Inverse Covariance Estimation with Structured Groups

Shaozhe Tao
University of Minnesota
taoxx120@umn.edu

Yifan Sun
Technicolor Research
yifan.sun@technicolor.com

Daniel Boley
University of Minnesota
boley@cs.umn.edu

Abstract

Estimating the inverse covariance matrix of $p$ variables from $n$ observations is challenging when $n \ll p$, since the sample covariance matrix is singular and cannot be inverted. A popular solution is to optimize for the $\ell_1$ penalized estimator; however, this does not incorporate structure domain knowledge and can be expensive to optimize. We consider finding inverse covariance matrices with group structure, defined as potentially overlapping principal submatrices, determined from domain knowledge (e.g. categories or graph cliques). We propose a new estimator for this problem setting that can be derived efficiently via the Frank-Wolfe method, leveraging chordal decomposition theory for scalability. Simulation results show significant improvement in sample complexity when the correct group structure is known. We also apply these estimators to 14,910 stock closing prices, with noticeable improvement when group sparsity is exploited.

1 Introduction

The inverse covariance matrix is of interest to statisticians in biology, finance, machine learning, etc. In finance, it is a key ingredient for computing value-at-risk, a factor in portfolio optimization. In graphical models, for $p$ random variables with true covariance matrix $C$, the sparsity pattern of $C^{-1}$ gives the conditional independence between each pair of variables. However, if $n \ll p$, then the sample covariance matrix $\hat{C}$ is not invertible, and the pseudoinverse $\hat{C}^+$ is inaccurately dense. The most popular alternative is the graphical LASSO (G-LASSO) estimator [Yuan and Lin, 2007; Banerjee et al., 2008], the solution to

$$\begin{align*}
\text{minimize} & \quad -\log\det(X) + \text{tr}(\hat{C}X) + \rho\|X\|_1 \\
\text{subject to} & \quad X \succeq 0
\end{align*}$$

(1)

for some regularization parameter $\rho > 0$. By adding a sparsity-inducing regularizer, the effective degrees of freedom are reduced, and it has been shown that the resulting estimator has a much lower sample complexity than inverting $\hat{C}$. However, this estimator does not incorporate any prior structural knowledge from the problem domain. Additionally, in general solving (1) is computationally challenging if $p$ is large.

Most existing methods for solving (1) require a sequence of eigenvalue decompositions (EDs) [Banerjee et al., 2006; Friedman et al., 2008; d’Aspremont et al., 2008; Yuan, 2012; Rolfis et al., 2012]. This is expensive if $p$ is large; a dense ED requires $O(p^3)$ computations, and sparse EDs (like Lanczos type methods) are not suitable when the full eigenvalue spectrum is needed. (Used this way, they may be far slower than dense methods.) There are some exceptions; for example [Scheinberg and Rish, 2009] updates each row in a block coordinate descent fashion, and maintains inverses using only rank-2 updates; [Dahl et al., 2008] uses chordal decomposition to compute Newton steps efficiently in an interior point solver; and [Meinshausen and Bühlmann, 2006] uses neighborhood selection, which enforces the conditional independence condition one variable at a time. These methods are more or less intuitive, relying on general convex optimization principles; however, their scalability is limited. On the other end of the spectrum is BIG-QUIC [Hsieh et al., 2013] which can solve up to 1 million variables. This breakthrough method simultaneously makes estimates of the matrix sparsity while also optimizing for it, and updating via block coordinate descent with carefully chosen (non-principal) submatrices. However, it demonstrates the tradeoff between simplicity and scalability; there are many intricate details for a successful implementation.

At the same time, there has been growing interest in the statistics community to exploit group structure in the estimators [Negahban et al., 2009; Chandrasekaran et al., 2012; Obozinski et al., 2011]. For example, [Mazumder and Hastie, 2012] proposes thresholding the sample covariance matrix in order to identify fully-connected components of the graphical model, effectively decomposing (1). More recently, [Hosseini and Lee, 2016] learns overlapping submatrix groups probabilistically and penalizes accordingly. To our knowledge, this is the only work that addresses overlapping group sparsity in matrices; however, repeated full EDs are still needed to find the inverse covariance estimate.

