Design of Portfolio of Stocks to Track an Index

Konstantinos Benidis and Daniel P. Palomar

2018-05-16

Contents

This vignette illustrates the design of sparse portfolios that aim to track a financial index with the package `sparseIndexTracking` (with a comparison with other packages) and gives a description of the algorithms used.

1 Comparison with other packages

There are currently no other R packages for index tracking. In paper [1] and monograph [2], a detailed comparison in terms of execution speed and performance is made with the mixed-integer quadratic programming (MIQP) solver Gurobi (for which R has an interface via package `ROI` and plugin `ROI.plugin.gurobi`).

2 Usage of the package

We start by loading the package and real data of the index S&P 500 and its underlying assets:

```
library(sparseIndexTracking)
library(xts)
data(data_2010_2011)
```

The data `data_2010_2011` contains a list with two xts objects:

1. `X`: A $T \times N$ xts with the daily linear returns of the N assets that were in the index during the period 2010-2011 (total T trading days)
2. `SP500`: A $T \times 1$ xts with the daily linear returns of the index S&P 500 during the same period.

Note that we use xts objects just for illustration purposes. The function `spIndexTrack()` can also be invoked passing simple data arrays or dataframes.

Based on the above quantities we create a training window, which we will use to create our portfolios, and a testing window, which will be used to assess the performance of the designed portfolios. For simplicity, here we consider the first six (trading) months of the dataset (~126 days) as the training window, and the subsequent six months as the testing window:

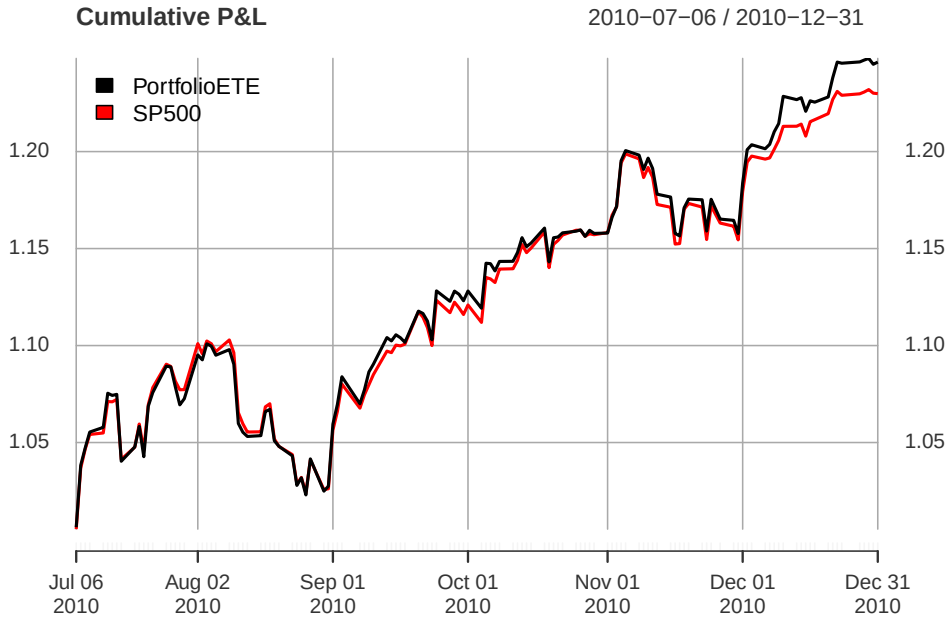
```
X_train <- data_2010_2011$X[1:126]
X_test <- data_2010_2011$X[127:252]
r_train <- data_2010_2011$SP500[1:126]
r_test <- data_2010_2011$SP500[127:252]
```

Now, we use the four modes (four available tracking errors) of the `spIndexTrack()` algorithm to design our portfolios:

