \mathbf{w} \\ \text{subject to} \quad & \mathbf{w} \in \mathcal{W}_u. \end{aligned} \tag{35}$$

Again, in order to get a closed-form update algorithm we need to majorize the quadratic term of the objective and decouple the variables. Following similar arguments as in Section III-D, we set $\mathbf{L}_2 = \frac{1}{T}\mathbf{X}^\top\mathbf{X} + \text{Diag}\left(\mathbf{a}_{p,\epsilon,l}^{(k)} \odot \boldsymbol{\nu}\right)$, and $\mathbf{M}_2 = \lambda_{\max}^{(\mathbf{L}_2)}\mathbf{I}$. The new optimization problem at the $(k+1)$-th iteration becomes:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \mathbf{w}^\top\mathbf{w} + {\mathbf{q}_2^{(k)}}^\top\mathbf{w} \\ \text{subject to} \quad & \mathbf{w} \in \mathcal{W}_u, \end{aligned} \tag{36}$$

where

$$\begin{aligned} \mathbf{q}_2^{(k)} = \frac{1}{\lambda_{\max}^{(\mathbf{L}_2)}} \Bigg(& 2\left(\mathbf{L}_2 - \lambda_{\max}^{(\mathbf{L}_2)}\mathbf{I}\right)\mathbf{w}^{(k)} + \lambda\mathbf{d}_{p,u}^{(k)} \\ & - \frac{2}{T}\mathbf{X}^\top\mathbf{r}^b + \mathbf{b}_{p,\epsilon,l}^{(k)} \odot \boldsymbol{\nu}\Bigg) \end{aligned} \tag{37}$$

is a constant depending on $\mathbf{w}^{(k)}$. The optimization problem (36) is in the form of (15) and can be solved via $\text{AS}_{1|u}(\mathbf{q}_2)$ from Propositions 1 and 2.

Algorithm 4 summarizes the overall iterative procedure to solve (31) with $\mathcal{W} = \mathcal{W}_u$. We refer to it as SLAITH.

C. Computational Complexity

In this section we study the computational complexity of the proposed algorithms LAITH and SLAITH. The analysis is very similar to the computational complexity analysis of the LAIT and SLAIT algorithms in Section III-E. Therefore we will highlight only the differences.

The overall complexity of LAITH is the same as the complexity of LAIT, i.e., $O\left(N_{\text{iter}}N^{3.5}\log(1/\delta)\right)$, since in each iteration we need to compute $\mathbf{d}_{p,l}^{(k)}$, $\mathbf{d}_{p,u}^{(k)}$ and $\mathbf{c}_{p,l}^{(k)}$ in $O(N)$, and then solve an SOCP.

Algorithm	Complexity
LAIT/LAITH	$O\left(N_{\text{iter}}N^{3.5}\log(1/\delta)\right)$
SLAIT/SLAITH	$O\left(N^2T + N_{\text{iter}}N^2\log(N)\right)$

TABLE I: Complexity of the proposed algorithms.

Similarly, the complexity of SLAITH is the same as the complexity of SLAIT, i.e., $O\left(N^2T + N_{\text{iter}}N^2\log(N)\right)$. It requires some more vector computations and vector-vector multiplications but these are lower order operations that do not affect the complexity order.

In Table I we summarize the complexity of all algorithms.

V. Extensions

In this section we provide a digression on different design choices that can be made for the index tracking problem.

Throughout the paper we have considered the index tracking problem with a set of general convex constraints $\mathcal{W}$. We have analyzed some special cases of $\mathcal{W}$, i.e., the set $\mathcal{W}_u$ defined in (14) and the non-convex holding constraints, providing specific algorithms. However, the list of possible constraints that a fund manager may impose on the index tracking problem is not long, making the specialized algorithms very useful in practice.

Apart from the constraints mentioned already, i.e., budget ($\mathbf{w}^\top\mathbf{1} = 1$), no short selling ($\mathbf{w} \geq \mathbf{0}$) and the non-convex holding constraints, we may wish to mitigate the weights for some groups of stocks (e.g., sectors), or respect the index sector repartition for the selected subset of stocks. These are all linear constraints that do not need any special consideration. Finally, there is one more commonly used convex constraint, namely the turnover constraint: $\|\bar{\mathbf{w}} - \mathbf{w}\|_1 \leq C$. Here, $\bar{\mathbf{w}}$ refers to the tracking portfolio in the previous time period, and C is an upper bound on the total change in the portfolio between two consecutive periods [10], [18]. The purpose of the turnover constraint is to control the transaction costs when we rebalance our portfolio. Nevertheless, this constraint is more meaningful in a dynamic tracking setting and it does not add any difficulty into the problem since it is already convex.

Another design choice that a fund manager has to make is the selection of an appropriate tracking measure. We have considered the empirical tracking error (ETE), defined in (1), which is a commonly used measure for static index tracking. However, there are many other choices that come naturally for the index tracking problem [18]. For example, another common measure is the downside risk (DR) relative to an index which is defined as:

$$\text{DR}(\mathbf{w}) = \frac{1}{T}\left\|(\mathbf{r}^b - \mathbf{X}\mathbf{w})^+\right\|_2^2, \tag{38}$$

that is, we are interested in minimizing the tracking error only when the index beats our portfolio. The downside risk is a convex function and therefore we can easily replace the ETE term in problems (7) and (29) and all the derivations for the general algorithms will still hold.

Interestingly, if we consider the set $\mathcal{W}_u$, specialized algorithms can be derived for the DR too, as described next.

Lemma 5. *The function DR*$(\mathbf{w})$ *is majorized at* $\mathbf{w}^{(k)}$ *by the quadratic convex function* $\frac{1}{T}\left\|\mathbf{r}^b - \mathbf{X}\mathbf{w} - \mathbf{y}^{(k)}\right\|_2^2$, *where*

$$\mathbf{y}^{(k)} = -\left(\mathbf{X}\mathbf{w}^{(k)} - \mathbf{r}^b\right)^+. \tag{39}$$

Proof. See Appendix D. □

Now, we can follow the same procedure as in the ETE case, i.e., expand the ℓ_2-norm and majorize the quadratic term based on Lemma 2. In the case where we do not include the holding constraints, the optimization problem at the $(k+1)$-th iteration can be written in the form of (15), with

$$\mathbf{q}_3^{(k)} = \frac{1}{\lambda_{\max}^{(\mathbf{L}_3)}}\left(2\left(\mathbf{L}_1 - \lambda_{\max}^{(\mathbf{L}_1)}\mathbf{I}\right)\mathbf{w}^{(k)} + \lambda\mathbf{d}_{p,u}^{(k)} + \frac{2}{T}\mathbf{X}^\top(\mathbf{y}^{(k)} - \mathbf{r}^b)\right). \tag{40}$$

Therefore it can be solved efficiently by the proposed SLAIT algorithm where we need to compute $\mathbf{q}_3$ instead of $\mathbf{q}_1$. If we include the holding constraints, the problem at the $(k+1)$-th iteration can be written in the form of (15), with

$$\mathbf{q}_4^{(k)} = \frac{1}{\lambda_{\max}^{(\mathbf{L}_4)}}\left(2\left(\mathbf{L}_2 - \lambda_{\max}^{(\mathbf{L}_2)}\mathbf{I}\right)\mathbf{w}^{(k)} + \lambda\mathbf{d}_{p,u}^{(k)} + \frac{2}{T}\mathbf{X}^\top(\mathbf{y}^{(k)} - \mathbf{r}^b) + \mathbf{b}_{p,\epsilon,l}^{(k)} \odot \boldsymbol{\nu}\right), \tag{41}$$

and we can use the proposed algorithm SLAITH where we need to compute $\mathbf{q}_4$ instead of $\mathbf{q}_2$.