We propose an estimator that exploits group structure, where a matrix group is described as either a principal submatrix or the matrix diagonal in Section 2. The solution $X$ is then described as a sum of these possibly overlapping components. We then apply the Frank-Wolfe method to derive...
the estimator in Section 3. The algorithm at each iteration decomposes into parallelizable eigenvalue computations on the submatrices. In this way, unlike [Mazumder and Hastie, 2012; Hosseini and Lee, 2016], this estimator explicitly uses the predetermined groups as components for decomposition, thus using group structure to improve both performance and computation cost. In Section 4 we give simulation results, which demonstrate that knowing and exploiting group structure significantly improves sample complexity. Finally, in Section 5, we show the performance of our model on the stock datasets.

2 Group Norm Constrained Estimator

For an index set $\gamma \subset \{1, \ldots, p\}$ and a vector $x \in \mathbb{R}^p$, define $x_\gamma$ as the subvector of $x$ indexed by $\gamma$; for the reverse, define the augmenting linear map $\gamma : \mathbb{R}^{|\gamma|} \rightarrow \mathbb{R}^p$ such that

$$(A_\gamma u)_{\gamma} = u, \quad (A_\gamma u)_i = 0 \text{ if } i \notin \gamma.$$ 

In [Obozinski et al., 2011], the overlapping group norm is defined as the solution to

$$\|x\|_G = \min_{u_1, \ldots, u_l} \left\{ \sum_{k=1}^l w_k \|u_k\| : \text{s.t. } x = \sum_{k=1}^l A_\gamma u_k \right\} \quad (2)$$

for some proper norm $\|\cdot\|$ and nonnegative weights $w_1, \ldots, w_l$. A common choice is $w_k = |\gamma_k|^{-1}$. Used as a penalty term or in a constraint, this norm is shown to promote group structure; a small subset of index sets $\gamma_k$ are “active”, and $x_i = 0$ whenever $i$ is not in an active set.

We extend this concept to matrices, by defining groups implicitly through index sets $\beta \subset \{1, \ldots, p\}$, where $X_{\beta, \beta}$ is the submatrix of $X$ selected by the rows and columns indicated by $\beta$. Let $\mathbb{S}^p$ denote the set of $p \times p$ matrices. We define $A_{\beta} : \mathbb{S}^{|\beta|} \rightarrow \mathbb{S}^p$ such that

$$(A_{\beta}(U))_{\beta, \beta} = U, \quad A_{\beta}(U)_{i,j} = 0 \text{ if } i \notin \beta \text{ or } j \notin \beta$$

and extend the overlapping group norm as the solution

$$\|X\|_G := \min_{v \in \mathbb{S}^p, U_1, \ldots, U_l} \left\{ \sum_{k=1}^l w_k \|U_k\|_F : \text{s.t. } X = \text{diag}(v) + \sum_{k=1}^l A_{\beta_k}(U_k) \right\} \quad (3)$$

Here, $X$ can be considered as the sum of smaller principal submatrices $U_k$ and a diagonal term $v$, and $w_k$ are nonnegative scalar weights. Note that the affine constraint imposes a sparsity pattern on $X$; if $X$ does not adhere to this pattern (e.g., $X_{ij} \neq 0$ for $i, j \notin \beta_k$, $\forall k$) then we define $\|X\|_G = \infty$.

2.1 Our estimator

Given $p$ random variables, define $C \in \mathbb{S}^p$ and $\hat{C} \in \mathbb{S}^p$ as the true and sample covariance matrices. The group norm regularized graphical LASSO estimator (NG-LASSO) is the solution to

$$\min_X -\log\det(X) + \text{tr}(\hat{C}X)$$

s.t. $X = \sum_{k=1}^l A_{\beta_k}(U_k) + \text{diag}(v)$

$$w_0 \|v\|_2 + \sum_{k=1}^l w_k \|U_k\|_F \leq \alpha$$

$$U_k \succeq 0, \quad k = 1, \ldots, l$$

$$v_i \geq 0, \quad i = 1, \ldots, p.$$ 

We note that the first two constraints in (4) can be equivalently written as $\|X\|_G \leq \alpha$ with $G$-norm defined in (3). As defined, this constraint restricts $X$ to be implicitly within the sparsity pattern defined by the groups $\beta_k$.