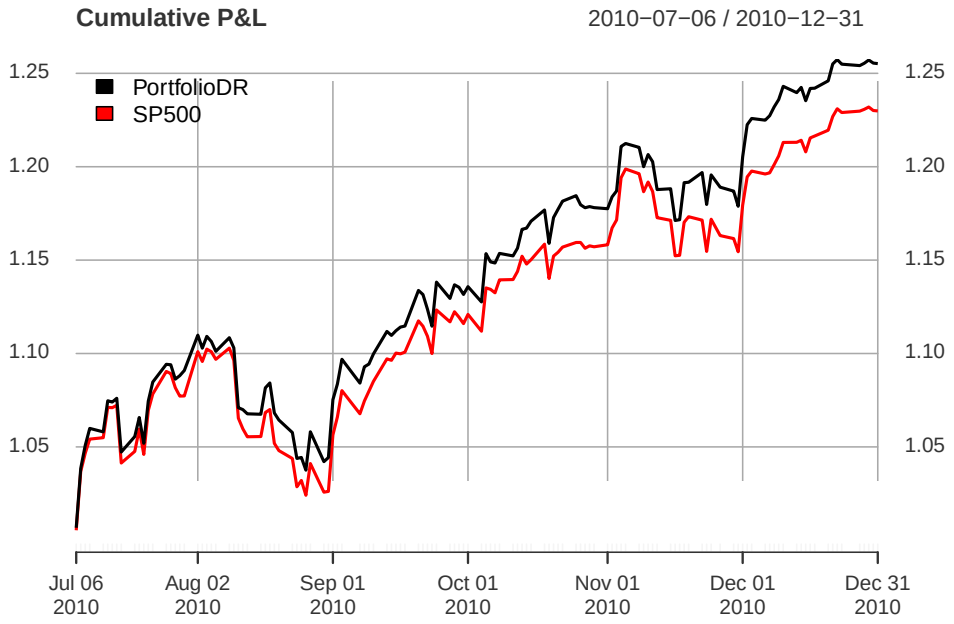
```
# ETE
w_ete <- spIndexTrack(X_train, r_train, lambda = 1e-7, u = 0.5, measure = 'ete')
cat('Number of assets used:', sum(w_ete > 1e-6))
#> Number of assets used: 45

# DR
w_dr <- spIndexTrack(X_train, r_train, lambda = 2e-8, u = 0.5, measure = 'dr')
cat('Number of assets used:', sum(w_dr > 1e-6))
#> Number of assets used: 42

# HETE
w_hete <- spIndexTrack(X_train, r_train, lambda = 8e-8, u = 0.5, measure = 'hete', hub = 0.05)
cat('Number of assets used:', sum(w_hete > 1e-6))
#> Number of assets used: 44

# HDR
w_hdr <- spIndexTrack(X_train, r_train, lambda = 2e-8, u = 0.5, measure = 'hdr', hub = 0.05)
cat('Number of assets used:', sum(w_hdr > 1e-6))
#> Number of assets used: 43
```

Finally, we plot the actual value of the index in the testing window in comparison with the values of the designed portfolios:

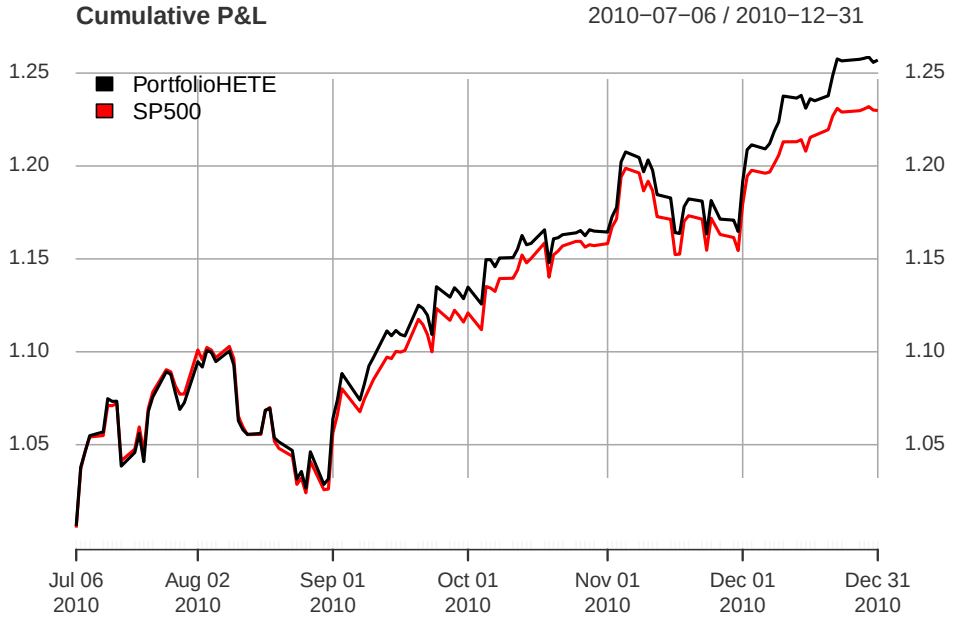
```
plot(cbind("PortfolioETE" = cumprod(1 + X_test %*% w_ete$w), cumprod(1 + r_test)),
     legend.loc = "topleft", main = "Cumulative P&L")
```



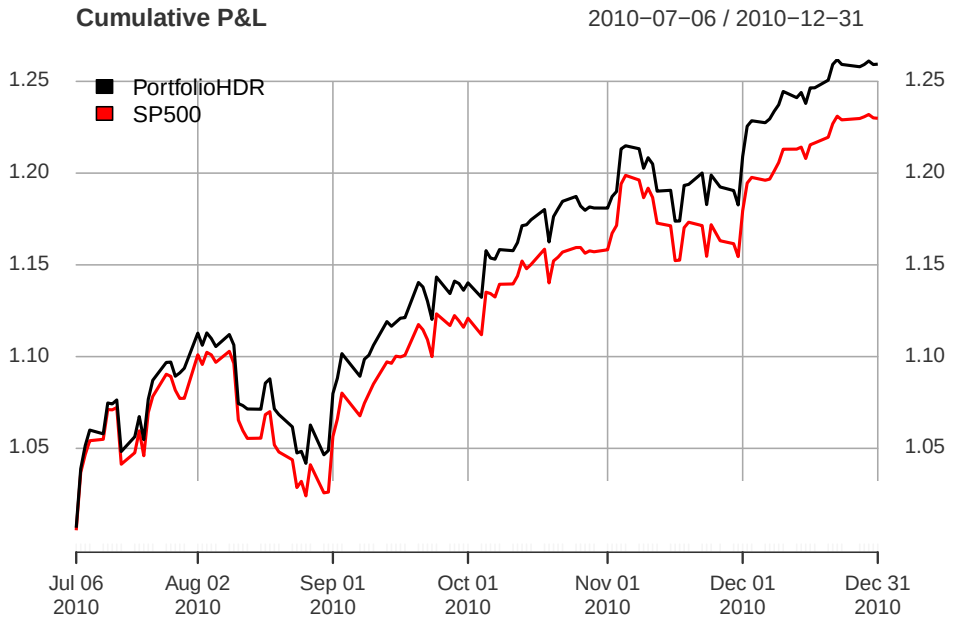

```
plot(cbind("PortfolioDR" = cumprod(1 + X_test %*% w_dr$w), cumprod(1 + r_test)),
     legend.loc = "topleft", main = "Cumulative P&L")
```




```
plot(cbind("PortfolioHETE" = cumprod(1 + X_test %*% w_hete$w), cumprod(1 + r_test)),
     legend.loc = "topleft", main = "Cumulative P&L")
```




```
plot(cbind("PortfolioHDR" = cumprod(1 + X_test %*% w_hdr$w), cumprod(1 + r_test)),
     legend.loc = "topleft", main = "Cumulative P&L")
```



In the above examples we used a single training and testing window. In practice, we need to perform this task sequentially for many windows in order to assess an algorithm or to distinguish the differences between the various tracking errors.

