# Efficient $k$-Support Matrix Pursuit

Hanjiang Lai[1,3], Yan Pan[2,⋆], Canyi Lu[1], Yong Tang[4], and Shuicheng Yan[1]

[1] Department of Electrical and Computer Engineering, National University of Singapore
[2] School of Software, Sun Yat-sen University, China
[3] School of Information Science and Technology, Sun Yat-sen University, China
[4] School of Computer Science, South China Normal University, China
{laihanj,canyilu}@gmail.com, panyan5@mail.sysu.edu.cn,
ytang@scnu.edu.cn, eleyans@nus.edu.sg

**Abstract.** In this paper, we study the $k$-support norm regularized matrix pursuit problem, which is regarded as the core formulation for several popular computer vision tasks. The $k$-support matrix norm, a convex relaxation of the matrix sparsity combined with the $\ell_2$-norm penalty, generalizes the recently proposed $k$-support vector norm. The contributions of this work are two-fold. First, the proposed $k$-support matrix norm does not suffer from the disadvantages of existing matrix norms towards sparsity and/or low-rankness: 1) too sparse/dense, and/or 2) column independent. Second, we present an efficient procedure for $k$-support norm optimization, in which the computation of the key *proximity operator* is substantially accelerated by binary search. Extensive experiments on subspace segmentation, semi-supervised classification and sparse coding well demonstrate the superiority of the new regularizer over existing matrix-norm regularizers, and also the orders-of-magnitude speedup compared with the existing optimization procedure for the $k$-support norm.

**Keywords:** $k$-support norm, subspace segmentation, semi-supervised classification, sparse coding.

## 1 Introduction

We consider the following matrix pursuit problem:

$$\min_{W} \lambda \Omega(W) + \frac{1}{2}||A - BW||_F^2, \tag{1}$$

where $\lambda > 0$ is a non-negative trade-off parameter, $||\cdot||_F$ is the Frobenius norm of a matrix and $\Omega(W)$ is the regularization term. $A, B$ are two given matrices with compatible dimensions.

(1) is a typical problem formulation for many computer vision tasks. For example, with $A = B = X$ and $X$ being a set of column samples, (1) becomes the formulation of subspace segmentation [11,3] which seeks to learn an affinity matrix $W$ satisfying $X \approx XW$: $\min_W \lambda \Omega(W) + \frac{1}{2}||X - XW||_F^2$. With $A = X$ being a set of column samples (e.g., SIFT features [12]) and $B = D$ being an over-complete dictionary, (1) becomes

[⋆] Corresponding author.

D. Fleet et al. (Eds.): ECCV 2014, Part II, LNCS 8690, pp. 617–631, 2014.

**Table 1.** The characteristics of the five matrix norms. Here $w_i$ is the $i$th column of the matrix $W$ and $\text{Diag}(w_i)$ is the diagonal matrix whose diagonal elements are $w_i$.

| | **$\ell_1$-norm** | **Frobenius** | **Nuclear** | **Trace Lasso** | **$k$-Support** |
|---|---|---|---|---|---|
| **Form** | $\|\|W\|\|_1$ | $\|\|W\|\|_F$ | $\|\|W\|\|_*$ | $\sum_i \|\|B\text{Diag}(w_i)\|\|_*$ | $\|\|W\|\|_k^{sp}$ |
| **Sparsity** | Sparse | Dense | Dense | Balanced | Balanced |
| **Column Dependency** | Independent | Independent | Dependent | Independent | Dependent |
| **Efficiency** | Efficient | Efficient | Slow | Slow | Slow → Efficient |

the formulation of sparse coding for image classification [20,21]: $\min_W \lambda\Omega(W) + \frac{1}{2}||X - DW||_F^2$.

In this paper, we focus on designing the regularization term $\Omega(W)$. In the past few years, there has been considerable research work on matrix regularization. The representative regularization norms include $\ell_1$-norm [3], nuclear norm [11], Frobenius norm [13] and trace lasso [5]. In addition to the four representative norms, the $k$-support norm [1] has recently been proposed, which is the tightest convex relaxation of sparsity combined with $\ell_2$-norm penalty. Table 1 lists the characteristics of these norms.

Regularization with the $\ell_1$-norm [3,20] is a commonly-used technique to pursue sparse models for variable or feature selection, which makes the models more interpretable. However, it tends to over-shrink large coefficients, which often degrades the model accuracies. Besides, in the context of matrix pursuit, regularization with the $\ell_1$ norm usually seeks a sparse representation of each row/column of the target matrix individually, regardless of the possible correlation between these rows/columns. For example, in multi-task learning or sparse coding problems, different rows/columns of the target matrix may be related in some sense, and it is advantageous to simultaneously pursue these rows/columns [21].

Regularization with the Frobenius norm has a grouping effect [13] and can reveal the correlated information. More importantly, it has the closed form solution, which makes it very efficient. However, it fails to encourage the sparsity and is also column independent as the $\ell_1$-norm.

Regularization with the nuclear norm is used to approximate the rank of a matrix. For example, Low Rank Representation (LRR) [11] jointly optimizes the affinity matrix under a global low rank constraint which makes it better capture the global structure of the data. However, it often results in dense solutions as the Frobenius norm. Also solving the nuclear norm minimization problem requires Singular Value Decomposition (SVD) operation, which limits its scalability on large datasets.

Regularization with the trace lasso [5] is a data correlation dependent method which can adaptively balance the $\ell_1$-norm and the Frobenius norm. For instance, Correlation Adaptive Subspace Segmentation (CASS) [14] uses the trace lasso to exhibit both sparsity and grouping effect, which makes it effective for subspace clustering. However, it also requires SVD for optimization as LRR and is column independent as $\ell_1$ and Frobenius norms.

It can be seen that these four norms all suffer from one or two of the following disadvantages: 1) too dense/sparse solution, 2) column independency. To overcome these disadvantages, in this paper, we consider the matrix pursuit problem regularized with the $k$-support norm [1], i.e. $\Omega(W) = (||W||_k^{sp})^2$, where we denote $||W||_k^{sp}$ as the $k$-support norm of the matrix $W$. Specifically, assuming $W \in \mathbb{R}^{d\times n}$, and $w$ =vec($W$) represents the vector constructed by concatenating the columns of $W$, $||W||_k^{sp}$ is defined as:

$$||W||_k^{sp} = \Big(\sum_{i=1}^{k-r-1}(|w|_i^{\downarrow})^2 + \frac{1}{r+1}(\sum_{i=k-r}^{nd}|w|_i^{\downarrow})^2\Big)^{\frac{1}{2}}, \tag{2}$$

where $|w|_i^{\downarrow}$ denotes the $i$th largest element in $|w|$ and $|w|_0^{\downarrow}$ is assumed to be $+\infty$. $r \in \{0, 1, ..., k-1\}$ satisfies $|w|_{k-r-1}^{\downarrow} > \frac{1}{r+1}(\sum_{i=k-r}^{nd}|w|_i^{\downarrow}) \geq |w|_{k-r}^{\downarrow}$. The $k$-support norm has two terms: $\ell_2$-norm penalty for the large components, and $\ell_1$-norm penalty for the small components. $k(1 \leq k \leq nd)$ is a tunable parameter of the cardinality to achieve a balance between the $\ell_2$-norm (when $k = nd$) and the $\ell_1$-norm (when $k = 1$). The $k$-support matrix norm provides an appropriate trade-off between model sparsity (by $\ell_1$-norm) and algorithmic stability[1] (by Frobenius norm) [19], which yields more stable solutions than the $\ell_1$-norm. Moreover, in (1), the $k$-support matrix norm facilitates simultaneously learning all the columns of the target matrix, and thus transferring knowledge from one column to another to improve generalization performance.