Other possible tracking measures are the Value-at-Risk (VaR) relative to an index [35], and the conditional Value-at-Risk (CVaR) relative to an index [36]. If the returns follow an elliptical distribution, VaR and CVaR are convex functions and in fact admit a closed-form expression [37]. In the case of non-elliptical distributions, CVaR remains convex (although VaR does not), and can be approximated by sampling the distribution [36]. In general, CVaR is preferred more in practice since it is a coherent measure of risk while VaR is not [38]. Both VaR (for elliptical distributions) and CVaR (for any distribution) can be used in our framework and replace the ETE defined in (1). CVaR can be included also as a convex constraint instead of replacing ETE in the objective [39].

Apart from the various performance measures, we can select a different penalty function. In all the presented formulations we have used the ℓ_2-norm to penalize the differences in the portfolio and the index. However, in order to be more robust to outliers we could use the Huber penalty function defined as (for the scalar case):

$$\phi(x) = \begin{cases} x^2, & |x| \le M, \\ M(2|x| - M), & |x| > M, \end{cases} \tag{42}$$

where $M > 0$ is a parameter. We can majorize this function at a point x_0 with a quadratic function of the form $a_0x^2 + b_0$, where $(a_0, b_0) = (1, 0)$ for $|x_0| \le M$, and $(a_0, b_0) = (M/|x_0|, M(|x_0| - M))$ for $|x_0| > M$. The proof is straightforward and omitted. With this quadratic bound, we can use the Huber penalty instead of the ℓ_2-norm with some minor modifications in the derived algorithms. Similar arguments can be made for the ℓ_1-norm and many more penalty functions.

VI. Convergence Analysis and Acceleration

A. Convergence

All the algorithms presented in this paper are based on the majorization-minimization framework. Given the sequence of points $(\mathbf{x}^{(k)})_{k\in\mathbb{N}}$ generated by the algorithm, we know that the sequence of objective values evaluated at these points is non-increasing. Since the constraint sets in our problems are compact, the sequence of objective values is bounded. Thus, it is guaranteed to converge to a finite value. In this section, we will analyze the convergence property of the sequence $(\mathbf{x}^{(k)})_{k\in\mathbb{N}}$ generated by the algorithms.

A unified convergence proof can be established given that all the optimization problems satisfy a minimum set of conditions. In particular, we require that all the conditions presented in Section III-B and (8) hold, that the objective function f is continuous and bounded below, and the constraint set is convex. These conditions are met by all the optimization problems we focused on. Further, consider the following assumptions [30, Section II.C]:

1) The sublevel set $\text{lev}_{\le f(\mathbf{x}_0)} f := \{\mathbf{x} \in \mathcal{X} | f(\mathbf{x}) \le f(\mathbf{x}_0)\}$ is compact given that $f(\mathbf{x}_0) < +\infty$.
2) $f(\mathbf{x})$ and $g(\mathbf{x}|\mathbf{x}^{(k)})$ are continuously differentiable with respect to $\mathbf{x}$.
3) For all $\mathbf{x}^{(k)}$ generated by the algorithm, there exists $\gamma \ge 0$ such that $\forall \mathbf{x} \in \mathcal{X}$, we have

$$(\nabla g(\mathbf{x}|\mathbf{x}^{(k)}) - \nabla g(\mathbf{x}^{(k)}|\mathbf{x}^{(k)}))^\top(\mathbf{x} - \mathbf{x}^{(k)}) \le \gamma\|\mathbf{x} - \mathbf{x}^{(k)}\|^2.$$

Given this minimum set of requirements and assumptions, the following are guaranteed:

1) The sequence of points $(\mathbf{x}^{(k)})_{k\in\mathbb{N}}$ produced by the MM algorithms converges.
2) The objective value f is non-increasing and converges to a limit $f^\star$, where $f^\star$ is a stationary value.

Therefore, it is guaranteed that all the algorithms presented in this paper converge to a stationary point.

B. Acceleration

The derivation of all the proposed algorithms is based on the majorization-minorization framework. In order to obtain surrogate functions that can be easily solved in closed-form many terms of the original functions were majorized twice. This can possibly lead to loose bounds which translates to a significantly large number of iterations for the MM algorithms to convergence. Thus, in this section we describe an acceleration scheme, called SQUAREM, that can improve significantly the convergence speed of the proposed algorithms.

SQUAREM was originally proposed in [40] to accelerate EM algorithms. The general idea is to evaluate the next two points of an iterative algorithm, compute a step-length η, and combine them to produce a point that decreases the objective value more than the two single steps. Since MM is a generalization of EM and the update rule of MM is

Algorithm 5 Accelerated SLAIT

Set $k = 0$, choose $\mathbf{w}^{(0)} \in \mathcal{W}_u$
repeat:
  $\mathbf{w}_1 = F_{\text{SLAIT}}(\mathbf{w}^{(k)})$
  $\mathbf{w}_2 = F_{\text{SLAIT}}(\mathbf{w}_1)$
  $\mathbf{r} = \mathbf{w}_1 - \mathbf{w}^{(k)}$
  $\mathbf{v} = \mathbf{w}_2 - \mathbf{w}_1 - \mathbf{r}$
  Compute the step-length $\eta = -\frac{\|\mathbf{r}\|_2}{\|\mathbf{v}\|_2}$
  $\mathbf{w} = \mathbf{w}^{(k)} - 2\eta\mathbf{r} + \eta^2\mathbf{v}$
  $\mathbf{w} = \text{AS}_{1|u}(-2\mathbf{w})$ (projection)
  while SLAIT$(\mathbf{w}) >$ SLAIT$(\mathbf{w}^{(k)})$
    $\eta \leftarrow (\eta - 1)/2$
    $\mathbf{w} = \mathbf{w}^{(k)} - 2\eta\mathbf{r} + \eta^2\mathbf{v}$
    $\mathbf{w} = \text{AS}_{1|u}(-2\mathbf{w})$ (projection)
  end while
  $\mathbf{w}^{(k+1)} = \mathbf{w}$
  $k \leftarrow k + 1$
until convergence
return $\mathbf{w}^{(k)}$

just a fixed-point iteration like EM, we can easily apply the SQUAREM acceleration method to MM algorithms with minor modifications. Without loss of generality we will present the accelerated version only of Algorithm 2 (SLAIT). The accelerated version of Algorithm 4 follows in a similar manner.

We denote by $F_{\text{SLAIT}}(\cdot)$ the fixed-point iteration map of the SLAIT algorithm, i.e., $\mathbf{w}^{(k+1)} = F_{\text{SLAIT}}(\mathbf{w}^{(k)})$, and by SLAIT$(\mathbf{w}^{(k)})$ the value of the objective function of (7) at the k-th iteration. The general SQUAREM method can cause two possible problems to the MM algorithms. First, the updated point may violate the constraints. To solve this issue we can project to the feasible set which is equivalent to solving the following optimization problem:

$$\begin{aligned} \underset{\mathbf{z}}{\text{minimize}} \quad & \|\mathbf{z} - \mathbf{w}\|_2^2 \\ \text{subject to} \quad & \mathbf{z} \in \mathcal{W}_u, \end{aligned} \tag{43}$$

where $\mathbf{w}$ is the updated point and $\mathbf{z}$ the projected. By expanding the norm, the objective can be rewritten as $\mathbf{z}^\top\mathbf{z} - 2\mathbf{w}^\top\mathbf{z}$. This problem is in the form of (15) and can be solved efficiently by $\text{AS}_{1|u}(-2\mathbf{w})$ from Propositions 1 and 2. The second problem is that the acceleration may violate the descend property of the MM algorithm. Thus, a backtracking step is adopted halving the distance of the step-length η and -1. As $\eta \to -1$, SLAIT$(\mathbf{w}^{(k+1)}) \leq$ SLAIT$(\mathbf{w}^{(k)})$ is guaranteed to hold[6]. The accelerated SLAIT is summarized in Algorithm 5.