Ideally, the ETE and HETE portfolios should have Excess P&L close to zero since their purpose is to track as closely as possible the index, whereas the DR and HDR portfolios should have a positive Excess P&L since their purpose is to beat the index. Finally, Huber should shine in periods of high volatility where many extreme returns are observed (like the great recession).

All of the above can be observed in Figures 1 - 3 where we applied the four modes of the algorithm in the index S&P 500 considering three different periods. All the constructed portfolios consist of 40 assets, the training and testing windows were set to 6 months and 1 month, respectively, while monthly returns were used. The upper plot of each period (Normalized P&L) shows the wealth of the index and the four portfolios, which are normalized to the index value each time they are rebalanced. The lower plot of each period (Excess P&L) shows the cumulative difference of the portfolios and the index due to normalization, i.e., it is equivalent to a second account that keeps track of our excess profits or losses.

For a more detailed discussion please refer to [1] and [2].

3 Explanation of the algorithms

3.1 `spIndexTrack()`: Sparse portfolio construction

Assume that an index is composed of N assets. We denote by $\mathbf{r}^b = [r_1^b, \ldots, r_T^b]^\top \in \mathbb{R}^T$ and $\mathbf{X} = [\mathbf{r}_1, \ldots, \mathbf{r}_T]^\top \in \mathbb{R}^{T \times N}$ the (arithmetic) net returns of the index and the N assets in the past T days, respectively, with $\mathbf{r}_t \in \mathbb{R}^N$ denoting the net returns of the N assets at the t-th day.

The goal of `spIndexTrack()` is the design of a (sparse) portfolio $\mathbf{w} \in \mathbb{R}_+^N$, with $\mathbf{w}^\top \mathbf{1} = 1$, that tracks closely the index, i.e., $\mathbf{X}\mathbf{w} \approx \mathbf{r}^b$, based on [1]. The underlying optimization problem that is solved is

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{minimize}} & \text{TE}(\mathbf{w}) + \lambda \|\mathbf{w}\|_0 \\ \text{subject to} & \mathbf{w}^\top \mathbf{1} = 1 \\ & \mathbf{0} \leq \mathbf{w} \leq u\mathbf{1}, \end{array} \tag{1}$$

where $\text{TE}(\mathbf{w})$ is a general tracking error (we will see specific tracking errors shortly), λ is a regularization parameter that controls the sparsity of the portfolio, and u is an upper bound on the weights of the portfolio.

The ℓ_0-"norm" is approximated by the continuous and differentiable (for $w \geq 0$) function