Problem (4) is a computationally tractable approximation of

$$\min_X -\log\det(X) + \text{tr}(\hat{C}X)$$

subject to $X \succeq 0, \quad \|X\|_G \leq \alpha$ \quad (5)

a natural group norm penalized version of the G-LASSO problem. Specifically, in (5), $\|X\|_G$ can be written in terms of smaller matrices $W_k \in \mathbb{S}^{|\beta_k|}$ and $z \in \mathbb{R}^p$. If additionally the sparsity pattern is chordal (i.e. if the intersection graph of the groups $\beta_k$ is a tree) then the positive semidefinite (PSD) matrix constraint $X \succeq 0$ can be decomposed into several smaller matrix constraints, via the equivalence in the following theorem.

Theorem 2.1 [Agler et al., 1988; Griewank and Toint, 1984] ([Grone et al., 1984 dual]) If $X \in \mathbb{S}^p$ has chordal sparsity, corresponding to groups $\beta_1, \ldots, \beta_l$, then

$$X \succeq 0 \iff X = \sum_{k=1}^l A_{\beta_k}(U_k), \quad U_k \succeq 0, \quad k = 1, \ldots, l.$$ 

In this case $X \succeq 0$ can be decomposed into smaller matrices $U_k \in \mathbb{S}^{|\beta_k|}$ and a positive diagonal $v \in \mathbb{R}^p$ (where $\mathbb{S}^p_+ \text{ and } \mathbb{R}^p_+$ are the PSD cone and nonnegative orthant, both of order $p$). Then (4) is equivalent to (5) if and only if at optimality, $W_k = U_k$ for all $k$, and $v = z$.

3 Optimization

The Frank-Wolfe algorithm has regained much attention in minimizing sparse problems [Jaggi, 2013], mimicking greedy approaches yet having guaranteed optimality for convex problems. The Frank-Wolfe algorithm for solving $\min_X \{f(X) : X \in \mathcal{D}\}$ is described in Alg. 1. Using step sizes $\eta^{[t]} = 2/(t + 2)$, it is known that the iterates of algorithm 1 converge at the sublinear rate of $O(1/t)$ [Frank and Wolfe, 1956; Dunn and Harshbarger, 1978]. We apply this algorithm for problem (4).

Algorithm 1 One step of Frank-Wolfe algorithm

**Input:** $X^{[t]} \in \mathcal{D}$, $t$-th iteration, step size $\eta$

1. Compute gradient $\nabla f(X^{[t]})$

2. Compute forward step: $S = \arg \min_{S \in \mathcal{D}} \langle U, \nabla f(X^{[t]}) \rangle$

3. Update primal variable: $X^{[t+1]} = (1 - \eta^{[t]} \|S\|_2^2)X^{[t]} + \eta^{[t]} S$

**Output:** optimal $X^{[t+1]}$
**Forward step** At each iteration, the forward step consists of \( l \) parallelizable projections on cones \( C_1, \ldots, C_l \). Specifically, at each forward step, we compute
\[
U_j^* = \frac{\alpha}{w_j\|Z_j\|_F} Z_j, \quad U_k = 0, \; \forall k \neq j.
\]
where index \( j = \arg \max \alpha w_k u_k^{-1}\|Z_k\|_F \) and \( Z_j = \text{proj}_{S_j}(-\nabla f(X)\beta_j, \beta_j) \). Then \( S = \sum_k w_k A_{\beta_k, \beta_k}(U_k) \). The derivations are given in appendix A.

**Gradient computation** In general, to compute the gradient \( \nabla \log \det(X) \) requires matrix inversion, which negates the computational complexity gain by decomposing the PSD cone. However, if the groups \( \beta_k \) form a chordal pattern, fast inversion methods exist [Liu, 1992; Andersen et al., 2013] which require at each step \( l \) inversions of matrices at most of order \( |\beta_k| \).