C. Sequential Decreasing Scheme

Throughout the paper we have used the function ρ_p as a proxy of the ℓ_0-"norm". The approximation is controlled by the parameter p and in particular, as $p \to 0$ we get $\rho_p \to \ell_0$. However, by setting a small value to p it is likely that the algorithm will get stuck to a local minimum [26]. To solve this issue we start with a large value of p, i.e., a "loose" approximation, and solve the corresponding optimization problem. Then, we sequentially decrease p, i.e., we "tighten" the approximation, and solve the problem again using the previous solution as an initial point. In practice we are interested only in the last, most "tight" problem.

A similar approach is followed for the penalty term presented in (28) and the parameter $\boldsymbol{\nu}$. We start with a small value of $\boldsymbol{\nu}$ and solve the corresponding optimization problem. If there are constraint violations we increase the value of $\boldsymbol{\nu}$ and solve the problem again using the previous solutions as an initial value. The algorithms terminate when there are no constraint violations.

[6]If $\eta = -1$ then $\mathbf{w}^{(k+1)} = F_{\text{SLAIT}}\left(F_{\text{SLAIT}}(\mathbf{w}^{(k)})\right)$, i.e., it becomes equivalent to just taking two normal steps in the non-accelerated algorithm. Thus, the non-increasing objective value is guaranteed by the MM properties.

Index	Data Period	T_{tr}	T_{tst}
S&P 500	01/01/10 - 31/12/15	252	252
Russell 2000	01/06/06 - 31/12/15	1000	252

TABLE II: Index information.

VII. Numerical Experiments

In this section we evaluate the performance of the proposed algorithms using historical data of the indices S&P 500 (Bloomberg ticker SPX:IND) and Russell 2000 (Bloomberg ticker RTY:IND). For both indices we use a rolling window where a training period T_{tr} is used to derive the optimal tracking portfolio and a testing period T_{tst} to approximate the index movement with the derived portfolio. The details of each index, i.e., the window sizes T_{tr} and T_{tst}, and the total data period T are presented in Table II.

All the experiments were performed on a PC with a 3.20GHz i5-4570 CPU and 8GB RAM.

A. Sparse Index Tracking

In this first experiment we compare the performance of the proposed algorithms LAIT and SLAIT. The solution of the optimization problem (3) given by the Gurobi solver for MIP problems, denoted as MIP_{Gur}, serves as the principal benchmark[7]. We further compare the proposed methods with the Hybrid Half-Thresholding algorithm [16], denoted as HHT, and the Diversity Method [7], denoted as $\text{DM}_{1/2}$, where the ℓ_p-"norm" approximation is used, with $p = 1/2$.

All the optimization steps of LAIT are evaluated using the MOSEK solver (SLAIT does not require a solver). For the MIP we chose the Gurobi solver since it is known for its good performance in mixed integer problems. The HHT algorithm (requires the minimization of a QP) is implemented using the function "quadprog" of Matlab which is also used by the authors of [16]. The ℓ_p-"norm" approximation of

[7]The MIP solution is optimal if the algorithm runs until full convergence. However, in practice one has to stop the MIP after a certain amount of time so the final solution may be suboptimal as seen later in the numerical results.

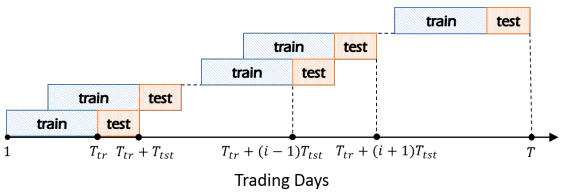
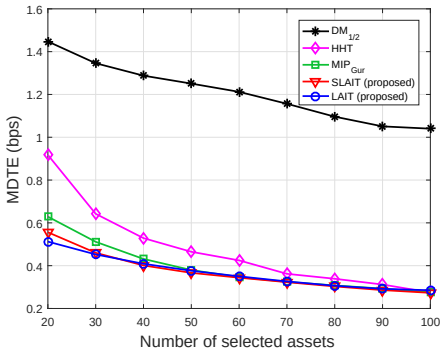
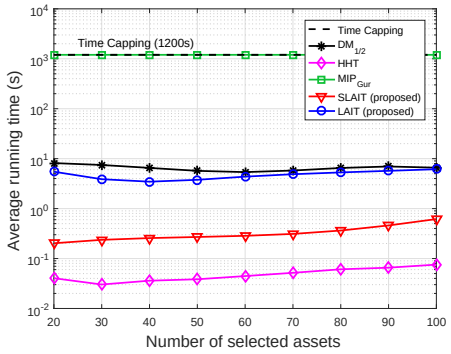


Fig. 4: Illustration of the rolling training and testing windows.

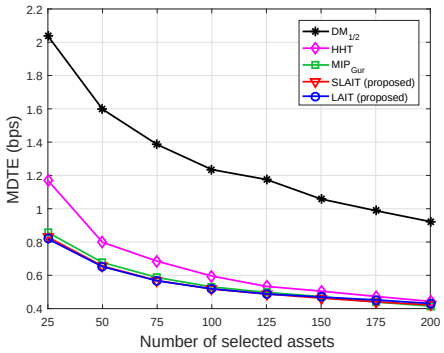


(a) Magnitude of daily tracking error.

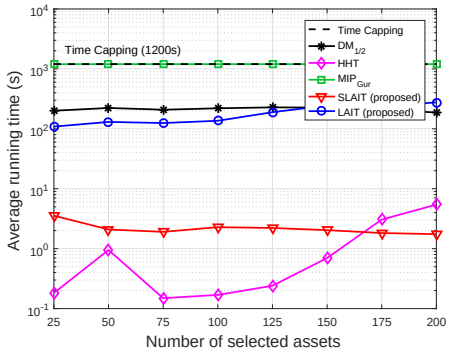


(b) Average running time.

Fig. 5: S&P 500: Comparison of the proposed algorithms LAIT, SLAIT with the benchmarks MIP_{Gur}, HHT and $\text{DM}_{1/2}$.



(a) Magnitude of daily tracking error.



(b) Average running time.

Fig. 6: Russell 2000: Comparison of the proposed algorithms LAIT, SLAIT with the benchmarks MIP_{Gur}, HHT and $\text{DM}_{1/2}$.

the diversity method is evaluated using the build-in function "fmincon" of Matlab. We keep the minimum constraint set $\mathcal{W}_u$ as defined in (14) for all algorithms, with $u = 0.05$. Finally, for practical reasons we have set the maximum running time of all algorithms to 1200 seconds.

Initially, we use the first T_{tr} days to design the portfolios, while we evaluate their performance in the next T_{tr} days. In the end of this testing period we need to redesign our portfolios. For this, we roll the training window and use the last T_{tr} days to design, and the next T_{tst} to evaluate the new portfolios. This scheme is shown pictorially in Figure 4. The total number of testing days is $T - T_{\text{tr}}$, i.e., we remove from the total data period T the initial window that we use only for training. Note that the portfolios we design at the beginning of each testing period do not remain constant during the testing period but they constantly change due to the price changes. To this end, for notational convenience we stack the tracking portfolios of all the testing days in a matrix[8] $\mathbf{W} \in \mathbb{R}_{+}^{N \times (T - T_{\text{tr}})}$.