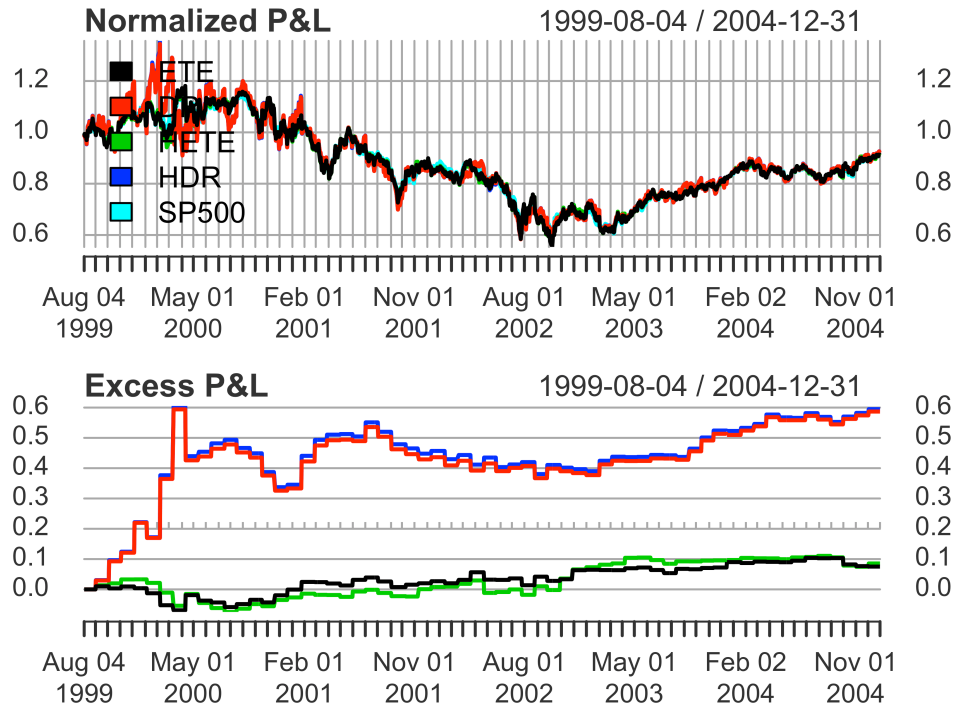


Figure 1: Dot-com bubble.

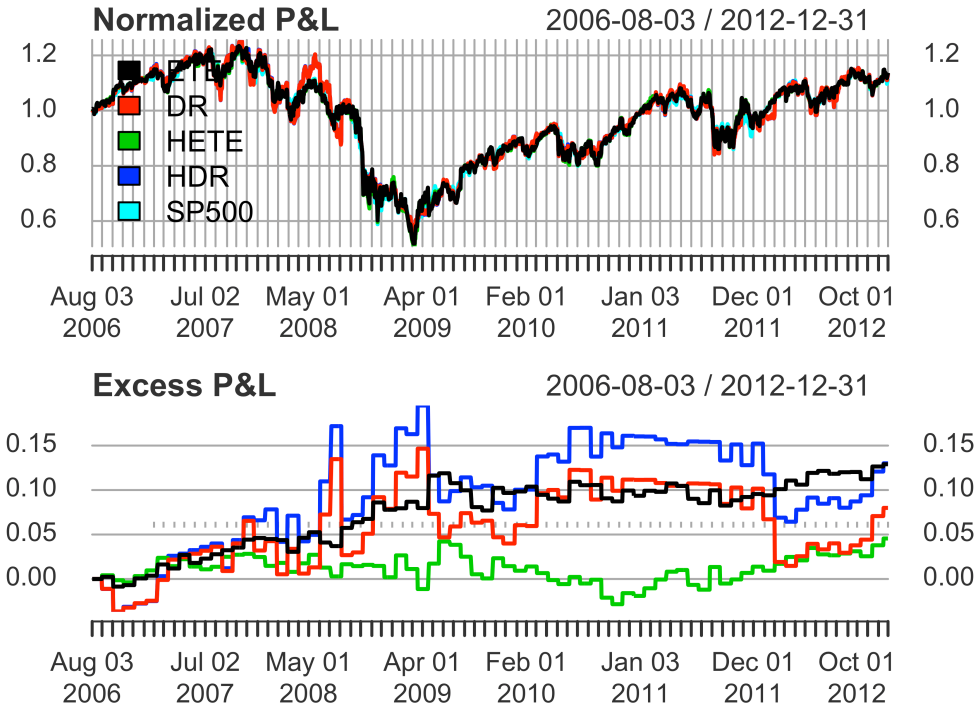


Figure 2: Great recession.