Applying both techniques, Alg. 2 describes the procedure for one iteration to find the NG-LASSO estimator (4). The main computational bottleneck at each step is a sequence of EDs of the submatrices \( \nabla f(X)\beta_k, \beta_k \) both for inverting \( X \) and for projecting on the PSD cone. For both operations, the complexity is \( O(|\beta_k|^3) \) per group. If \( |\beta_k| < p/l \) excluding the diagonal group, then the total per-iteration complexity of the proposed optimization procedure has a per-iteration complexity of \( O(p^3/l^2 + p) \), and much smaller than \( O(p^3) \) for G-LASSO.

4 Numerical Simulations

Here we show two simulation results: one on a general group sparsity pattern, and the other on a specific chordal pattern (banded).

To pick \( \alpha \) and \( \rho \), we swept powers of two in \( \{2^{-5}, \ldots, 2^5\} \) and picked the best performing \( \rho \) or \( \alpha \) for each test. In cases where the best performing \( \rho \) or \( \alpha \) was on the boundary, additional parameters were tested until this was no longer the case.

4.1 Baselines

As one baseline, we solve (1). However, since group structure also reveals matrix sparsity, for fair comparison we also solve (1) restricted to the sparsity pattern induced by the groups:
\[
\begin{align*}
\min_X & \quad -\log \det(X) + \text{tr}(\hat{C}X) + \rho \|X\|_1 \\
\text{s.t.} & \quad X \succeq 0 \\
& \quad X \in B := \{X \mid X_{ij} = 0 \text{ if } i, j \notin \beta_k \text{ \forall } k\},
\end{align*}
\]
which we call restricted group LASSO (RG-LASSO). We solve these baselines using the Douglas-Rachford method [Lions and Mercier, 1979; Combettes and Pesquet, 2011] for minimizing the sum of \( m \) convex functions, \(^1\) detailed exactly as in [Combettes and Pesquet, 2011]. Alg 10.27, with
\[
f_1(X) = -\log \det(X) + \text{tr}(\hat{C}X), \quad f_2(X) = \rho \|X\|_1
\]
and \( f_3, f_4 \) as indicator functions for constraints
\[
f_3(X) = \begin{cases} 0 & X \succeq 0 \quad \text{else} \\ \infty & X \in B \end{cases}, \quad f_4(X) = \begin{cases} 0 & X \in B \\ \infty & \text{else} \end{cases}.
\]
The proximal operator [Moreau, 1962] for a convex function \( f(X) \) is defined as
\[
\text{prox}_f(Z) = \arg \min_X f(X) + (1/2)\|X - Z\|_F^2
\]
and is defined for all \( Z \), even if \( Z \) is not in the domain of \( f \). (This is especially useful for \( f = \log \det \) and \( Z \neq \infty \).) From optimality conditions, it can be shown that
\[
\text{prox}_{f_1}(Z) := \text{V diag}(q)V^T, \quad 2q_k = -d_k + \sqrt{d_k^2 + 4t}
\]
where \( \text{V diag}(d)V^T \) is the eigenvalue decomposition of \( t\hat{C} - Z \). Similarly, \( \text{prox}_{f_2} \) is the well-known shrinkage operator, and \( \text{prox}_{f_3}, \text{prox}_{f_4} \) are projections on their respective constraint spaces.

4.2 Random sparsity

For \( X \in \mathbb{S}^p \), we randomly select \( l \) groups \( \beta_k \subset \{1, \ldots, p\} \) of size \( b \), and assume that this is the known group structure. Additionally, we select \( \{\sigma_G, l\} \) “active” groups (for \( 0 < \sigma_G < 1 \); the identity of these groups are not known in training. In this simulation, we investigate the sample size required to recover the active groups, comparing G-LASSO, RG-LASSO, and NG-LASSO. In general, the sparsity patterns here are not chordal.

To determine the sparsity pattern of \( X \), i.e. the estimate of \( C^{-1} \), we threshold so that
\[
i, j \text{ is a nonzero index if } \frac{|X_{ij}|}{\max_k |X_{k}|} > \theta,
\]
for \( \theta \in [0, 1] \). Figure 1 shows the receiver operating characteristic (ROC) curve for \( p = 100 \) variables and \( n = 25 \) samples, where \( \theta \) is implicitly swept. It is clear that NG-LASSO outperforms both other estimators; however, it is also evident that restricting the sparsity already offers much improvement. In order to remain agnostic to the right choice of \( \theta \), we use the area under the ROC curve (AUC) as the primary metric of estimator quality. Figure 2 plots the AUC for varying sample sizes \( n \). Again, it is clear that for most cases, NG-LASSO outperforms RG-LASSO and G-LASSO. The exception is at very small values of \( n \), where the number of observations is too low.