The $k$-support norm regularization problem can be solved by the Accelerated Proximal Gradient (APG) method [2] or Alternating Direction Method [10] (ADM), which both require computing the *proximity operator* of the $k$-support norm. The time complexity of the method in [1] is $O(nd(k + log(nd)))$ when the $k$-support norm is applied on a $d \times n$ matrix. This time complexity is impractically high when $n$, $d$ and $k$ are large. To tackle this issue, we propose a novel efficient procedure, which reduces the time complexity to $O((nd + k)\log(nd))$. Since $k$ is usually set to be much larger than $n$ in practice, such time reduction (with a factor of $k$) can substantially accelerate the optimization procedure with the $k$-support norm.

The main contributions of this paper can be summarized as follows:

1) We propose the $k$-support matrix regularizer, which can well balance the sparsity and density of the solution, and is column dependent. We apply the proposed regularizer to several popular computer vision problems. The experiment results show that the $k$-support matrix regularizer outperforms the state-of-the-art popular norms/regularizers.

2) We propose an efficient procedure to solve the *proximity operator* of the $k$-support norm, which is $100 \sim 1000$ times faster than the method in [1] for moderate or large matrix pursuit problems.

## 2 *k*-Support Matrix Pursuit and Optimization

### 2.1 Problem Definition and Optimization Overview

In this paper, we particularly address the matrix pursuit problem with the $k$-support norm:

$$\min \frac{\lambda}{2}(||W||_k^{sp})^2 + \frac{1}{2}||A - BW||_F^2. \tag{3}$$

[1] Algorithmic stability means that given two similar datasets, the outputs of the algorithm change very little [15].

The $k$-support norm is a convex relaxation to seek a sparse, low $\ell_2$-norm linear predictor [1], as shown in the following convex hull:

$$C_k = \text{conv}\{w \in \mathbb{R}^{nd} \mid ||w||_0 \leq k, ||w||_2 \leq 1\},$$

where $w = vec(W)$. It consists of two parts: one is the $\ell_0$-norm constraint and the other is the $\ell_2$-norm constraint. The $k$-support norm is defined as the norm whose unit ball equals $C_k$. An equivalent but more intuitive definition is in Eq. (2).

The main advantage of the $k$-support norm is that it provides the flexibility of tuning the cardinality $k$ in $W$ (i.e., the number of non-zero elements in $W$). This parameter, which is often selected by cross validation, can be regarded as an upper bound estimation of the number of non-zero elements in the optimal $W$. The $k$-support regularized matrix pursuit with a fine-tuned $k$ has the potential to obtain more predictive [1] and more stable [19] solutions. Another advantage of the $k$-support norm is that it is column dependent, which makes it effective for selecting the informative variables cross all columns.

The optimization problem (3) can be solved by first-order proximal gradient algorithms, such as the Accelerated Proximal Gradient (APG) method as in [2]. As shown in Algorithm 1, the problem (3) is solved based on the APG scheme.

**Algorithm 1.** Accelerated Proximal Gradient method

```
Input: $A, B, \lambda, k, T$ and $L$.
Initialize: $W_0 = P_1 = 0$, $\alpha_1 = 1$, $M = B^T B$, $N = B^T A$.
For $t = 1, \cdots, T$
  Let $G_t = MP_t - N$ (gradient)
  $W_t = arg\min_W \frac{\lambda}{2}(||W||_k^{sp})^2 + \langle W - P_t, G_t \rangle + \frac{L}{2}||W - P_t||_F^2$ (proximity operator)
  $\alpha_{t+1} = \frac{1+\sqrt{1+4\alpha_t^2}}{2}$
  $P_{t+1} = W_t + \Big(\frac{\alpha_t - 1}{\alpha_{t+1}}\Big)(W_t - W_{t-1})$
End For
output: $W_T$.
```

In Algorithm 1, the *proximity operator* is a computationally expensive step in each iteration:

$$\min_W \frac{\beta}{2}||W - V||_F^2 + \frac{1}{2}(||W||_k^{sp})^2, \tag{4}$$

where $V = P_t - G_t/L$ and $\beta = L/\lambda$. It is equivalent to the following problem:

$$\min_w \frac{\beta}{2}||w - v||_F^2 + \frac{1}{2}(||w||_k^{sp})^2, \tag{5}$$

where $w = \text{vec}(W) \in \mathbb{R}^{nd}$, and $v = \text{vec}(V) \in \mathbb{R}^{nd}$. Argyriou *et al.* [1] proposed an algorithm to solve (5) with time complexity $O(nd(k + \log(nd)))$. Since the parameter $k$ is regarded as an upper bound estimation of the number of non-zero elements in $W$, $k$ is usually set to be much larger than $n$ (e.g., $k = O(nd)$). Hence, the computation of the *proximity operator* generally dominates the whole algorithm, and its complexity

$O(nd(k + \log(nd))) \approx O(ndk)$ is impractically high for matrix pursuit problems with large $n$, $d$ and $k$.

To accelerate Algorithm 1, in the next subsection, we propose a much more efficient procedure to solve the *proximity operator* problem, where the time complexity is reduced to $O((nd + k)\log(nd))$.

### 2.2 Binary Search Procedure to Solve the Proximity Operator

In this subsection, we present an efficient procedure to solve (5) based on binary search [8]. For ease of description, some notations are defined. We define $z$ as the vector obtained by sorting the absolute values of the elements in $v$ in a descending order, i.e., $z = |v|^{\downarrow}$. We also define $s_i$ as $z_i$'s corresponding index in $v$: $|v_{s_i}| = z_i$. The sign function is defined as: sign$(x) = 1$ if and only if $x \geq 0$, and otherwise sign$(x) = -1$.

According to the work [1], the solution $w$ of the optimization problem (5) can be obtained by:

$$w_{s_i} = \text{sign}(v_{s_i})q_i, \tag{6}$$

where $q_i$ is defined by:

$$q_i = \begin{cases} \frac{\beta}{\beta+1} z_i & \text{if } i = 1, ..., k-r-1; \\ z_i - \frac{\sum_{i=k-r}^{l} z_i}{l-k+r+1+\beta(r+1)} & \text{if } i = k-r, ..., l; \\ 0 & \text{if } i = l+1, ..., nd. \end{cases} \tag{7}$$

In (7), the integers $r$ and $l$ must satisfy

$$\begin{cases} \frac{1}{\beta+1} z_{k-r-1} > \frac{\sum_{i=k-r}^{l} z_i}{\beta(r+1)+l-k+r+1} \geq \frac{1}{\beta+1} z_{k-r}; \\ z_l > \frac{\sum_{i=k-r}^{l} z_i}{\beta(r+1)+l-k+r+1} \geq z_{l+1}. \end{cases} \tag{8}$$

To obtain every $w_i$ (and corresponding $q_i$), two integers $r$ and $l$ must be efficiently found for (7) which satisfy the constraints in (8), which is very challenging. For example, in Algorithm 1 in the work [1], the time complexity $O(ndk)$ is needed to find the integer pair $r, l$ for all $q_i$, $i = 1, 2, ..., nd$. Since $k$ is usually set to be much larger than $n$, this complexity is unaffordable in large matrix pursuit problems.