First, we measure how close the proposed algorithms can replicate a given index. To this end, for a given sparsity level we compute the magnitude of the daily tracking error (MDTE) defined as:

$$\text{MDTE} = \frac{1}{T - T_{\text{tr}}} \left\| \text{diag}(\mathbf{X}\mathbf{W}) - \mathbf{r}^b \right\|_2, \tag{44}$$

where $\mathbf{X} \in \mathbb{R}^{(T - T_{\text{tr}}) \times N}$ and $\mathbf{r}^b \in \mathbb{R}^{T - T_{\text{tr}}}$. All the tracking error result are presented in basis points (bps)[9]. Apart from the tracking error, we further compute the average[10] running time of each algorithm for the different sparsity levels.

In Figures 5a and 6a we compare the tracking error of all the algorithms using the daily returns of the S&P 500 and Russell 2000, respectively. We observe that the proposed algorithms outperform significantly the HHT and $\text{DM}_{1/2}$ algorithms in terms of tracking error. Compared to MIP_{Gur}, the proposed algorithms perform slightly better for small cardinalities and have effectively the same MDTE for larger cardinalities.

The average running time of the algorithms is presented in Figures 5b and 6b for the two indices, respectively. We observe that all the algorithms apart from MIP_{Gur} need only a few seconds to converge. On the other hand, the MIP_{Gur} algorithm consumes always all the allowed running time.

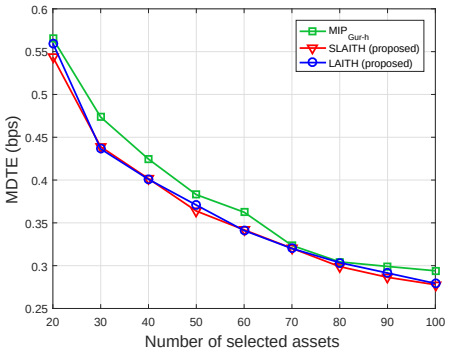
B. Sparse Index Tracking with Holding Constraints

Now, we consider the case where we have the non-convex holding constraints and we compare the algorithms LAITH and SLAITH. The solution of the optimization problem (4) given by the Gurobi solver for MIP problems, denoted as

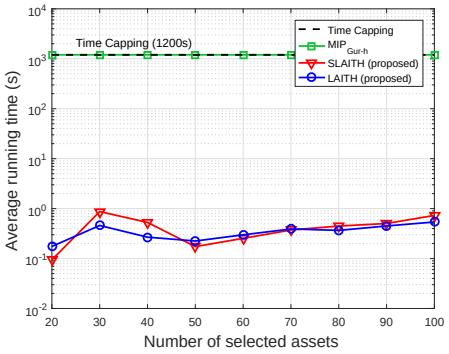
[8]The use of a matrix is just to present (44) in a compact form. It is possible to denote the portfolio at the t-th day as $\mathbf{w}_t$ and use an appropriate summation.

[9]One basis point is equal to 0.01%.

[10]For a fixed sparsity level, we need to design $\lceil \frac{T - T_{\text{tr}}}{T_{\text{tst}}} \rceil$ portfolios, one for each testing window. The time averaging is over these portfolios.

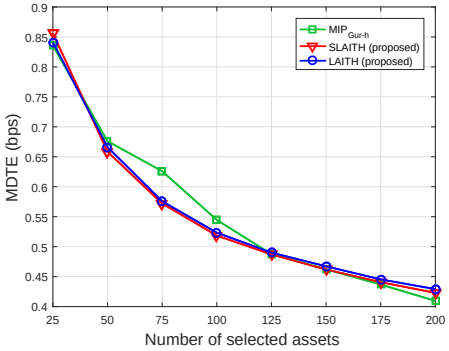


(a) Magnitude of daily tracking error.

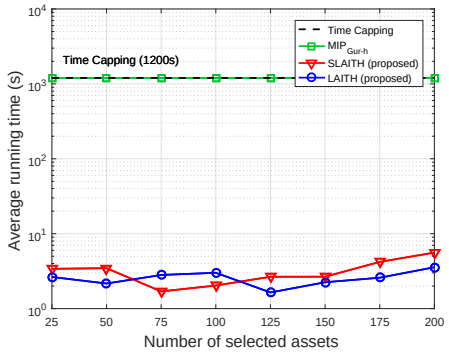


(b) Average running time.

Fig. 7: S&P 500: Comparison of the proposed algorithms LAITH, SLAITH with the benchmark $\text{MIP}_{\text{Gur-h}}$.



(a) Magnitude of daily tracking error.



(b) Average running time.

Fig. 8: Russell 2000: Comparison of the proposed algorithms LAITH, SLAITH with the benchmark $\text{MIP}_{\text{Gur-h}}$.

$\text{MIP}_{\text{Gur-h}}$, serves as the main benchmark. The algorithms HHT and $\text{DM}_{1/2}$ are not considered here since they are not suitable for non-convex constraints. Again, we keep the minimal constraint set $\mathcal{W}_u$ as defined in (14), with $u = 0.05$. For $\text{MIP}_{\text{Gur-h}}$ we further include the non-convex lower bound constraint with $l = 0.001$. The proposed algorithms take into account this constraints through the penalty term in the objective.

The comparison of the algorithms in terms of tracking error using the daily returns of the S&P 500 and Russell 2000 is given in Figures 7a and 8a, respectively. In this case, it is clear that the proposed algorithms outperform the $\text{MIP}_{\text{Gur-h}}$ algorithm in terms of tracking error. This is due to the fact that now the problem is more complex and an MIP solver needs substantially more time than 1200 seconds to reach to a good solution. In Figures 7b and 8b we illustrate the average running time of the algorithms. Again, LAITH and SLAITH converge orders of magnitude faster than $\text{MIP}_{\text{Gur-h}}$.

C. Downside Risk

In all the simulations up to this point, the proposed algorithms and all the benchmarks minimize the empirical tracking error (ETE), defined in (1). However, as shown in Section V, we can use any convex tracking error in the proposed algorithms. In particular, we examined in details the downside risk (DR) measure, defined in (38). The goal of DR is to replicate the index while avoiding too large drawdowns. Therefore, the tracking error of the DR portfolio becomes worse but the drawdowns are less severe (a crucial attribute especially for high leverage positions) which results to higher returns overall. To this end, the DR portfolio cannot be compared in a fair manner with the benchmarks that are designed to only replicate an index. Nevertheless, it is interesting to analyze the impact the different tracking measures have in the behavior of the designed portfolio. Therefore we examine the returns of portfolios with different tracking measures, taking into account the associated transaction costs.

First, we consider the following common transaction costs model[11] that applies in U.S. markets: the cost per transaction is $\$0.005 \cdot n$, where n is the number of shares we buy or sell, with minimum cost \$1 and maximum cost 0.5% of trade volume. Note that this cost applies for each transaction separately and each transaction is associated with the purchase or sell of only one asset. This shows the great advantage of sparse portfolios in terms of transaction costs. Finally, since the costs depend on our budget, we assume a budget of \$1 million.