\(^1\) This ADMM-type of method has the advantage over the projected gradient method because it has no step size restrictions based on the objective function’s Lipschitz constant, which for \( \log \det \) is unbounded.
4.3 Banded sparsity

Next, we present a more extensive experiment with a chordal sparsity pattern; specifically, a banded pattern. For $X \in \mathbb{R}^p$, we assume that the true sparsity pattern consists of a nonzero diagonal and some active diagonal blocks of size $b$, where $b$ is known but the true sparsity pattern is not. This gives in total $l = p - b + 1$ candidate groups $\beta_k = \{k, \ldots, k + b\}$ for $k = 1, \ldots, l$. Among $l$ groups, we assume $\sigma_G l$ groups are active (where $0 < \sigma_G < 1$). Moreover, we simulate in-group sparsity; that is, for $0 < \sigma_I \leq 1$, we fix $\Pr(X_{ij} \neq 0|i, j \in \beta_k) = \sigma_I$. Note that using only known information, we must assume the sparsity pattern is banded with bandwidth $b$.

We construct an invertible $C^{-1}$ with the true sparsity pattern, and form a sample covariance matrix $\hat{C}$ by sampling from a multivariate Gaussian with 0 mean and covariance $C$. The goal is to use the estimators to correctly recover the sparsity pattern of $C$ using $\hat{C}$ where the number of observations $n$ is as small as possible.

Figure 3 shows a small example when $p = 100$, $n = 50$ and $\sigma_I = 0.25$. There are in total 90 groups in the banded sparsity pattern, where the 9 active groups (true sparsity) are in blue. We pick the estimator nonzeros by thresholding on the absolute value, choosing the threshold to, in each case, minimize $\min \{\# \text{ true positives}, \# \text{ true negatives} \}$.

It is clear that, for this small example, G-LASSO (left) yields many spurious nonzeros. By simply restricting the sparsity pattern to $B$, the performance of RG-LASSO (center) already improves significantly, but NG-LASSO is still the best, since it accounts for sparsity in group selection as well.

Table 1 gives the result of a more extensive experiment, where the threshold, $\alpha$, and $p$ are picked to maximize the AUC, which is given for several $p$, $n$ and $\sigma_I$. Here, we see that NG-LASSO is comparable with RG-LASSO when $p \approx n$, but is consistently better for $p \gg n$; both, however, are considerably improved over G-LASSO.

Table 2 gives the per-iteration and total runtime of the three methods. In all cases, the per-iteration runtime depends only on $p$ and $b$, and for larger $p$, NG-LASSO enjoys a much smaller per-iteration runtime. Of course, the total runtime (influenced also by the number of iterations) is also important; however, our experiments suggest this value is much less predictable. Several trends do emerge, though; for all methods, the runtime grows significantly when $n$ is very small or $\sigma_I$ is very low, suggesting that problem conditioning influences convergence rate as well. However, the key takeaway is that for very large $p$ it is impossible to maintain full EDs for each iteration, and some decomposition must be used.

5 Financial application

In this section, we examine the performance of G-LASSO, RG-LASSO, and NG-LASSO in estimating the inverse covariance matrix for 14910 stocks, over a period of 100 or fewer days. This problem arises in finance in Markowitz portfolio optimization [Markowitz, 1952], which discusses optimal portfolio allocations using only mean and covariance information. Specifically, when the objective is to minimize return volatility, it is advised to invest in each stock a weight $w_i$ of one’s assets, where $w$ minimizes the following quadratic optimization
The sample covariance is then calculated as

$$\hat{C}_{ij} = \frac{1}{|R_i \cap R_j|} \sum_{k \in R_i \cap R_j} u_i[k]u_j[k].$$

We solve (1), (6), and (4) sweeping $\rho, \alpha$ for powers of 2 from $2^{-5}$ to $2^6$, using cross validation to pick the best $\rho$ and $\gamma$. 3

Unlike the controlled simulation, we have no real understanding of the ground truth, so we test the estimator’s generalizability, by measuring the maximum likelihood over the test samples. Specifically, we compute the negative log likelihood (NLL) for precision matrix $X$ and samples $\{u_i\}_{i \in T}$ in the test set:

$$\text{NLL} = -\log \det(X) + \frac{X_{ij}}{|T_i \cap T_j|} \sum_{k \in T_i \cap T_j} u_i[k]u_j[k].$$

2Not every day’s value is provided for every stock.

