Fast Randomized Singular Value Thresholding for Nuclear Norm Minimization

Tae-Hyun Oh*
KAIST
thoh.kaist.ac.kr@gmail.com

Yasuyuki Matsushita
Microsoft Research Asia / Osaka University
yasumat@ist.osaka-u.ac.jp

Yu-Wing Tai
KAIST
yuwing@gmail.com

In So Kweon
KAIST
iskweon77@kaist.ac.kr

Abstract

Rank minimization problem can be boiled down to either Nuclear Norm Minimization (NNM) or Weighted NNM (WNNM) problem. The problems related to NNM (or WNNM) can be solved iteratively by applying a closed-form proximal operator, called Singular Value Thresholding (SVT) (or Weighted SVT), but they suffer from high computational cost to compute a Singular Value Decomposition (SVD) at each iteration. In this paper, we propose an accurate and fast approximation method for SVT, called fast randomized SVT (FRSVT), where we avoid direct computation of SVD. The key idea is to extract an approximate basis for the range of a matrix from its compressed matrix. Given the basis, we compute the partial singular values of the original matrix from a small factored matrix. While the basis approximation is the bottleneck, our method is already severalfold faster than thin SVD. By adopting a range propagation technique, we can further avoid one of the bottleneck at each iteration. Our theoretical analysis provides a stepping stone between the approximation bound of SVD and its effect to NNM via SVT. Along with the analysis, our empirical results on both quantitative and qualitative studies show our approximation rarely harms the convergence behavior of the host algorithms. We apply it and validate the efficiency of our method on various vision problems, e.g. *subspace clustering, weather artifact removal, simultaneous multi-image alignment and rectification.*

1. Introduction

Low-rank matrix recovery problems arise in many engineering and applied science problems, and rank minimization techniques have attracted tremendous interests. Rank minimization is a crucial regularizer to derive a low-rank solution, and is required in many mathematical models in computer vision and machine learning [4, 9, 13, 20, 22, 26, 29, 28, 31, 36, 38]. As rank minimization using the l_0-norm is an NP-hard problem, it is typically relaxed using the nuclear norm (*i.e.*, $\|\cdot\|_*$, the sum of all the singular values), which is the convex envelope of the rank function.

As a simple example, a nuclear norm minimization (NNM) problem is expressed as:

$$\mathbf{X}^* = \operatorname{argmin}_{\mathbf{X}} f(\mathbf{X}) + \tau\|\mathbf{X}\|_*, \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{m\times n}$, and $\tau > 0$ is a regularization parameter. The function $f(\mathbf{X})$ can be defined according to different applications, *e.g.*, $f(\mathbf{X}) = \|\mathbf{O} - \mathbf{X}\|_1$ in robust principal component analysis (RPCA) [3], $f(\mathbf{X}) = \frac{1}{2}\|\mathbf{AX} - \mathbf{B}\|_F^2$ in multivariate regression and multi-class learning [25], and $f(\mathbf{X}) = \frac{1}{2}\|\pi_\Psi(\mathbf{O}) - \pi_\Psi(\mathbf{X})\|_F^2$ in matrix completion [3]. Here, $\mathbf{O}$ is measured data, $\|\cdot\|_1$ and $\|\cdot\|_F$ denote the l_1 and the Frobenius norms, and $\pi_\Psi(\cdot)$ is an orthogonal projection operator setting $[\pi_\Psi(\mathbf{X})]_{i,j} = [\mathbf{X}]_{i,j}$ for $(i,j) \in \Psi$ and 0 otherwise. Also, weighted nuclear norm minimization (WNNM), which can be non-convex according to weights [9, 13, 27], is used to better approximate $\operatorname{rank}(\cdot)$.

Unless the loss function $f(\mathbf{X})$ is a proximity term, most previous works use first-order optimization algorithms, *e.g.*, dual method [7], accelerated proximal gradient [14], augmented Lagrange multiplier [17], alternating direction method [17, 18], and iterative reweighted least squares [23]. All these approaches, NNM, WNNM and their regularized versions, iteratively solve the simple NNM subproblem defined as the following nuclear norm plus the proximity term:

Problem (Nuclear norm minimization). *For $\tau \geq 0$ and $\mathbf{A} \in \mathbb{R}^{m\times n}$,*

$$\mathbf{X}^* = \operatorname{argmin}_{\mathbf{X}} \tau\|\mathbf{X}\|_* + \frac{1}{2}\|\mathbf{X} - \mathbf{A}\|_F^2, \tag{2}$$

where the optimal $\mathbf{X}^*$ can be obtained by the singular value thresholding operator defined as:

Definition 1 (Singular value thresholding[1] [1]). *The problem (2) has a closed form solution by the singular value*

*This work was done while the first author was an intern at Microsoft Research Asia.