We use the index S&P 500 for the period given in Table II (excluding again the first training window). We construct two tracking portfolios using the proposed SLAIT algorithm. In the first we use the ETE measure and in the second the DR

[11] https://www.interactivebrokers.com/en/index.php?f=commission&p=stocks

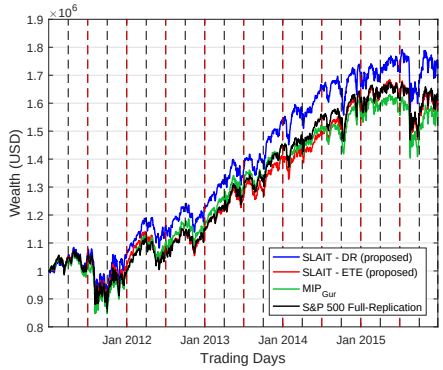


Fig. 9: Comparison of the out-of-sample returns of tracking portfolios. Each portfolio is composed by 40 assets apart from the full-replication portfolio. The vertical black dashed lines denote the redesign (and rebalancing) dates whereas the red ones denote only rebalancing.

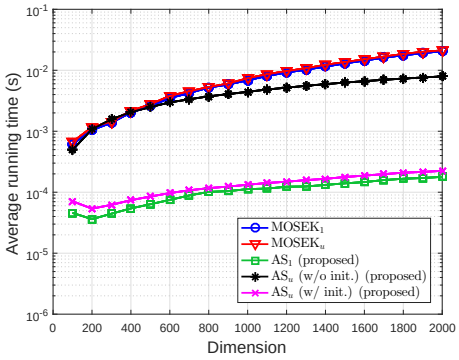


Fig. 10: Average running time of AS_1 and AS_u. Comparison with the algorithms MOSEK_1 and MOSEK_u that correspond to a direct implementation of (15) using the MOSEK solver for the cases where $u = 1$ and $u < 1$, respectively. Each curve is an average of 500 random trials.

measure. Each portfolio consist of only 40 assets. We consider a rebalancing frequency of 3 months and a redesign frequency of 6 months. The training window of the portfolios is again $T_{\text{tr}} = 252$ days. However, the testing window is up to the next rebalancing or redesign, therefore is set to 3 months.

Figure 9 illustrates the wealth of the portfolios that use different tracking measures. We have included a portfolio constructed by the algorithm MIP_{Gur} for reference. Further, we illustrate the performance of a portfolio replicating exactly the index S&P 500 (full-replication), without transaction costs. The vertical dashed red lines correspond to the rebalancing dates and the black ones to the redesign dates.

It is clear that the constructed portfolios are consistent to their objective. We observe that the SLAIT-ETE portfolio stays close to the full-replication portfolio (although we have subtracted the transaction costs). The same holds for the MIP_{Gur} portfolio, although overall it seems to diverge more than the SLAIT-ETE portfolio (however, there are some periods that is closer to the full-replication portfolio). This result is consistent to the one in Figure 5a, where for 40 assets we observe that the MIP_{Gur} portfolio has a slightly larger daily tracking error. Finally, the SLAIT-DR portfolio outperforms significantly the full-replication portfolio as we would expect.

Notice that the SLAIT-DR portfolio has a very large tracking error compared to SLAIT-ETE. However, it has a much lower downside risk. In the end, the tracking measures are just proxies for the real objective of a tracking portfolio, i.e., the return. It is obvious that in a real investment, the DR measure is in general more attractive.

D. Computational Complexity of $\text{AS}_{1|u}$

For the specialized algorithms SLAIT and SLAITH we have presented two closed-form update algorithms, i.e., AS_1 and AS_u, that solve the inner optimization problems (24) and (36) of the general form (15). Here, we compare the performance of these algorithms in terms of average running time with a direct implementation of (15) using the MOSEK solver.

For a given problem dimension N, we randomly generate 500 vectors $\mathbf{q} \in \mathbb{R}^N$. To test AS_1 we consider the constraint set $\mathcal{W}_u$ with $u = 1$, while for AS_u we set $u = 20/N$. We sequentially increase the problem dimension from $N = 100$ to $N = 2000$ to examine the scalability of the algorithms.

Figure 10 illustrates the average running time of AS_1 and AS_u (with and without initialization) compared to the MOSEK solver. It is clear that the proposed algorithms are more than one order of magnitude faster. Further, they scale well with dimension since their average running time increased less than half order of magnitude for $N = 100$ to $N = 2000$.

VIII. Conclusions

Index tracking, in all of its forms, requires efficient algorithms for the construction of tracking portfolios. This is a challenging task since the need for sparsity to reduce the costs, the requirement for low tracking error, and the practical constraint for low running time are in general opposing goals and hard to combine. In this paper we have derived fast and efficient algorithms for the high-dimensional sparse index tracking problem. The proposed algorithms consider a general convex set of constraints. We have further derived special algorithms that include the non-convex holding constraints. A general consideration was given in different tracking measures and especially in the downside risk. Numerical experiments have shown the superiority of the proposed algorithms since they combine two key attributes: they match or outperform (especially in the case of holding constraints) existing benchmarks in terms of tracking error and require a minimal running time to converge. These attributes, combined with the flexibility in tracking measures and constraints, make the proposed algorithms very attractive for practical use.

Appendix A
Proof of Proposition 1

In the case where $u = 1$ we can drop the constraint $\mathbf{w} \leq \mathbf{u}$ since it becomes implicit from the other two constraints, i.e., $\mathbf{w}^\top \mathbf{1} = 1$ and $\mathbf{w} \geq \mathbf{0}$. Further, without loss of generality, throughout this proof we will assume that $\mathbf{q}$ is sorted in ascending order, i.e., $q_i \leq q_j$ for $i < j$.

With this simplification, the Lagrangian of (15) is:

$$\mathcal{L}(\mathbf{w},\mu,\boldsymbol{\nu}) = \mathbf{w}^\top\mathbf{w} + \mathbf{q}^\top\mathbf{w} + \mu(\mathbf{w}^\top\mathbf{1} - 1) - \boldsymbol{\nu}^\top\mathbf{w}.$$

From the derivative of the Lagrangian we get:

$$w_i = \frac{1}{2}(\nu_i - \mu - q_i). \tag{45}$$

We identify three cases:

a) $\mu+q_i > 0$: It must hold that $\nu_i \geq \mu+q_i > 0$ since $w_i \geq 0$ (primal feasibility). Further, if $\nu_i > 0$ then necessarily $w_i = 0$ (complementary slackness).
b) $\mu + q_i < 0$: It must hold that $w_i > 0$ since $\nu_i \geq 0$ (dual feasibility). Further, since $w_i > 0$, it holds that $\nu_i = 0$ (complementary slackness) and therefore $w_i = -(\mu + q_i)/2$ (from (45)).
c) $\mu + q_i = 0$: The only solution is $w_i = \nu_i = 0$ (complementary slackness).

We can state this result more compactly as follows:

$$w_i = \begin{cases} 0, & \text{if } \mu + q_i \geq 0, \quad (46a) \\ -(\mu + q_i)/2, & \text{if } \mu + q_i < 0. \quad (46b) \end{cases}$$

Thus, for a given μ, only the w_i's corresponding to the smaller q_i's are not zero. Also, if $w_i > 0$, then $w_j > 0$ for all $j < i$.

Now, we need to find the optimal value of the dual variable μ. Assume we know that K_{opt} weights are positive, i.e., $\mathbf{w}_{[1:K_{\text{opt}}]} > \mathbf{0}$ and $\mathbf{w}_{[K_{\text{opt}}+1:N]} = \mathbf{0}$. From $\mathbf{w}^\top\mathbf{1} = 1$ (primal feasibility), substituting $\mathbf{w}$ given by (46a) and (46a) we get:

$$\mathbf{w}^\top\mathbf{1} = 1 \quad \Longrightarrow -\sum_{i=1}^{K_{\text{opt}}}(\mu + q_i)/2 = 1.$$

With some trivial term rearrangements we get the value of μ:

$$\mu = -\frac{\sum_{i=1}^{K_{\text{opt}}} q_i + 2}{K_{\text{opt}}}. \tag{47}$$

Therefore, a straightforward way to find the optimal solution is to start with $K = 1$ non-zero weights, compute μ from (47) and check the conditions (46a) and (46b). If they hold then $K = K_{\text{opt}}$, else we need to increase K.