To tackle this issue, we propose a new method to find the integer pair $r, l$ for all $q_i$ based on binary search. The proposed procedure to solve (5) is shown in Algorithm 2, in which we define $T_{r,l} = \sum_{i=k-r}^{l} z_i$. In each iteration, given $r$, we search for an $l$ that satisfies the second condition in (8), by a binary search procedure (see Algorithm 3). If the obtained $l$ also satisfies the first condition in (8), we jump out of the loop and calculate the final $w$ with the obtained $l$ and $r$. Otherwise we increase $r$ by one and continue the loop.

To efficiently obtain $l$ that satisfies the second condition in (8), we propose to use a binary search scheme as shown in Algorithm 3. In Algorithm 3, we search for $l$ in the range from $k - r$ to $nd$.

If $z_{low}$ is zero, which implies that $z_j$, $j = low, ..., high$, are all zeros, we obtain $l = low = k - r$. Otherwise, we use a trick like the binary search scheme [8] to reduce the search range by half in each iteration. Specifically, with the assumption that the obtained $l$ satisfies the second condition in (8), we consider two cases:

**Algorithm 2.** Procedure to solve the *proximity operator*

```
Input: $v \in R^{nd}$, $\beta$, $k$.
Initialize: $z = |v|^{\downarrow}$, $z_0 = +\infty$, $z_{nd+1} = -\infty$.
For r = k-1,...,0
    Obtain $l$ by BinarySearch(z,k-r,nd) (see Algorithm 3);
    If $\frac{1}{\beta+1} z_{k-r-1} > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} \geq \frac{1}{\beta+1} z_{k-r}$
        break;
    End If
End For
For $i = 1, 2, ..., nd$, calculate $q_i$ by (7);
For $i = 1, 2, ..., nd$, calculate $w_i$ by (6);
Output: $w$.
```

- **Case 1.** If we also have $z_{mid} > \frac{T_{r,mid}}{mid-k+r+1+\beta(r+1)}$ for a certain index $mid$, then we can prove that $mid \leq l$ (see Lemma 1). Hence, we only need to search for $l$ in the range $[mid, high]$, by ignoring the search in $[low, mid-1]$.
- **Case 2.** If we also have $z_{mid} \leq \frac{T_{r,mid}}{mid-k+r+1+\beta(r+1)}$, then we can prove that $l \leq mid - 1$ (see Lemma 1). Hence, we only need to search for $l$ in the range $[low, mid-1]$, by ignoring the search in $[mid, high]$.

By using these facts, we follow the binary search scheme to design a loop to find $l$. In each iteration, we set $mid$ to be $\ulcorner \frac{low+high}{2} \urcorner$ (Line 7 in Algorithm 3) where $\ulcorner x \urcorner$ represents the smallest integer which is larger than $x$. Then we reduce the search range of $l$ by either Case 1 (Line 9 in Algorithm 3) or Case 2 (Line 11 in Algorithm 3).

**Algorithm 3.** Binary search procedure to find $l$

```
1.  Input: $z, low, high$,
2.  Output: $l$.
3.  If $z_{low} = 0$
4.      return $l$ = $low$.
5.  End If
6.  While $low < high - 1$
7.      $mid = \ulcorner \frac{low+high}{2} \urcorner$.
8.      If $z_{mid} > \frac{T_{r,mid}}{mid-k+r+1+\beta(r+1)}$
9.          $low = mid$.
10.     Else
11.         $high = mid - 1$.
12.     End If
13. End While
14. return $l = low$.
```

**Justification.** We give the following two lemmas to prove the correctness of Algorithm 3. The detailed proof of these lemmas can be found in the supplementary material.

Lemma 1 provides the foundation to design a binary search for $l$ in Algorithm 3 so that we can efficiently reduce the search range.

**Lemma 1.** *Suppose that* $l$ *satisfies* $z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} \geq z_{l+1}$. *Let* $low, high, mid$ *be variables generated by Algorithm 3. Then*
*(1) If* $z_{mid} > \frac{T_{r,mid}}{mid-k+r+1+\beta(r+1)}$ *then* $mid \leq l$*;*
*(2) If* $z_{mid} \leq \frac{T_{r,mid}}{mid-k+r+1+\beta(r+1)}$ *then* $l \leq mid - 1$.

A natural question arises in Algorithm 3: whether there always is an $l$ satisfying the second condition in (8) in the range $[k-r, d]$? The answer is "yes" by the following lemma.

**Lemma 2.** *For any* $r \in [0, k-1]$*, if* $z_{k-r} > 0$*, there exists an* $l$ *such that* $z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} \geq z_{l+1}$ *and* $l \in [k-r, nd]$*. If* $z_{k-r} = 0$*, we can obtain* $l = k-r$.

## 3 Applications of $k$-Support Matrix Pursuit

The matrix pursuit problem is the core for various vision applications. In this section, we apply the $k$-support regularizer to the following popular vision related problems: subspace clustering, semi-supervised learning and sparse coding, and show how these tasks benefit from the $k$-support regularizer.

### 3.1 Subspace Segmentation and Semi-supervised Learning

Subspace segmentation aims to partition a set of data points into multiple (linear or affine) subspaces. It is a widely-used technique in various visual tasks, such as face clustering [11] and motion segmentation [3].

Semi-supervised learning has received considerable attention in computer vision literature. In this work, we consider the graph based semi-supervised learning (SSL) [22], where the label information can be propagated to the unlabeled data through a certain affinity matrix.

A core step in both subspace segmentation and graph based semi-supervised learning is to learn a good affinity matrix to represent the structure of the data points. Here, we describe how to apply the $k$-support graph for these two tasks.

Let $X = [x_1, x_2, \cdots, x_n] \in \mathbb{R}^{m \times n}$ be a matrix whose columns are $n$ data samples of $m$-dimension. Suppose that these samples are drawn from different subspaces, and each sample can be expressed as a linear combination of all the data points $X$: $x_i = Xw_i$, where $w_i$ characterizes how other samples contribute to the reconstruction of $x_i$. Since $x_i$ should be associated with only a few samples drawn from the same subspace, $W$ should be sparse but not too sparse. Thus we use the $k$-support norm to infer such an affinity matrix: $\min_W \frac{1}{2}(||W||_k^{sp})^2, s.t.\ X = XW$. For the case with noises, we consider the following problem instead:

$$\min_W \frac{\lambda}{2}(||W||_k^{sp})^2 + \frac{1}{2}||X - XW||_F^2, \tag{9}$$

where $\lambda > 0$ is a parameter for controlling the tradeoff between the model complexity and the reconstruction error.

The detailed steps of the two learning tasks based on the inferred $k$-support norm are as follows. First, a good affinity matrix $W$ is learned, that is, solving the problem (9), after which we symmetrize it by $W = (|W| + |W^T|)/2$. Then, for subspace clustering, the spectral clustering algorithm [17] is used on the matrix $W$ to segment data, while for semi-supervised learning, the Markov random walks algorithm [16] is adopted to propagate label information to the unlabeled data through the matrix $W$.