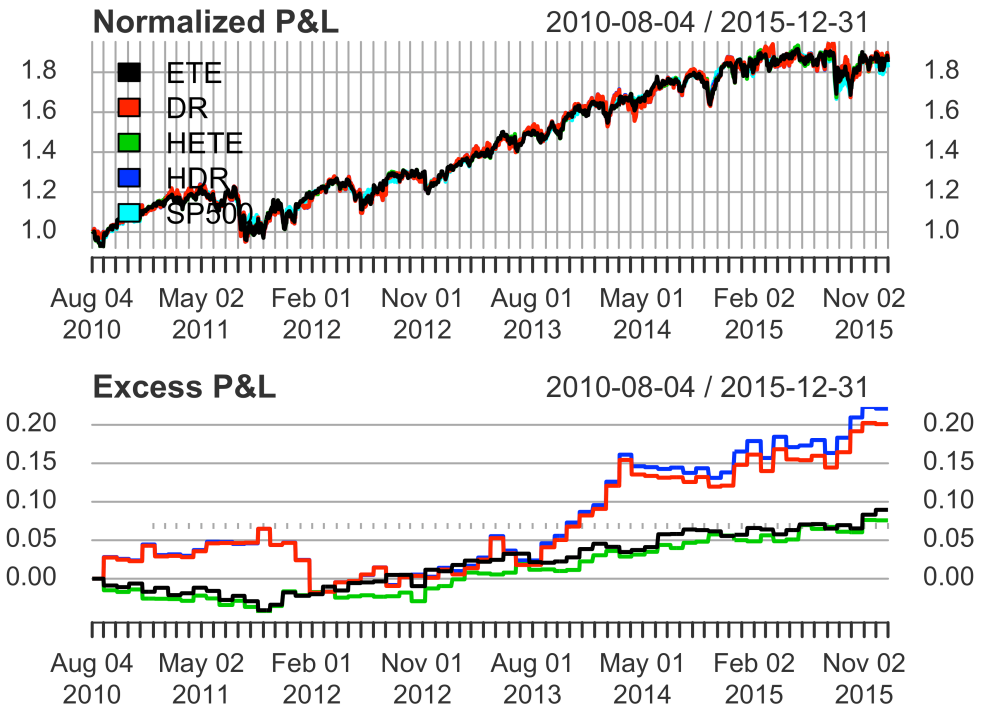


Figure 3: Stable market.

$$\rho_{p,u}(w) = \frac{\log(1 + w/p)}{\log(1 + u/p)}, \tag{2}$$

where $p > 0$ is a parameter that controls the approximation. This leads to the following approximate problem:

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{minimize}} & \text{TE}(\mathbf{w}) + \lambda \mathbf{1}^\top \boldsymbol{\rho}_{p,u}(\mathbf{w}) \\ \text{subject to} & \mathbf{w}^\top \mathbf{1} = 1 \\ & \mathbf{0} \leq \mathbf{w} \leq u\mathbf{1}, \end{array} \tag{3}$$

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

There are four available tracking errors $\text{TE}(\mathbf{w})$ in `spIndexTrack()`:

- Empirical tracking error (ETE):

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

- Downside risk (DR):

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

- Huber empirical tracking error (HETE):

$$\text{HETE}(\mathbf{w}) = \frac{1}{T}\mathbf{1}^\top \boldsymbol{\phi}\left(\mathbf{r}^b - \mathbf{X}\mathbf{w}\right)$$

- Huber downside risk (HDR):

$$\mathrm{HDR}(\mathbf{w}) = \frac{1}{T}\mathbf{1}^\top \boldsymbol{\phi}\left((\mathbf{r}^b - \mathbf{X}\mathbf{w})^+\right)$$

where $\boldsymbol{\phi}(\mathbf{x}) = [\phi(x_1), \ldots, \phi(x_T)]^\top$ and

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

with $M > 0$ being the Huber parameter.

Regardless of the selected tracking error measure, problem (3) can be solved via Majorization-Minimization (MM) [3] with an iterative closed-form update algorithm (with iterations denoted by k). It can be shown that all of the above variations boil down to the iterative optimization of the following convex problem:

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{minimize}} & \mathbf{w}^\top\mathbf{w} + \mathbf{q}^{(k)\top}\mathbf{w} \\ \text{subject to} & \mathbf{w} \in \mathcal{W}_u, \end{array} \tag{4}$$

where

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

and $\mathbf{q}^{(k)} \in \mathbb{R}^N$.