However, it is not hard to prove that if $K < K_{\text{opt}}$, then $\mu + q_{K+1} < 0$ which violates (46a). Similarly if $K > K_{\text{opt}}$, then $\mu + q_K > 0$ which condition (46b)[12]. Thus, by knowing if K should be increased or decreased we can do a binary search to find K_{opt} that terminates in at most $\log(N)$ steps.

Appendix B
Proof of Proposition 2

Without loss of generality, we will assume that $\mathbf{c} = \mathbf{q} + 2\mathbf{u}$ is ordered in ascending order for a general upper bound $\mathbf{u}$, i.e., $c_i \leq c_j$ for $i < j$. Since this proof follows similar steps to the proof of Proposition 1 we will skip the details.

The Lagrangian of (15) is:

$$\begin{aligned}\mathcal{L}(\mathbf{w},\mu,\boldsymbol{\nu}_1,\boldsymbol{\nu}_2) =& \mathbf{w}^\top\mathbf{w} + \mathbf{q}^\top\mathbf{w} + \mu(\mathbf{1}^\top\mathbf{w} - 1) \\ & - \boldsymbol{\nu}_1^\top\mathbf{w} + \boldsymbol{\nu}_2^\top(\mathbf{w} - \mathbf{u}).\end{aligned}$$

[12]Start from $K = 1$. If $K < K_{\text{opt}}$ it should hold that $\mu + q_2 \geq 0$ (else $K_{\text{opt}} = 1$). Use this condition for $K = 2$ and proceed in the same way. It can be easily seen that if $K < K_{\text{opt}}$, (46b) always holds however (46a) cannot hold. The intuition for the case $K > K_{\text{opt}}$ is the same.

From the derivative of the Lagrangian we get:

$$w_i = \frac{1}{2}(\nu_{1,i} - \nu_{2,i} - \mu - q_i). \tag{48}$$

Considering the different cases we get the following:

$$w_i = \begin{cases} 0, & \text{if } \mu + c_i \geq 2u_i, \quad (49a) \\ -(\mu + q_i)/2, & \text{if } 0 < \mu + c_i < 2u_i, \quad (49b) \\ u_i, & \text{if } \mu + c_i < 0\ . \quad (49c) \end{cases}$$

These conditions state the following: for a given μ, a subset of w_i's that correspond to the smallest c_i's will take the maximum possible value u_i. Another subset of w_i's with small enough c_i's will take some non-zero value less than u_i. Finally, the w_i's that correspond to the larger c_i's will be zero.

Now, we need to determine the value of μ. Assume we know that $K_{1,\text{opt}}$ weights are equal to u_i and $K_{2,\text{opt}}$ weights are positive and less than u_i, i.e., $\mathbf{w}_{[1:K_{1,\text{opt}}]} = \mathbf{u}_{[1:K_{1,\text{opt}}]}$, $\mathbf{0} < \mathbf{w}_{[K_{1,\text{opt}}+1:K_{1,\text{opt}}+K_{2,\text{opt}}]} < \mathbf{u}_{[K_{1,\text{opt}}+1:K_{1,\text{opt}}+K_{2,\text{opt}}]}$ and $\mathbf{w}_{[K_{2,\text{opt}}+1:N]} = \mathbf{0}$. Then, from $\mathbf{w}^\top\mathbf{1} = 1$ (primal feasibility), the value of μ is:

$$\mu = -\frac{\sum_{i=K_{1,\text{opt}}+1}^{K_{1,\text{opt}}+K_{2,\text{opt}}} q_i - 2\sum_{i=1}^{K_{1,\text{opt}}} u_i + 2}{K_{2,\text{opt}}}. \tag{50}$$

In order to evaluate μ we need to determine $K_{1,\text{opt}}$ and $K_{2,\text{opt}}$. In a similar manner as in the proof of Proposition 1, we can start with $K = 1$ non-zero weights and sequentially increase its value until we find the optimal one. For a given value of K we can do a binary search to identify K_1 weights with maximum value and K_2 positive weights with a value less than the upper bound, where $K = K_1 + K_2$. Unfortunately, if the conditions (49a)-(49c) are not satisfied for the derived K_1 and K_2 we cannot determine if we need to increase or decrease the total number of non-zero weights K. This leads to one linear search with a binary search in each step with a combined complexity $O(N\log(N))$.

A better approach is to set $K = K_{\text{opt}}$, where K_{opt} is the number of positive weights in the case where $u = 1$ (see Appendix A), since by imposing an upper bound constraint there will be at least K_{opt} weights that are not zero.

An interesting point is that $K_{2,\text{opt}}$ cannot be zero since it is the denominator in (50). We know that $K_{1,\text{opt}} + K_{2,\text{opt}} > 0$ must hold, i.e., there is at least one non-zero weight, since $\mathbf{w}^\top\mathbf{1} = 1$. However, for $K_{2,\text{opt}} = 0$ we get $\mu = +\infty$ and from the conditions (49a)-(49c) we see that all the weights should be zero. Therefore, this observation shows that it is not possible all the non-zero weights to be equal to their upper bound u_i (since then $K_{2,\text{opt}} = 0$). Although theory shows that we cannot get a solution that all the non-zero weights are equal to their upper bounds, in practice this could happen if we have a group of some extremely negative c_i's while the rest of c_i's are much larger. In this case, due to roundoff errors and finite precision, a solution where all the non-zero weights are equal to their upper bounds is possible and it needs a special consideration.

Finally, in the special case where $\mathbf{u} = u\mathbf{1}$, μ becomes:

$$\mu = -\frac{\sum_{i=K_{1,\text{opt}}+1}^{K_{1,\text{opt}}+K_{2,\text{opt}}} q_i - 2K_{1,\text{opt}}u + 2}{K_{2,\text{opt}}}. \tag{51}$$

Further, note that in this case, sorting according to $\mathbf{c}$ is equivalent to sorting according to $\mathbf{q}$.

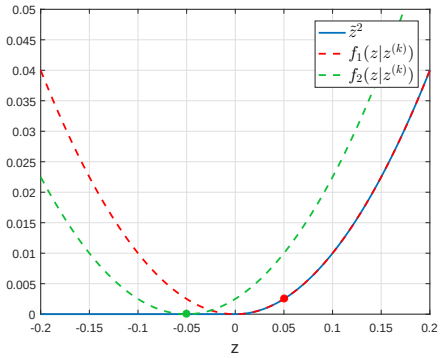


Fig. 11: Majorization cases of $\tilde{z}^2$: a) $z^{(k)} = 0.05 > 0$ and b) $z^{(k)} = -0.05 \leq 0$.

Appendix C
Proof of Lemma 4

Consider the concave function $f(x) = \sqrt{x}$ for $x \in [0, u]$. An upper bound of a concave function at any point x_0 is its first-order Taylor approximation, i.e.,

$$\sqrt{x} \leq \sqrt{x_0} + \frac{1}{2\sqrt{(x_0)}}(x - x_0).$$

By setting $x = (\alpha w + \beta)^2 + \epsilon^2$ we get the following bound:

$$\begin{aligned}
&\sqrt{(\alpha w + \beta)^2 + \epsilon^2} \\
\leq &\sqrt{(\alpha w_0 + \beta)^2 + \epsilon^2} + \frac{\alpha^2(w^2 - w_0^2) + 2\alpha\beta(w - w_0)}{2\sqrt{((\alpha w_0 + \beta)^2 + \epsilon^2)}} \\
= &\frac{\alpha^2 w^2 + 2\alpha\beta w}{2\sqrt{((\alpha w_0 + \beta)^2 + \epsilon^2)}} + const.
\end{aligned}$$

By majorizing the square root term of $\tilde{h}_{p,\epsilon,l}(w, w^{(k)})$ following the aforementioned approach, the result of Lemma 4 is straightforward.