### 3.2 Collaborative Sparse Coding

Let $X$ be a set of $m$-dimensional local descriptors extracted from an image, i.e., $X = [x_1, x_2, \cdots, x_n] \in \mathbb{R}^{m \times n}$. The dictionary $D \in \mathbb{R}^{m \times d}$ contains $d$ bases. Sparse coding is a task which seeks a sparse linear combination of the bases from an over-complete dictionary $D$, to recover the input signal $x$ with low reconstruction errors.

Two representative works are locally linear codes (LLC) [18] and the $\ell_1$-norm regularized sparse coding (ScSPM) [20]. LLC chooses the $k$ nearest bases of the input $x_i$ within the dictionary $D$ by Euclidean distance: $\min_W \sum_{i=1}^n ||x_i - Dw_i||_2^2 + \lambda(||h_i \odot w_i||)^2, s.t.\ 1^T w_i = 1, \forall i$, where $\odot$ denotes the element-wise multiplication, and $h_i$ is the locality adaptor, i.e., the similarity between each basis vector and the input descriptor. ScSPM minimizes the following optimization problem: $\min_W \sum_{i=1}^n ||x_i - Dw_i||_2^2 + \lambda||w_i||_1$.

Suppose each image is partitioned into $M$ sub-regions, and the $j$th sub-region $X_j$ includes $n_j$ local descriptors: $X_j = [x_{j_1}, \ldots, x_{j_{n_j}}]$. If $M$ is large, the local descriptors within the same sub-region would be similar or have high correlations to some extent, and the sparse codes of these local descriptors would have correlated patterns. Hence, from the viewpoint of multi-task learning, it is natural to jointly learn the sparse codes for the local descriptors within the same sub-region. However, both ScSPM and LLC solve the sparse coding problem of each local descriptor individually, regardless of the possible correlations of descriptors in the same sub-region. To address this problem, we propose collaborative sparse coding using the $k$-support matrix norm:

$$\min_W \sum_{j=1}^{M} \frac{1}{2}||X_j - DW_j||_2^2 + \frac{\lambda}{2}(||W_j||_k^{sp})^2, \tag{10}$$

where $W_j$ is the sparse code matrix of the descriptors in the $j$th sub-region. Note that all the elements in $W_j$ are sorted in a descending order, with $\ell_2$-norm penalty being applied to the large elements and $\ell_1$-norm penalty being applied to the small elements. Hence, $||W_j||_k^{sp}$ can be regarded as a global regularizer on the $j$th sub-region, where the informative variables in $W_j$ are assigned with large values and the irrelevant variables are assigned with small values. Moreover, Xu *et al.* [19] showed that sparsity and algorithmic stability are two desired properties for designing a leaning algorithm. Xu *et al.* [19] proved that $\ell_1$-regularized regression is sparse but not stable, while $\ell_2$-regularized regression has strong stability. Hence, for some similar local descriptors (e.g., they are from the same sub-region), ScSPM would output different codes although they have high correlations. Intuitively, it is benefitial to consider these correlated descriptors together and collaborative sparse coding is desirable. The $k$-support matrix norm makes a trade-off between model sparsity (by $\ell_1$-norm) and model stability (by Frobenius norm), and also is column dependent. Hence, it is a natural choice for collaborative sparse coding.

# 4 Experiments

In this section, we evaluate the effectiveness of the proposed $k$-support matrix regularization on three applications: 1) subspace segmentation, 2) semi-supervised classification and 3) sparse coding for image classification. Also we evaluate in detail the running time of our solver for the $k$-support norm by comparing it with the previous method shown in [1].

## 4.1 Experiments on Subspace Segmentation

We apply the proposed algorithm to solve the $k$-support matrix regularized subspace segmentation problem (9). The performance is compared with the following baselines: Sparse Subspace Clustering (SSC) [3], Low-Rank Representation (LRR) [11], Least Squares Regression (LSR) [13], Non-Negative Low-Rank and Sparse graph (NNLRS) [22] and Correlation Adaptive Subspace Segmentation (CASS) [14]. Our proposed method is referred to as KMP for **K**-support norm regularized **M**atrix **P**ursuit. The results of SSC, LRR, LSR and CASS are cited from [14]. The results of NNLRS and KMP are obtained by our careful implementations. For fair comparison, we follow the experimental settings as [14], where the parameters are tuned to the best for each method.

After constructing the affinity matrix, the normalized cuts method [17] is used to segment the data into different subspaces. The clustering result is evaluated by the accuracy/error, where the accuracy is calculated by the best matching rate of the predicted label and the ground truth, while the error rate is calculated by $1 -$ accuracy.

The experiments are conducted on Extended Yale B [4] for face clustering, and on Hopkins 155 [2] for motion segmentation.

The Extended Yale B dataset consists of 16,128 images of 28 human subjects with 9 poses and 64 illumination conditions. Following the settings in [14], we conduct two experiments on Extended Yale B. In the first experiment, we use the images in the first 5 classes, and then project the images onto 30 dimensions by principal component analysis (PCA). In the second experiment, we use the samples in the first 10 classes, and the images are projected onto 60 dimensions by PCA. Each class contains 64 images and each image is resized into 32×32 pixels. Table 2 shows the segmentation accuracy of each method.

The Hopkins 155 motion database contains 156 video sequences and each sequence has two or three motions (each motion corresponds to a subspace). Since each sequence is a sole data set, there are 156 subspace segmentation problems in total. We project the data onto 12 dimensions by PCA as in [14]. Table 3 shows the maximum, mean and standard deviation of the error rates of the 156 sequences.

The comparison results on Extended Yale B database and Hopkins 155 motion database indicate that KMP performs better than all the baselines. For example, in the first 5 classes, KMP achieves 94.69% accuracy, which indicates an improvement of 14.38% over SSC and 2.50% over LSR. The experimental results can empirically

[2] `http://www.vision.jhu.edu/data/hopkins155/`

**Table 2.** The accuracies of different algorithms on the Extended Yale B database (%). Note that the results of SSC, LRR, LSR and CASS are directly cited from [14].

| | SSC [14] | LRR [14] | LSR [14] | NNLRS | CASS [14] | KMP |
|---|---|---|---|---|---|---|
| **5** | 80.31 | 86.56 | 92.19 | 89.69 | 94.03 | **94.69** |
| **10** | 52.19 | 65.00 | 73.59 | 79.37 | 81.88 | **82.25** |

**Table 3.** The error rates of different algorithms on the Hopkins 155 database (%). Note that the results of SSC, LRR, LSR and CASS are directly cited from [14].

| | SSC [14] | LRR [14] | LSR [14] | NNLRS | CASS [14] | KMP |
|---|---|---|---|---|---|---|
| **Max** | 39.53 | 36.36 | 36.36 | 33.09 | **32.85** | 39.58 |
| **Mean** | 4.02 | 3.23 | 2.50 | 3.27 | 2.42 | **1.99** |
| **STD** | 10.04 | 6.06 | 5.62 | 6.82 | 5.84 | **4.93** |

justify that balancing the $\ell_1$-norm and the Frobenius norm can help significantly improve the accuracy. On Hopkins 155 motion database, the mean error of KMP is 1.99%, compared with 2.42% of the second best algorithm CASS. KMP is column dependent, which may be the underlying reason for KMP to outperform CASS.