[1] A similar result for WNNM can be found in [9], called WSVT.

thresholding (SVT) operator $\mathbb{S}_\tau(\cdot)$ *as*

$$\mathbf{X}^* = \mathbb{S}_\tau(\mathbf{A}) = \mathbf{U}_\mathbf{A}\mathcal{S}_\tau(\mathbf{\Sigma}_\mathbf{A})\mathbf{V}_\mathbf{A}^\top, \tag{3}$$

where $\mathcal{S}_\tau(x) = \text{sgn}(x) \cdot \max(|x| - \tau, 0)$ *is the soft shrinkage operator [10], and* $\mathbf{U}_\mathbf{A}\Sigma_\mathbf{A}\mathbf{V}_\mathbf{A}^\top$ *is the SVD of* $\mathbf{A}$.

The major computational bottleneck for NNM and WNNM problems is the necessity of solving Eq. (2) multiple times, where SVD computation occupies the largest computation cost (*e.g.*, $O(mn\ \min(m,n))$ for a SVD [8].

In this paper, we propose a fast SVT technique to accelerate general NNM and WNNM methods. Our method is motivated by the previous study of a randomized SVD proposed by Halko *et al*. [11], and we extend the original general method in several respects for better solving the NNM and WNNM problems that we focus in this paper. As a result, we propose an algorithm that we call *fast randomized SVT* (FRSVT). We present the connection between FRSVT and low-rank approximation with both theoretical and empirical analyses, and show the effectiveness of the proposed method via simulations and a few applications. Specifically, this paper makes the following contributions:

- We develop a successive truncated low-rank decomposition that can be generally applied to NNM and WNNM problems. Our method achieves high speed and rarely harms the convergence behavior of the host algorithms without loss of accuracy.
- By exploiting the proximity of the range space over iterations, our method propagates the estimated basis of the previous iteration to the next iteration for acceleration. We call the proposed technique *range propagation* (RP).
- We provide theoretical analysis between the low-rank approximation and our FRSVT method. In addition, we show the empirical stability and behavior of our method with respect to varying parameters.
- We apply FRSVT to various computer vision applications and show the performance gain in comparison with previous methods.

2. Related Works

Candès *et al*. [3] showed that, under some mild condition, the solution of NNM is equivalent to the solution of rank minimization in conjunction with sparse outlier model [3]. Inspired by the success of the convex surrogate for the rank minimization, low-rank observation is exploited by various computer vision applications, such as rain removal [4], de-noising [9], inpainting [13], motion segmentation by subspace clustering [20], structure from motion [22], background subtraction [26], tag transduction [26], high dynamic range imaging [29], batch image alignment [31], photometric stereo [36], image rectification [38], and nuclear norm regularized learning models [25].

Due to the high computational complexity of SVD in the SVT operator, a fast SVT is always necessary for both small- and large-scale problems for real-timeness and scalability, and there have been continued efforts to improve the speed of low-rank algorithms. Liu *et al*. [21] efficiently solve the NNM on a small matrix by factorizing a matrix into two small matrices. Although the work achieved significant speed-up, global optimum cannot be guaranteed, because the factorization introduces non-convexity like other factorization methods [39, 6].

With retaining the advantage of convexity, Liu *et al*. [22] exactly solve RPCA on a small sub-sampled matrix and propagate the seed solution to other parts via ℓ_1 filtering in a linear time. Since the work only focuses on RPCA, a fast SVT method is still needed as a tool for NNM to be applied to large-scale problems. Cai *et al*. [2] avoid explicit SVD computation using the dual of SVT. Since their method uses Newton iterations with an inverse matrix, the input matrix should be preprocessed by the complete orthogonal decomposition [8] (COD) to ensure a non-singular square matrix. The standard COD consists of twice of QR decompositions with column/row pivoting, which requires $O(mn\ \min(m,n))$, so the reduction of computation complexity is still limited.