Appendix D
Proof of Lemma 5

For convenience set $\mathbf{z} = \mathbf{r}^b - \mathbf{X}\mathbf{w}$. Then:

$$\mathrm{DR}(\mathbf{w}) = \frac{1}{T}\left\|(\mathbf{z})^+\right\|_2^2 = \frac{1}{T}\sum_{i=1}^{T}\tilde{z}_i^2,$$

where

$$\tilde{z}_i = \begin{cases} z_i, & \text{if } z_i > 0, \\ 0, & \text{if } z_i \leq 0. \end{cases}$$

Now, we can majorize each $\tilde{z}_i^2$ term to get an upper bound for $\mathrm{DR}(\mathbf{z})$. We need to consider two cases, i.e., majorization on a point $z_i^{(k)} > 0$ and on a point $z_i^{(k)} \leq 0$.

1) For a point $z_i^{(k)} > 0$, $f_1(z_i|z_i^{(k)}) = z_i^2$ is an upper bound of $\tilde{z}_i^2$, with $f_1(z_i^{(k)}|z_i^{(k)}) = \left(z_i^{(k)}\right)^2 = \left(\tilde{z}_i^{(k)}\right)^2$.
2) For a point $z_i^{(k)} \leq 0$, $f_2(z_i|z_i^{(k)}) = \left(z_i - z_i^{(k)}\right)^2$ is an upper bound of $\tilde{z}_i^2$, with $f_2(z_i^{(k)}|z_i^{(k)}) = \left(z_i^{(k)} - z_i^{(k)}\right)^2 = 0 = \left(\tilde{z}_i^{(k)}\right)^2$.

For both cases the proofs are straightforward and they are easily shown pictorially. Figure 11 illustrates these two cases.

Now, we can majorize $\tilde{z}_i^2$ at any point $z_i^{(k)}$ as follows:

$$\begin{aligned}
\tilde{z}_i^2 \leq &\begin{cases} f_1(z_i|z_i^{(k)}), & \text{if } z_i^{(k)} > 0, \\ f_2(z_i|z_i^{(k)}), & \text{if } z_i^{(k)} \leq 0, \end{cases} \\
= &\begin{cases} (z_i - 0)^2, & \text{if } z_i^{(k)} > 0, \\ (z_i - z_i^{(k)})^2, & \text{if } z_i^{(k)} \leq 0, \end{cases} \\
= &(z_i - y_i^{(k)})^2,
\end{aligned}$$

where

$$\begin{aligned}
y_i^{(k)} = &\begin{cases} 0, & \text{if } z_i^{(k)} > 0, \\ z_i^{(k)}, & \text{if } z_i^{(k)} \leq 0, \end{cases} \\
= &-(-z_i^{(k)})^+.
\end{aligned}$$

Thus, $\mathrm{DR}(\mathbf{z})$ is majorized as follows:

$$\mathrm{DR}(\mathbf{w}) = \frac{1}{T}\sum_{i=1}^{T}\tilde{z}_i^2 \leq \frac{1}{T}\sum_{i=1}^{T}(z_i - y_i^{(k)})^2 = \frac{1}{T}\left\|\mathbf{z} - \mathbf{y}^{(k)}\right\|_2^2.$$

Substituting back $\mathbf{z} = \mathbf{r}^b - \mathbf{X}\mathbf{w}$, we get

$$\mathrm{DR}(\mathbf{w}) \leq \frac{1}{T}\left\|\mathbf{r}^b - \mathbf{X}\mathbf{w} - \mathbf{y}^{(k)}\right\|_2^2.$$

where $\mathbf{y}^{(k)} = -(-\mathbf{z}^{(k)})^+ = -(\mathbf{X}\mathbf{w}^{(k)} - \mathbf{r}^b)^+$.

References

[1] B. M. Barber and T. Odean, "Trading is hazardous to your wealth: The common stock investment performance of individual investors," *The Journal of Finance*, vol. 55, no. 2, pp. 773–806, 2000.
[2] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.
[3] S. L. Fuller, "The evolution of actively managed exchange-traded funds," *The Review of Securities and Commodities Regulation*, vol. 41, no. 8, pp. 89–96, 2008.
[4] M. Kosev, T. Williams, *et al.*, "Exchange-traded funds," *RBA Bulletin*, pp. 51–59, Mar. 2011.
[5] A. A. Santos, "Beating the market with small portfolios: Evidence from brazil," *EconomiA*, vol. 16, no. 1, pp. 22–31, 2015.
[6] K. Naumenko and O. Chystiakova, "An empirical study on the differences between synthetic and physical ETFs," *International Journal of Economics and Finance*, vol. 7, no. 3, p. 24, 2015.
[7] R. Jansen and R. Van Dijk, "Optimal benchmark tracking with small portfolios," *The Journal of Portfolio Management*, vol. 28, no. 2, pp. 33–39, 2002.
[8] J. E. Beasley, N. Meade, and T.-J. Chang, "An evolutionary heuristic for the index tracking problem," *European Journal of Operational Research*, vol. 148, no. 3, pp. 621–643, 2003.
[9] D. Maringer and O. Oyewumi, "Index tracking with constrained portfolios," *Intelligent Systems in Accounting, Finance and Management*, vol. 15, no. 1-2, pp. 57–71, 2007.
[10] A. Scozzari, F. Tardella, S. Paterlini, and T. Krink, "Exact and heuristic approaches for the index tracking problem with UCITS constraints," *Annals of Operations Research*, vol. 205, no. 1, pp. 235–250, 2013.
[11] K. J. Oh, T. Y. Kim, and S. Min, "Using genetic algorithm to support portfolio optimization for index fund management," *Expert Systems with Applications*, vol. 28, no. 2, pp. 371–379, 2005.
[12] C. Dose and S. Cincotti, "Clustering of financial time series with application to index and enhanced index tracking portfolio," *Physica A: Statistical Mechanics and its Applications*, vol. 355, no. 1, pp. 145–151, 2005.
[13] D. Bianchi and A. Gargano, "High-dimensional index tracking with cointegrated assets using an hybrid genetic algorithm," *Manuscript available at: http://ssrn. com/abstract*, vol. 1785908, 2011.
[14] C. Alexander, "Optimal hedging using cointegration," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 357, no. 1758, pp. 2039–2058, 1999.