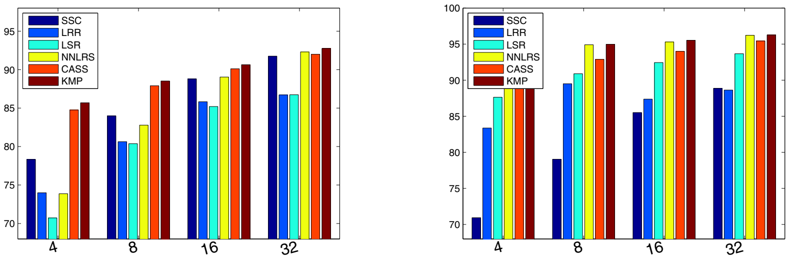
### 4.2 Experiments on Semi-supervised Classification

For semi-supervised classification, we compare the performances of the algorithms on two datasets: Extended Yale B and USPS [6]. USPS [3] is a handwritten digit dataset which has ten classes (the digits range from 0 to 9).

Following the settings in [14], on Extended Yale B, we use the images in the first 10 classes. On USPS, we randomly select 100 images from each class for our experiments. Each image in USPS is resized into 16×16 pixels. In all experiments, we randomly select 4, 8, 16, 32 images from each class as labeled samples, respectively, and the rest images are used as unlabeled samples. After obtaining the matrix $W$, we select the Markov random walks algorithm [16] for semi-supervised learning. For fair comparison with previous work, the parameters of all methods are tuned to achieve the best performance. We run each experiment for 20 times and report the averaged accuracies.

Fig. 1 shows the comparison results, from which two observations can be made. First, the $k$-support matrix norm can well balance the $\ell_1$-norm and the Frobenius norm. SSC ($\ell_1$-norm) achieves a good performance on USPS dataset, but performs poorly on Extended Yale B. LSR (Frobenius norm) performs well on Extended Yale B, but yields a poor performance on USPS. Compared with them, the $k$-support matrix norm performs well on both datasets. Second, the KMP performs better than CASS. The reason may be that the $k$-support matrix norm can capture the cross column information while CASS cannot.

[3] http://www.cad.zju.edu.cn/home/dengcai/Data/MLData.html



(a) USPS

(b) Extended Yale B

**Fig. 1.** The accuracies of semi-supervised classification based on different affinity matrices (%). Note that the results of SSC, LRR, LSR and CASS are directly cited from [14] on Extended Yale B database, and obtained by directly running their open source codes on USPS database.

### 4.3 Experiments on Collaborative Sparse Codings

The proposed algorithm is also applied to the sparse coding problem (10) for image classification, and compared with two state-of-the-art sparse coding methods: ScSPM [20] and LLC [18]. The performances are evaluated on two datasets: UIUC-Sport [9] and Scene15 [7].

On each dataset, we obtain a dictionary containing 1,024 bases using the $k$-means clustering. The results of ScSPM and LLC are obtained by directly using the online source codes[4]. In our image classification system, we use the open source code of ScSPM [20] to implement SIFT extraction, spatial pyramid matching and max pooling. For our collaborative sparse coding, we partition each image into a set of $20 \times 20$ non-overlap sub-regions. Then the proposed $k$-support matrix pursuit algorithm is performed on each sub-region to obtain the sparse codes, respectively.

**Results on UIUC-Sport Dataset.** The UIUC-Sport dataset contains images collected from 8 classes of different sports such as badminton, rock climbing and sailing. There are 1,579 images in total and the number of images in each category ranges from 137 to 250. For each category, we randomly select 70 images as training data, and another 60 images as test data. As shown in Table 4, the $k$-support regularized sparse coding method shows superior performance over ScSPM and LLC.

**Table 4.** Image classification results on UIUC-Sport dataset(%)

| Method | Average Classification Accuracy |
|---|---|
| ScSPM [20] | 82.74 ± 1.46 |
| LLC [18] | 84.58 ± 1.19 |
| KMP | **85.55 ± 1.23** |

[4] www.ifp.illinois.edu/~jyang29

**Results on Scene15 Dataset.** The Scene15 dataset is frequently used for scene classification. It contains 4,485 images. The images are collected from 15 categories such as forest, highway and mountain. Each category contains 200 to 400 images. For each category, we randomly select 100 images as the training data and use the remaining images as the test data. The proposed method outperforms ScSPM and LLC.

**Table 5.** Image classification results on Scene15 dataset(%)

| Method | Average Classification Accuracy |
|---|---|
| ScSPM [20] | 80.28 ± 0.93 |
| LLC [18] | 81.57 ± 0.50 |
| KMP | **83.04 ± 0.43** |

From the two results, we can observe that it is advantageous to jointly learn the sparse codes within the same sub-region. KMP also considers the sparsity and stability of the algorithm, while ScSPM (with $\ell_1$ norm) is not stable [19], which may be the underlying reason for the better performance of KMP than ScSPM.

### 4.4 Experiments on Algorithmic Complexity

In this subsection, we evaluate the running time of our solver by comparing it with the previous method shown in [1]. We use the open source code of the $k$-support norm [5] to make a fair comparison. Since it uses the linear search algorithm, we call it "LS-KMP". Our method is referred to as "BS-KMP", which means using binary search for the $k$-support norm.

We report the running time of LS-KMP and BS-KMP on all the above databases. Note that both algorithms use the same implementation except for the computation of the *proximity operator*. The running time shown in Table 6 is the cost to compute the affinity or sparse code matrix. For Scene15 and UIUC datasets, we only report the average running time of the first 10 images. We can see that the proposed procedure performs 2 or 3 orders of magnitude faster than the LS-KMP on the first three datasets. On Scene15 and UIUC datasets, BS-KMP only has 10 times faster than the LS-KMP. This is because the value of $k$ is set to be a small integer in these two datasets, e.g., $k = 10$.

**Table 6.** Running time comparison between LS-KMP and BS-KMP on five real datasets (in sec)

| | **YaleB** | **Hopkins** | **USPS** | **Scene15** | **UIUC** |
|---|---|---|---|---|---|
| **LS-KMP** | 3530 | 66174 | 30707 | 31.14 | 33.16 |
| **BS-KMP** | **5.25** | **400.55** | **191.05** | **3.31** | **3.01** |

[5] http://cvc.centrale-ponts.fr/personnel/andreas/

We conduct simulated experiments to observe the effects of the parameters on the running time of the *proximity operator* with different values of $\beta$ and $k$:

$$\min_{w} \frac{\beta}{2}||w - v||^2 + \frac{1}{2}(||w||_k^{sp})^2. \tag{11}$$

We generate a vector $v \in \mathbb{R}^{10000}$, which is randomly sampled from the norm distribution $N(0, 1)$. For each setting, the running time is averaged over 10 different $v$.

**Parameter $k$.** Table 7 shows how the running time varies with the different values of the predefined parameter $k$. The parameter $k$ is chosen from the set $\{1000, 2000, \cdots, 9000\}$. We fix $\beta = 10$ in these experiments.

**Table 7.** Running time comparison under different $k$ on synthetic data (in sec)

| k | 1000 | 2000 | 3000 | 4000 | 5000 | 6000 | 7000 | 8000 | 9000 |
|---|---|---|---|---|---|---|---|---|---|
| **LS-KMP** | 6.1842 | 12.4235 | 16.3917 | 18.0764 | 17.8485 | 16.1820 | 13.3029 | 9.5786 | 5.0788 |
| **BS-KMP** | **0.0020** | **0.0024** | **0.0032** | **0.0083** | **0.0159** | **0.0252** | **0.0371** | **0.0497** | **0.0627** |

It can be seen that the proposed method runs significantly faster than the LS-KMP, especially when $k$ is set near half value of the dimension, e.g., $k = [4000, 5000, 6000]$. Table 7 shows that BS-KMP is $81 \sim 5177$ times faster than the LS-KMP on simulated data.