3As before, if the best $\rho$ or $\alpha$ is on the boundary, additional powers of 2 are computed until this is no longer the case.

<table>
<thead>
<tr>
<th>$p$</th>
<th>$n$</th>
<th>G</th>
<th>RG</th>
<th>NG</th>
<th>RG</th>
<th>NG</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>10</td>
<td>2.2e2</td>
<td>2.5e2</td>
<td>7.0e2</td>
<td>2.7e2</td>
<td>5.6e2</td>
</tr>
<tr>
<td>500</td>
<td>10</td>
<td>2.0e3</td>
<td>1.8e3</td>
<td>3.7e3</td>
<td>1.3e3</td>
<td>3.5e3</td>
</tr>
<tr>
<td>500</td>
<td>100</td>
<td>1.3e3</td>
<td>9.1e2</td>
<td>4.1e3</td>
<td>9.3e2</td>
<td>3.1e3</td>
</tr>
<tr>
<td>1000</td>
<td>10</td>
<td>2.5e3</td>
<td>2.6e3</td>
<td>7.9e3</td>
<td>2.8e3</td>
<td>6.1e3</td>
</tr>
<tr>
<td>1000</td>
<td>100</td>
<td>1.1e13</td>
<td>4.4e3</td>
<td>7.1e3</td>
<td>4.4e3</td>
<td>7.4e3</td>
</tr>
<tr>
<td>2500</td>
<td>100</td>
<td>-</td>
<td>1.1e4</td>
<td>3.9e4</td>
<td>1.1e4</td>
<td>1.7e4</td>
</tr>
<tr>
<td>5000</td>
<td>100</td>
<td>-</td>
<td>1.5e12</td>
<td>1.1e7</td>
<td>5.9e10</td>
<td>7.9e4</td>
</tr>
<tr>
<td>14910</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>8.6e12</td>
<td>-</td>
<td>5.2e5</td>
</tr>
</tbody>
</table>

Table 3: Best test negative log likelihood for different methods, varying the number of stocks ($p$) and observations ($n$). $G$ = G-LASSO. RG = RG-LASSO. NG = NG-LASSO. ‘-’ = runtime was too long.

In this case, the smaller the value, the better.

Table 3 gives the test NLL for various $p$ and $n$. As $p$ grows (and especially as $p/n$ grows) it is clear that G-LASSO has poorer performance, though the behavior is not as consistent as in table 1. Additionally, it seems that restricting the sparsity pattern gives the regularization needed to achieve very good performance, though we note that for very large $p$, NG-LASSO still does better.

Table 4 gives the runtime of each experiment for the best set of parameters, showing 1) the time to do one set of EDs ($p \times p$ for LASSO and sum of $|\beta_k| \times |\beta_k|, k = 1, \ldots l$ for group methods), 2) the average per-iteration runtime ($\approx 1$ ED) and 3) the total runtime. Since in this application all groups are nonoverlapping (all stocks are assigned a single sector and industry) computing RG-LASSO can be done in a fully decomposable manner, and can run at the same per-iteration speed as NG-LASSO. However, because the Douglas-Rachford method requires keeping 9 copies of the variables as separate iterates, it runs out of memory at $p = 14910$. Therefore our method still maintains the scalability advantage.