[15] Y. Feng and D. P. Palomar, "A signal processing perspective on financial engineering," *Foundations and Trends® in Signal Processing*, vol. 9, pp. 1–231, Aug. 2016.
[16] F. Xu, Z. Xu, and H. Xue, "Sparse index tracking based on $L_{1/2}$ model and algorithm," *arXiv preprint arXiv:1506.05867*, 2015.
[17] T. F. Coleman, Y. Li, and J. Henniger, "Minimizing tracking error while restricting the number of assets," *Journal of Risk*, vol. 8, no. 4, p. 33, 2006.
[18] A. A. Gaivoronski, S. Krylov, and N. Van der Wijst, "Optimal portfolio selection and dynamic benchmark tracking," *European Journal of Operational Research*, vol. 163, no. 1, pp. 115–131, 2005.
[19] R. Roll, "A mean/variance analysis of tracking error," *The Journal of Portfolio Management*, vol. 18, no. 4, pp. 13–22, 1992.
[20] H. Markowitz, "Portfolio selection*," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.
[21] J. Brodie, I. Daubechies, C. De Mol, D. Giannone, and I. Loris, "Sparse and stable markowitz portfolios," *Proceedings of the National Academy of Sciences*, vol. 106, no. 30, pp. 12267–12272, 2009.
[22] A. Takeda, M. Niranjan, J.-Y. Gotoh, and Y. Kawahara, "Simultaneous pursuit of out-of-sample performance and sparsity in index tracking portfolios," *Computational Management Science*, vol. 10, no. 1, pp. 21–49, 2013.
[23] K. Andriosopoulos, M. Doumpos, N. C. Papapostolou, and P. K. Pouliasis, "Portfolio optimization and index tracking for the shipping stock and freight markets using evolutionary algorithms," *Transportation Research Part E: Logistics and Transportation Review*, vol. 52, pp. 16–34, 2013.
[24] T.-J. Chang, N. Meade, J. E. Beasley, and Y. M. Sharaiha, "Heuristics for cardinality constrained portfolio optimisation," *Computers & Operations Research*, vol. 27, no. 13, pp. 1271–1302, 2000.
[25] R. Ruiz-Torrubiano and A. Suárez, "A hybrid optimization approach to index tracking," *Annals of Operations Research*, vol. 166, no. 1, pp. 57–71, 2009.
[26] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, pp. 877–905, Dec. 2008.
[27] J. Song, P. Babu, and D. P. Palomar, "Sparse generalized eigenvalue problem via smooth optimization," *IEEE Transactions on Signal Processing*, vol. 63, pp. 1627–1642, Apr. 2015.
[28] K. Benidis, Y. Sun, P. Babu, and D. P. Palomar, "Orthogonal sparse PCA and covariance estimation via Procrustes reformulation," *IEEE Transactions on Signal Processing*, vol. 64, pp. 6211–6226, Dec. 2016.
[29] D. R. Hunter and K. Lange, "A tutorial on MM algorithms," *The American Statistician*, vol. 58, pp. 30–37, Feb. 2004.
[30] Y. Sun, P. Babu, and D. P. Palomar, "Majorization-minimization algorithms in signal processing, communications, and machine learning," *IEEE Transactions on Signal Processing*, vol. 65, pp. 794–816, Feb. 2016.
[31] D. P. Palomar and J. R. Fonollosa, "Practical algorithms for a family of waterfilling solutions," *IEEE Transactions on Signal Processing*, vol. 53, no. 2, pp. 686–695, 2005.
[32] J. Song, P. Babu, and D. P. Palomar, "Optimization methods for designing sequences with low autocorrelation sidelobes," *IEEE Transactions on Signal Processing*, vol. 63, pp. 3998–4009, Aug. 2015.
[33] Y. E. Nesterov and M. J. Todd, "Self-scaled barriers and interior-point methods for convex programming," *Mathematics of Operations research*, vol. 22, no. 1, pp. 1–42, 1997.
[34] M. A. Figueiredo, J. M. Bioucas-Dias, and R. D. Nowak, "Majorization–minimization algorithms for wavelet-based image restoration," *IEEE Transactions on Image Processing*, vol. 16, pp. 2980–2991, Dec. 2007.
[35] A. A. Gaivoronski and G. Pflug, "Value-at-risk in portfolio optimization: properties and computational approach," *Journal of Risk*, vol. 7, no. 2, pp. 1–31, 2005.
[36] R. T. Rockafellar and S. Uryasev, "Optimization of conditional value-at-risk," *Journal of Risk*, vol. 2, pp. 21–42, 2000.
[37] Z. M. Landsman and E. A. Valdez, "Tail conditional expectations for elliptical distributions," *North American Actuarial Journal*, vol. 7, no. 4, pp. 55–71, 2003.
[38] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath, "Coherent measures of risk," *Mathematical Finance*, vol. 9, no. 3, pp. 203–228, 1999.
[39] M. Wang, C. Xu, F. Xu, and H. Xue, "A mixed 0–1 LP for index tracking problem with CVaR risk constraints," *Annals of Operations Research*, vol. 196, no. 1, pp. 591–609, 2012.
[40] R. Varadhan and C. Roland, "Simple and globally convergent methods for accelerating the convergence of any EM algorithm," *Scandinavian Journal of Statistics*, vol. 35, no. 2, pp. 335–353, 2008.

Konstantinos Benidis received the M.Eng. degree from the School of Electrical and Computer Engineering, National Technical University of Athens (NTUA), Greece, in 2011, and the M.Sc. degree on Information and Communication Technologies from the Polytechnic University of Catalonia (UPC), Barcelona, Spain, in 2013.

He is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology (HKUST). His research interests are in convex optimization and efficient algorithms, with applications in signal processing, financial engineering, and machine learning.

Yiyong Feng received a B.E. degree in Electronic and Information Engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2010, and a Ph.D. degree in the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology (HKUST) under the supervision of Prof. Daniel P. Palomar in August 2015. He was also with the systematic market-making group at Credit Suisse (Hong Kong) from March 2013 to August 2013 and From September 2015 to March 2016. From April 2016 to May 2016, he was a research associate with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology. Since June 2016, he has been a research analyst with Three Stones Capital Limited, Hong Kong.

His research interests are in convex optimization, nonlinear programming, and robust optimization, with applications in signal processing, financial engineering, and machine learning.

Daniel P. Palomar (S'99-M'03-SM'08-F'12) received the Electrical Engineering and Ph.D. degrees from the Technical University of Catalonia (UPC), Barcelona, Spain, in 1998 and 2003, respectively.

He is a Professor in the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology (HKUST), Hong Kong, which he joined in 2006. Since 2013 he is a Fellow of the Institute for Advance Study (IAS) at HKUST. He had previously held several research appointments, namely, at King's College London (KCL), London, UK; Stanford University, Stanford, CA; Telecommunications Technological Center of Catalonia (CTTC), Barcelona, Spain; Royal Institute of Technology (KTH), Stockholm, Sweden; University of Rome La Sapienza, Rome, Italy; and Princeton University, Princeton, NJ. His current research interests include applications of convex optimization theory, game theory, and variational inequality theory to financial systems, big data systems, and communication systems.

Dr. Palomar is an IEEE Fellow, a recipient of a 2004/06 Fulbright Research Fellowship, the 2004 and 2015 (co-author) Young Author Best Paper Awards by the IEEE Signal Processing Society, the 2015-16 HKUST Excellence Research Award, the 2002/03 best Ph.D. prize in Information Technologies and Communications by the Technical University of Catalonia (UPC), the 2002/03 Rosina Ribalta first prize for the Best Doctoral Thesis in Information Technologies and Communications by the Epson Foundation, and the 2004 prize for the best Doctoral Thesis in Advanced Mobile Communications by the Vodafone Foundation and COIT.

He has been a Guest Editor of the IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING 2016 Special Issue on "Financial Signal Processing and Machine Learning for Electronic Trading", an Associate Editor of IEEE TRANSACTIONS ON INFORMATION THEORY and of IEEE TRANSACTIONS ON SIGNAL PROCESSING, a Guest Editor of the IEEE SIGNAL PROCESSING MAGAZINE 2010 Special Issue on Convex Optimization for Signal Processing, the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS 2008 Special Issue on "Game Theory in Communication Systems," and the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS 2007 Special Issue on "Optimization of MIMO Transceivers for Realistic Communication Networks."