**Parameter $\beta$.** Table 8 shows how the running time varies with the different values of $\beta$. The parameter $\beta$ is chosen from the set $\{0.01, 0.1, 1, 10, 100\}$. We fix $k = 5000$ in these experiments. As shown in Table 8, the running time of LS-KMP sharply changes with different $\beta$, while BS-KMP is more stable.

**Table 8.** Running time comparison under different $\beta$ on synthetic data (in sec)

| $\beta$ | 0.01 | 0.1 | 1 | 10 | 100 |
|---|---|---|---|---|---|
| **LS-KMP** | 0.0737 | 0.7016 | 5.6410 | 17.8485 | 22.4130 |
| **BS-KMP** | **0.0401** | **0.0374** | **0.0293** | **0.0159** | **0.0124** |

## 5 Conclusions and Future Work

In this paper, we proposed to use the $k$-support matrix norm to address the matrix pursuit problem, which is valuable for a wide range of applications including subspace segmentation, semi-supervised classification and sparse coding. The $k$-support matrix norm is advantageous and provides a flexible trade-off between model sparsity and prediction accuracy. To tackle the time-consuming *proximity operator* problem of the $k$-support norm, we also developed an efficient binary search procedure, which can substantially reduce the running time. Extensive experiments showed the superior effectiveness of the $k$-support matrix norm over the state-of-the-art baseline norms.

In future work, we plan to study the effects of using the $k$-support matrix norm for the construction error term, and intend to investigate the relationship between the $k$-support loss function and the outlier/Gaussian noise of the data.

**Acknowledgements.** This work was funded in part by National Science Foundation of China (grant No. 61370021, 61003045, 61100080), Natural Science Foundation of Guangdong Province, China (grant No. S2013010011905). The corresponding author is Yan Pan.

## References

1. Argyriou, A., Foygel, R., Srebro, N.: Sparse prediction with the $k$-support norm. In: Advances in Neural Information Processing Systems, pp. 1466–1474 (2012)
2. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences 2(1), 183–202 (2009)
3. Elhamifar, E., Vidal, R.: Sparse subspace clustering. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2790–2797 (2009)
4. Georghiades, A.S., Belhumeur, P.N., Kriegman, D.J.: From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(6), 643–660 (2001)
5. Grave, E., Obozinski, G., Bach, F.: et al.: Trace lasso: a trace norm regularization for correlated designs. Advances in Neural Information Processing Systems 2, 5 (2011)
6. Hull, J.J.: A database for handwritten text recognition research. IEEE Transactions on Pattern Analysis and Machine Intelligence 16(5), 550–554 (1994)
7. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2169–2178 (2006)
8. Lewis, G.N., Boynton, N.J., Burton, F.W.: Expected complexity of fast search with uniformly distributed data. Information Processing Letters 13(1), 4–7 (1981)
9. Li, L.J., Fei-Fei, L.: What, where and who? classifying events by scene and object recognition. In: IEEE International Conference on Computer Vision, pp. 1–8 (2007)
10. Lin, Z., Liu, R., Su, Z.: Linearized alternating direction method with adaptive penalty for low-rank representation. In: Advances in Neural Information Processing Systems, pp. 612–620 (2011)
11. Liu, G., Lin, Z., Yu, Y.: Robust subspace segmentation by low-rank representation. In: International Conference on Machine Learning, pp. 663–670 (2010)
12. Lowe, D.G.: Object recognition from local scale-invariant features. In: IEEE International Conference on Computer Vision, vol. 2, pp. 1150–1157 (1999)
13. Jia, K., Chan, T.-H., Ma, Y.: Robust and efficient subspace segmentation via least squares regression. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 331–344. Springer, Heidelberg (2012)
14. Lu, C., Feng, J., Lin, Z., Yan, S.: Correlation adaptive subspace segmentation by trace lasso. In: IEEE International Conference on Computer Vision (2013)
15. Shalev-Shwartz, S., Shamir, O., Sridharan, K., Srebro, N.: Learnability and stability in the general learning setting. In: Annual Conference of Learning Theory (2009)

16. Szummer, M., Jaakkola, T.: Partially labeled classification with markov random walks. In: Advances in Neural Information Processing Systems, pp. 945–952 (2001)
17. Von Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing 17(4), 395–416 (2007)
18. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3360–3367 (2010)
19. Xu, H., Caramanis, C., Mannor, S.: Sparse algorithms are not stable: A no-free-lunch theorem. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(1), 187–193 (2012)
20. Yang, J., Yu, K., Gong, Y., Huang, T.: Linear spatial pyramid matching using sparse coding for image classification. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1794–1801 (2009)
21. Yuan, X.T., Liu, X., Yan, S.: Visual classification with multitask joint sparse representation. IEEE Transactions on Image Processing 21(10), 4349–4360 (2012)
22. Zhuang, L., Gao, H., Lin, Z., Ma, Y., Zhang, X., Yu, N.: Non-negative low rank and sparse graph for semi-supervised learning. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2328–2335 (2012)

# Supplementary Material of "Efficient $k$-Support Matrix Pursuit"

Anonymous ECCV submission

Paper ID 384

In this supplementary material, we prove the two lemma in the main paper.

**Lemma 1.** *Suppose that* $l$ *satisfies* $z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} \geq z_{l+1}$. *Let* $low, high, mid$ *are variables generated by the Algorithm 3, Then*
*(1) If* $z_{mid} > \frac{T_{r,mid}}{mid-k+r+1+\beta(r+1)}$ *then* $mid \leq l$.
*(2) If* $z_{mid} \leq \frac{T_{r,mid}}{mid-k+r+1+\beta(r+1)}$ *then* $l \leq mid - 1$.

And

**Lemma 2.** *For any* $r$ *such that* $r \in [0, k-1]$. *If* $z_{k-r} > 0$, *it definitely exists a* $l$ *satisfies* $z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} \geq z_{l+1}$ *in the range of* $[k-r, nd]$. *If* $z_{k-r} = 0$, *we can return* $l = k - r$.

Before we prove the Lemma 1, we need the following two lemmas.

**Lemma 3.** *If* $z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)}$, *then* $z_{l-1} > \frac{T_{r,l-1}}{l-1-k+r+1+\beta(r+1)}$, *where* $T_{r,l} = \sum_{i=k-r}^{l} z_i$.

*Proof.*

$$\begin{aligned} & z_{l-1} * (l - 1 - k + rL + 1 + \beta(r+1)) \\ = & \; z_{l-1} * (l - k + r + 1 + \beta(r+1)) - z_{l-1} \\ \geq & \; z_l * (l - k + r + 1 + \beta(r+1)) - z_{l-1} \\ > & \; T_{r,l} - z_{l-1} \\ = & \; T_{r,l-1} \end{aligned} \tag{1}$$

where the first inequality follows $z_{l-1} \geq z_l$ and the second inequality is the assumption. ■

**Lemma 4.** *If* $z_l \leq \frac{T_{r,l}}{l-k+r+1+\beta(r+1)}$, *then* $z_{l+1} \leq \frac{T_{r,l+1}}{l+1-k+r+1+\beta(r+1)}$.