6 Conclusion

We present an inverse covariance estimator that regularizes for overlapping group sparsity, and provide better estimates when $p \gg n$. This is a challenging problem, common in finance, cli-
Table 4: Runtimes (in seconds) of various algorithms for different matrix sizes ($C$ is $p \times p$). S = sectors. I = industries. S-ED (I-ED) ≈ time to compute $p_i \times p_i$ ED where $p_1, \ldots, p_l$ are the sizes of the $l$ sector (industry) groups. G = G-LASSO. RG = RG-LASSO. NG = NG-LASSO. “*” = runtime was too long.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$p \times p$ ED</th>
<th>S-ED</th>
<th>I-ED</th>
</tr>
</thead>
<tbody>
<tr>
<td>G l it.</td>
<td>0.381</td>
<td>1.33</td>
<td>20.5</td>
</tr>
<tr>
<td>RG (S) l it.</td>
<td>&lt; 0.1</td>
<td>0.20</td>
<td>1.6</td>
</tr>
<tr>
<td>RG (I) l it.</td>
<td>&lt; 0.1</td>
<td>0.17</td>
<td>0.74</td>
</tr>
<tr>
<td>NG (S) l it.</td>
<td>&lt; 0.1</td>
<td>0.20</td>
<td>1.4</td>
</tr>
<tr>
<td>NG (I) l it.</td>
<td>&lt; 0.1</td>
<td>0.18</td>
<td>0.72</td>
</tr>
<tr>
<td>G all</td>
<td>20.0</td>
<td>1324.2</td>
<td>-</td>
</tr>
<tr>
<td>RG (S) all</td>
<td>3.0</td>
<td>88.9</td>
<td>713.6</td>
</tr>
<tr>
<td>RG (I) all</td>
<td>2.5</td>
<td>74.0</td>
<td>341.5</td>
</tr>
<tr>
<td>NG (S) all</td>
<td>2.5</td>
<td>29.8</td>
<td>169.1</td>
</tr>
<tr>
<td>NG (I) all</td>
<td>4.6</td>
<td>26.9</td>
<td>79.7</td>
</tr>
</tbody>
</table>

For notational convenience, define $c_k = \text{vec}(\nabla f(X) u_k)$ for $k = 1, \ldots, l$. Then we can rewrite the objective function in (9) as

$$\max_{u_k} \sum_k (-c_k)^T u_k$$

with constraints unchanged. From Moreau’s decomposition, any vector $a$ can be written as the sum of its projection on a closed convex cone $C$ and its polar cone $C^\circ$, of which are orthogonal. If we then expand

$$c_k^T u_k = \text{proj}_{C_k}(c_k)^T u_k + \text{proj}_{C_k^\circ}(c_k)^T u_k$$

then since feasible $u_k \in C_k$, by definition of polar cone $\text{proj}_{C_k^\circ}(c_k)^T u_k \leq 0$, and $a = 0$ only if $u_k = s_k \text{proj}_{C_k}(c_k)$. This is the optimal choice of direction for $u_k$, since it also maximizes the first term $\text{proj}_{C_k}(c_k)^T u_k$, and does not affect the norm constraint. If we additionally define scalars

$$a_k = \|P_{C_k}(-c_k)\|_2, \quad b_k = w_k \|P_{C_k}(-c_k)\|_2$$

then an even simpler equivalent formulation is

$$\max_{s_k} a_k^T s$$

subject to $b^T s \leq \alpha, \quad s \geq 0$

which is a linear program with a known optimal solution of

$$s_i = \begin{cases} \alpha/b_i & \text{if } i = \arg\max_i a_i/b_i \\ 0 & \text{else.} \end{cases}$$

Substituting gives the closed form solution in the text.

B  Yahoo! Finance data scraping details

Using the Yahoo! ticker downloader \(^4\) we downloaded 27684 tickers for different stocks. We then used the Yahoo! finance API \(^5\) to gather daily open, high, low, close, volume, and adjusted closing prices. We chose to monitor daily closing prices. We define groups as industries or sectors, as described in

http://biz.yahoo.com/p/.

Any stock that we could not identify with an industry and sector was removed. We then prune the data to make sure it is as dense as possible. This resulted in 14,910 stocks and 1,005 daily closing prices. In total there are 9 sectors and 214 industries, with an average sector size 1656.7 and industry size 69.3. All stock vectors were then demeaned.

References


---

\(^4\)https://pypi.python.org/pypi/Yahoo-ticker-downloader

\(^5\)http://chart.finance.yahoo.com/table.csv?symbol=TICKERNAMEHERE\&a=400\&b=23\&c=2016\&d=0\&e=23\&f=2017\&g=d&ignore=.csv