What differentiates the various tracking errors is the exact form of $\mathbf{q}^{(k)}$ that we need to compute at each iteration k of the algorithm:

$$\begin{aligned}
\mathbf{q}^{(k)}_{\mathrm{ETE}} &= \frac{1}{\lambda^{(\mathbf{L}_1)}_{\max}}(2(\mathbf{L}_1 - \lambda^{(\mathbf{L}_1)}_{\max}\mathbf{I})\mathbf{w}^{(k)} + \lambda\mathbf{d}^{(k)}_{p,u} - \frac{2}{T}\mathbf{X}^\top\mathbf{r}^b), \\
\mathbf{q}^{(k)}_{\mathrm{DR}} &= \frac{1}{\lambda^{(\mathbf{L}_1)}_{\max}}(\frac{2}{T}2(\mathbf{L}_1 - \lambda^{(\mathbf{L}_1)}_{\max}\mathbf{I})\mathbf{w}^{(k)} + \lambda\mathbf{d}^{(k)}_{p,u} + \frac{2}{T}\mathbf{X}^\top(\mathbf{y}^{(k)} - \mathbf{r}^b)), \\
\mathbf{q}^{(k)}_{\mathrm{HETE}} &= \frac{1}{\lambda^{(\mathbf{L}_2)}_{\max}}(2(\mathbf{L}_2 - \lambda^{(\mathbf{L}_2)}_{\max}\mathbf{I})\mathbf{w}^{(k)} + \lambda\mathbf{d}^{(k)}_{p,u} - \frac{2}{T}\mathbf{X}^\top\mathrm{Diag}(\mathbf{a}^{(k)})\mathbf{r}^b), \\
\mathbf{q}^{(k)}_{\mathrm{HDR}} &= \frac{1}{\lambda^{(\mathbf{L}_3)}_{\max}}(2(\mathbf{L}_3 - \lambda^{(\mathbf{L}_3)}_{\max}\mathbf{I})\mathbf{w}^{(k)} + \lambda\mathbf{d}^{(k)}_{p,u} + \frac{2}{T}\mathbf{X}^\top\mathrm{Diag}(\mathbf{b}^{(k)})(\mathbf{c}^{(k)} - \mathbf{r}^b)),
\end{aligned}$$

where $\lambda^{(\mathbf{A})}_{\max}$ denotes the maximum eigenvalue of a matrix $\mathbf{A}$, $\mathbf{I}$ denotes the identity matrix, $\mathrm{Diag}(\mathbf{x})$ is a diagonal matrix with the vector $\mathbf{x}$ at its principal diagonal, and

$$\mathbf{d}^{(k)}_{p,u} = \left[d_{p,u}(w^{(k)}_1), \ldots, d_{p,u}(w^{(k)}_N)\right]^\top, \tag{5}$$

$$d_{p,u}(w^{(k)}) = \frac{1}{\log(1 + u/p)(p + w^{(k)})}, \tag{6}$$

$$\mathbf{y}^{(k)} = -(\mathbf{X}\mathbf{w}^{(k)} - \mathbf{r}^b)^+, \tag{7}$$

$$\mathbf{a}^{(k)} = [a([\mathbf{r}^b - \mathbf{X}\mathbf{w}^{(k)}]_1), \ldots, a([\mathbf{r}^b - \mathbf{X}\mathbf{w}^{(k)}]_T)]^\top, \tag{8}$$

$$a(x) = \begin{cases} 1 & |x| \leq M \\ \frac{M}{|x|} & |x| > M, \end{cases} \tag{9}$$

$$\mathbf{b}^{(k)} = [b([\mathbf{r}^b - \mathbf{X}\mathbf{w}^{(k)}]_1), \ldots, b([\mathbf{r}^b - \mathbf{X}\mathbf{w}^{(k)}]_T)]^\top, \tag{10}$$

$$b(x) = \begin{cases} \frac{M}{M - 2x} & x < 0 \\ 1 & 0 \leq x \leq M \\ \frac{M}{x} & x > M, \end{cases} \tag{11}$$