*Proof.*

$$\begin{aligned} & z_{l+1} * (l + 1 - k + r + 1 + \beta(r+1)) \\ = & \; z_{l+1} * (l - k + r + 1 + \beta(r+1)) + z_{l+1} \\ \leq & \; z_l * (l - k + r + 1 + \beta(r+1)) + z_{l+1} \\ \leq & \; T_{r,l} + z_{l+1} \\ = & \; T_{r,l+1} \end{aligned} \tag{2}$$

where the first inequality follows $z_l \geq z_{l+1}$ and the second inequality is the assumption. ■

According to Lemma 3 and Lemma 4, we can proof the Lemma 1.

*Proof.* (1) Assumed that the claim does not hold. Thus, we have $l+1 \leq mid$. According to $z_{mid} > \frac{T_{r,mid}}{mid-k+r+1+\beta(r+1)}$ and Lemma 3, we know $z_{l+1} > \frac{T_{r,l+1}}{l+1-k+r+1+\beta(r+1)}$. Hence

$$\begin{aligned}
& z_{l+1} > \frac{T_{r,l+1}}{l+1-k+r+1+\beta(r+1)} \\
\rightarrow\ & z_{l+1}(l-k+r+1+\beta(r+1)) + z_{l+1} > T_{r,l+1} \\
\rightarrow\ & z_{l+1}(l-k+r+1+\beta(r+1)) > T_{r,l} \\
\rightarrow\ & z_{l+1} > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)}
\end{aligned} \tag{3}$$

This is a contradiction with $\frac{T_{r,l}}{l-k+r+1+\beta(r+1)} \geq z_{l+1}$.

(2) Assumed that the claim does not hold. Thus, we have $l \geq mid$. According to $z_{mid} \leq \frac{T_{r,mid}}{mid-k+r+1+\beta(r+1)}$ and Lemma 4, we know $z_l \leq \frac{T_{r,l}}{l-k+r+1+\beta(r+1)}$. This is a contradiction with $z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)}$. ■

Now, we proof the Lemma 2.

*Proof.* First, when $z_{k-r} = 0$, it means that $z_{k-r+1} = ... = z_{nd} = 0$. In such case, there is not exist a $l$ satisfies $z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)}$. Any $l \in [k-r, nd]$ can be returned and not influence the result. Hence, we can simply let $l = k-r$.

Now, we consider the case of $z_{k-r} > 0$. To proof $l \in [k-r, nd]$, we only need to show (1) $z_{k-r} > \frac{T_{r,k-r}}{k-r-k+r+1+\beta(r+1)}$ and (2) $z_{nd+1} \leq \frac{T_{r,nd+l}}{nd+1-k+r+1+\beta(r+1)}$. If both (1) and (2) are satisfied, according to Lemma 1, we have $l \geq k-r$ and $l \leq nd$. Hence $l \in [k-r, nd]$.

It is easy to verify that both (1) and (2) are true. Since $z_{k-r} - \frac{T_{r,k-r}}{k-r-k+r+1+\beta(r+1)} = z_{k-r} - \frac{z_{k-r}}{1+\beta(r+1)} > 0$, we have the (1). Since $z_{d+1} = -\infty$, it is less than or equal to any value, hence we have the (2).

Now, we show which $l \in [k-r, nd]$ satisfies the inequalities. Since both (1) and (2) are true, this indicates that we can find at least a $l \in [k-r, nd]$ satisfies $z_l - \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} > 0$ and $z_{l+1} - \frac{T_{r,l+1}}{l+1-k+r+1+\beta(r+1)} \leq 0$.

We have $z_{l+1} - \frac{T_{r,l+1}}{l+1-k+r+1+\beta(r+1)} \leq 0 \Rightarrow z_{l+1} \leq \frac{T_{r,l}+z_{l+1}}{l+1-k+r+1+\beta(r+1)} \Rightarrow (l+1-k+r+1+\beta(r+1))z_{l+1} \leq T_{r,l} + z_{l+1} \Rightarrow (l-k+r+1+\beta(r+1))z_{l+1} \leq T_{r,l} \Rightarrow z_{l+1} \leq \frac{T_{r,l}}{l-k+r+1+\beta(r+1)}$.

Hence,$z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} > z_{l+1}$. we find the $l$. ■

**Lemma 5.** *If $z_{k-r} > 0$, there is an unique $l$ satisfies $z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} \geq z_{l+1}$ in the range of $[k-r, d]$.*

*Proof.* Since $z_l > \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} \geq z_{l+1}$, we have (1) $z_l - \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} > 0$ and (2) $z_{l+1} - \frac{T_{r,l+1}}{l+1-k+r+1+\beta(r+1)} \leq 0$. (According to the Lemma 2).

We show there is an unique $l \in [k-r, d]$ satisfies the inequalities. Assumed the claim does not hold. Thus, there exists $l$ and $\hat{l}$ for which $l < \hat{l}$, we have

$$\begin{cases} z_l - \frac{T_{r,l}}{l-k+r+1+\beta(r+1)} > 0 \\ z_{l+1} - \frac{T_{r,l+1}}{l+1-k+r+1+\beta(r+1)} \leq 0 \end{cases}$$

and

$$\begin{cases} z_{\hat{l}} - \frac{T_{r,\hat{l}}}{\hat{l}-k+r+1+\beta(r+1)} > 0 \\ z_{\hat{l}+1} - \frac{T_{r,\hat{l}+1}}{\hat{l}+1-k+r+1+\beta(r+1)} \leq 0 \end{cases}$$

Since $z_{l+1} - \frac{T_{r,l+1}}{l+1-k+r+1+\beta(r+1)} \leq 0$ and $\hat{l} \geq l+1$, according to Lemma 4, we have $z_{\hat{l}} - \frac{T_{r,\hat{l}}}{\hat{l}-k+r+1+\beta(r+1)} \leq 0$. This is a contradiction with $z_{\hat{l}} - \frac{T_{r,\hat{l}}}{\hat{l}-k+r+1+\beta(r+1)} > 0$. ■

# 1 Computation of the proximity operator

$$\min_w \frac{\beta}{2}||w-v||_F^2 + \frac{1}{2}(||w||_k^{sp})^2 \tag{4}$$

Argyriou *et al.* [8] showed that this computation of the *proximity operator* can be done in $O(nd(k + log(nd)))$ steps. Here we include the derivation for self-containedness.

Before we present the solution, we firstly give the following two lemmas. Lemma 6 indicates that the each component of the optimal solution $w^*$ has the same sign of its counterpart in $v$. Lemma 7 shows that if $|v_i|$ is the $j$th largest element of $|v|$, then $|w_i|$ is the $j$th largest element of $|w^*|$.