$$\mathbf{c}^{(k)} = [c([\mathbf{r}^b - \mathbf{X}\mathbf{w}^{(k)}]_1), \ldots, c([\mathbf{r}^b - \mathbf{X}\mathbf{w}^{(k)}]_T)]^\top, \tag{12}$$

$$c(x) = \begin{cases} x & x < 0 \\ 0 & x \geq 0, \end{cases} \tag{13}$$

$$\mathbf{L}_1 = \frac{1}{T}\mathbf{X}^\top \mathbf{X}, \tag{14}$$

$$\mathbf{L}_2 = \frac{1}{T}\mathbf{X}^\top \mathrm{Diag}(\mathbf{a}^{(k)})\mathbf{X}, \tag{15}$$

$$\mathbf{L}_3 = \frac{1}{T}\mathbf{X}^\top \mathrm{Diag}(\mathbf{b}^{(k)})\mathbf{X}. \tag{16}$$

The following propositions provide a waterfilling structured solution of problem (4), considering two special cases, namely, $u = 1$ and $u < 1$.

Proposition 3.1 (AS_1)**.** The optimal solution of the optimization problem (4) with $u = 1$ is

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

with

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

and

$$\mathcal{A} = \big\{j \big| \mu + q_j < 0\big\},$$

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

Proposition 3.2 (AS_u)**.** The optimal solution of the optimization problem (4) with $u < 1$ is

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

with

$$\mu = -\frac{\sum_{j\in\mathcal{B}_2} q_j + 2 - \mathrm{card}(\mathcal{B}_1)2u}{\mathrm{card}(\mathcal{B}_2)},$$

and

$$\mathcal{B}_1 = \big\{j \big| \mu + q_j \leq -2u\big\},$$
$$\mathcal{B}_2 = \big\{j \big| -2u < \mu + q_j < 0\big\},$$

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

We refer to the iterative procedure of Proposition 3.1 as $\mathrm{AS}_1(\mathbf{q})$ (Active-Set for $u = 1$) and of Proposition 3.2 as $\mathrm{AS}_u(\mathbf{q})$ (Active-Set for general $u < 1$). The iterative closed-form update algorithm is given in Algorithm 1 (where $\mathrm{AS}_{1|u}(\mathbf{q})$ means $\mathrm{AS}_1(\mathbf{q})$ or $\mathrm{AS}_u(\mathbf{q})$).

Algorithm 1
1. Set $k = 0$ and choose an initial point $\mathbf{w}^{(0)}$ (by default set to $\mathbf{w}^{(0)} = \frac{1}{N}\mathbf{1}$)
2. Compute $\mathbf{q}$ according to the selected tracking error
3. Find the optimal solution $\mathbf{w}^\star$ with $\mathrm{AS}_{1|u}(\mathbf{q})$ and set it equal to $\mathbf{w}^{(k+1)}$
4. $k \leftarrow k + 1$
5. Repeat steps 2-4 until convergence
6. Return $\mathbf{w}^{(k)}$

Finally, note that the approximate problem is controlled by the parameter p, and in particular, as $p \to 0$ we get $\rho_{p,u} \to \ell_0$. However, by setting small values to p, it is likely that the algorithm will get stuck to a local minimum. To solve this issue we start with large values for 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, "tightest" problem. For each problem that is solved (i.e., for fixed p) we utilize an acceleration scheme that increases the convergence speed of the MM algorithm. For details, please refer to [4].

References

[1] K. Benidis, Y. Feng, and D. P. Palomar, "Sparse portfolios for high-dimensional financial index tracking," *IEEE Transactions on Signal Processing*, vol. 66, no. 1, pp. 155–170, Jan. 2018.

[2] K. Benidis, Y. Feng, and D. P. Palomar, *Optimization methods for financial index tracking: From theory to practice*. Foundations and Trends in Optimization, Now Publishers, 2018.

[3] 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, no. 3, pp. 794–816, Feb. 2017.

[4] 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.