**Lemma 6.** *Let $w^*$ be the optimal solution to the minimization problem given in Eq. (4). Then $w_i^* v_i \geq 0$ for all $i = 1, ..., nd$.*

*Proof.* Assume that the claim does not hold. Thus, there exists $i$ for which $w_i^* v_i < 0$. Let $w$ be a vector such that $w_k = w_k^*$ for all $k \neq i$ and $w_i = 0$. It is easy to verify that (1) $||w^*||_k^{sp} \geq ||w||_k^{sp}$ and (2) $||w^* - v||^2 > ||w - v||^2$. We thus find a solution $w$ which attains an objective value smaller than that of $w^*$. This is a contradiction with $w^*$ is the optimal solution. ■

**Lemma 7.** *Let $w^*$ be the optimal solution to the minimization problem given in Eq. (4). Then for any $i, j$, if $|v_i| \geq |v_j|$, we also have $|w_i| \geq |w_j|$.*

*Proof.* Assume that the claim does not hold. Thus, there exists $i, j$ for which $|v_i| \geq |v_j|$ and $|w_i^*| < |w_j^*|$. Let $w$ be a vector such that $w_k = w_k^*$ for all $k \neq i, k \neq j$ and $w_i = sign(v_i)|w_j^*|, w_j = sign(v_j)|w_i^*|$. Therefore, $\frac{\lambda}{2}(||w^*||_k^{sp})^2 + \frac{1}{2}||w^* - v||^2 - \frac{\lambda}{2}(||w||_k^{sp})^2 - \frac{1}{2}||w - v||^2 = \frac{1}{2}((w_i^* - v_i)^2 + (w_j^* - v_j)^2 - (sign(v_i)|w_j^*| - v_i)^2 - (sign(v_j)|w_i^*| - v_j)^2) = -|w_i^*||v_i| - |w_j^*||v_j| + |w_j^*||v_i| + |w_i^*||v_j| = (|w_j^*| - |w_i^*|)(|v_i| - |v_j|) \geq 0$.

Hence, $w$ attains an objective value less than or equal to that of $w^*$. This is a contradiction. ■

Based on the lemma 6 and lemma 7, we can rewrite the optimization problem as

$$\begin{aligned} &\min_q \frac{1}{2\beta}(\sum_{i=1}^{k-r-1} q_i^2 + \frac{1}{r+1}(\sum_{i=k-r}^{nd} |q_i|)^2) + \frac{1}{2}||q-z||^2 \\ &s.t. \;\; q_1 \geq q_2 \geq ... \geq q_{nd} \geq 0 \\ &\quad q_{k-r-1} > \frac{1}{r+1}(\sum_{i=k-r}^{nd} q_i) \geq q_{k-r} \end{aligned} \quad (5)$$

where $z$ denotes the vector obtained by sorting the absolute value of $v$ in a descending order, $z_1 \geq z_2 \geq ... \geq z_{nd} \geq 0$. Let $s$ be denoted as the corresponding index, $|v_{s_i}| = z_i$. Once we obtain the optimal solution of Eq. (5), we can construct the solution of Eq. (4) by setting

$$w_{s_i} = sign(v_{s_i})q_i. \quad (6)$$

Now, we consider to solve the Eq. (5). Without the constrains, Eq.(5) can be rewrite as the following two sub problems:

$$\min_{q_1,\ldots,q_{k-r-1}} \frac{1}{2} \sum_{i=1}^{k-r-1} (q_i^2/\beta + (q_i - z_i)^2) \quad (7)$$

$$\min_{q_{k-r},\ldots,q_{nd}} \frac{1}{2\beta(r+1)}(\sum_{i=k-r}^{nd} |q_i|)^2 + \frac{1}{2}\sum_{i=k-r}^{nd} (q_i - z_i)^2 \quad (8)$$

Eq (7) is a simple problem. The optimal solution is

$$q_i = \frac{\beta}{\beta+1} z_i \;\; for\ i = 1, ..., k-r-1 \quad (9)$$

We take the derivative of Eq.(8) with respect to $q_j$ to zero, where $j = k-r, ..., nd$. we obtain

$$\frac{1}{\beta(r+1)}(\sum_{i=k-r}^{nd} |q_i|)\nabla|q_j| + (q_j - z_j) = 0 \quad (10)$$

where $\nabla|q_j|$ is the sub-gradient of $|q_j|$. Since $q_j \geq 0$, we have

$$\nabla|q_j| = \begin{cases} \{c_j \in R | 0 \leq c_j \leq 1\} & \text{if } q_j = 0 \\ 1 & \text{if } q_j > 0 \end{cases}$$

Hence, we need to discuss the two cases for finding the solution of Eq.(8). Suppose that $q_{k-r} \geq ... \geq q_l > 0$ and $q_{l+1} = ... = q_{nd} = 0$. Substitution it into

Eq. (10), we have

$$\begin{aligned} &\frac{1}{\beta(r+1)}(\sum_{i=k-r}^{l} q_i) + (q_{k-r} - z_{k-r}) = 0 \\ &\dots \\ &\frac{1}{\beta(r+1)}(\sum_{i=k-r}^{l} q_i) + (q_l - z_l) = 0 \\ &\frac{1}{\beta(r+1)}(\sum_{i=k-r}^{l} q_i) \geq z_{l+1} \end{aligned} \tag{11}$$

Hence, the optimal solution of Eq (8) is

$$q_i = \begin{cases} z_i - \frac{\sum_{i=k-r}^{l} z_i}{l-k+r+1+\beta(r+1)} & \text{if } i = k-r, ..., l \\ 0 & \text{if } i = l+1, ..., nd \end{cases} \tag{12}$$

Substitution the solution into Eq. (11), we have $l$ satisfies

$$z_l > \frac{\sum_{i=k-r}^{l} z_i}{\beta(r+1) + l - k + r + 1} \geq z_{l+1}. \tag{13}$$

Now, we consider the constrain $q_{k-r-1} > \frac{1}{r+1}(\sum_{i=k-r}^{nd} q_i)^2 \geq q_{k-r}$. Substitution the solutions of Eq. (8) and Eq. (7) into it, we have

$$\begin{aligned} &q_{k-r-1} > \frac{1}{r+1}\sum_{i=k-r}^{nd} q_i \geq q_{k-r} \\ \Rightarrow &\frac{\beta}{\beta+1} z_{k-r-1} > \beta(\frac{\sum_{i=k-r}^{l} z_i}{\beta(r+1) + l - k + r + 1}) \\ &\geq z_{k-r} - \frac{\sum_{i=k-r}^{l} z_i}{\beta(r+1) + l - k + r + 1} \\ \Rightarrow &\frac{1}{\beta+1} z_{k-r-1} > \frac{\sum_{i=k-r}^{l} z_i}{\beta(r+1) + l - k + r + 1} \geq \frac{1}{\beta+1} z_{k-r} \end{aligned} \tag{14}$$

Hence, the solution of Eq. (5) is

$$q_i = \begin{cases} \frac{\beta}{\beta+1} z_i & \text{if } i = 1, ..., k-r-1 \\ z_i - \frac{\sum_{i=k-r}^{l} z_i}{l-k+r+1+\beta(r+1)} & \text{if } i = k-r, ..., l \\ 0 & \text{if } i = l+1, ..., nd \end{cases} \tag{15}$$

where $r$ and $l$ satisfy that

$$\begin{cases} \frac{1}{\beta+1} z_{k-r-1} > \frac{\sum_{i=k-r}^{l} z_i}{\beta(r+1)+l-k+r+1} \geq \frac{1}{\beta+1} z_{k-r} \\ z_l > \frac{\sum_{i=k-r}^{l} z_i}{\beta(r+1)+l-k+r+1} \geq z_{l+1} \end{cases} \tag{16}$$