In other thread of works, it has been shown that the exact SVD computation is unnecessary in the inner loop of NNM. Mu *et al*. [26] proposed a compressed optimization by random projection. Ma *et al*. [25] solved NNM related problems with a linear-time approximate SVD [5]. However, these methods are occasionally unstable, because the input matrix itself is approximated by sampling or projection, where the original information is impaired and the randomness leads to unstable incorrect results. In contrast to these methods, our method only approximates subspace bases to guarantee that most spectrum information is retained.

3. Fast Randomized Singular Value Thresholding (FRSVT)

The basic idea of our method shares the idea of Liu *et al*. [22] and Ma *et al*. [25], in that the solution of Eq. (2) can be found by applying SVT to a small matrix instead of the original large matrix. Instead of sampling columns or rows of a matrix as in [22, 25], our method extracts a small core matrix by finding orthonormal bases with the unitary invariant property. Specifically, since the NNM defined in Eq. (2) consists of unitary invariant norms, the following equality holds:

Proposition 1. *Let* $\mathbf{A} = \mathbf{QB} \in \mathbb{R}^{m\times n}$, *where* $\mathbf{Q} \in \mathbb{R}^{m\times n}$ *has orthonormal columns. Then,*

$$\mathbb{S}_\tau(\mathbf{A}) = \mathbf{Q}\,\mathbb{S}_\tau(\mathbf{B}), \tag{4}$$

where $\mathbb{S}_\tau(\cdot)$ *is the SVT operator.*

Algorithm 1 Fast Randomized Singular Value Thresholding (FRSVT) algorithm

Input : $\mathbf{A} \in \mathbb{R}^{m\times n}, \tau > 0, l = k + p > 0$ and $q \geq 0$. For range propagation, the orthonormal column matrix $\tilde{\mathbf{Q}}$ of the previous iteration.

if not Range propagation **then**
  Sample Gaussian random matrix $\mathbf{\Omega} \in \mathbb{R}^{n\times l}$
  $\mathbf{Y} = \mathbf{A}\mathbf{\Omega}$
  $\mathbf{Q} = \mathrm{QR\_CP}(\mathbf{Y})$
else
  Sample Gaussian random matrix $\mathbf{\Omega} \in \mathbb{R}^{n\times p}$
  $\mathbf{Y} = \mathbf{A}\mathbf{\Omega}$
  $\mathbf{Q}_\mathbf{Y} = \mathrm{PartialOrthogonalization}(\tilde{\mathbf{Q}}, \mathbf{Y})$
  $\mathbf{Q} = [\tilde{\mathbf{Q}}, \mathbf{Q}_\mathbf{Y}]$
end if
repeat
  $\mathbf{Q} = \mathrm{QR}(\mathbf{A}\mathbf{A}^\top\mathbf{Q})$
until η times
$[\mathbf{H}, \mathbf{C}] = \mathrm{QR}(\mathbf{A}^\top\mathbf{Q})$
$[\mathbf{W}, \mathbf{P}] = \mathrm{PolarDecomposition}(\mathbf{C})$
$[\mathbf{V}, \mathbf{D}] = \mathrm{EigenDecomposition}(\mathbf{P})$
$\mathbb{S}_\tau(\mathbf{A}) = (\mathbf{QV})\,\mathcal{S}_\tau(\mathbf{D})\,(\mathbf{HWV})^\top$

Output : $\mathbb{S}_\tau(\mathbf{A}), \tilde{\mathbf{Q}} = \mathbf{QV}$

Proof. For the derivation, refer to the supplementary material. This also holds for most of the WSVT cases. □

In general, SVT requires SVD computation, and its complexity is $O(mn^2)$[2]. Based on Proposition 1, we can avoid expensive computation by instead computing SVT on a smaller matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$, when $\mathbf{Q} \in \mathbb{R}^{m\times n}$ is available. Given $\mathbf{Q} \in \mathbb{R}^{m\times k}$ that best approximates $\mathbf{A}$ by a rank-k matrix, the complexity of SVD of $\mathbf{B} \in \mathbb{R}^{k\times n}$ becomes $O(nk^2)$. Therefore, when $k \ll n$, the computation speed can be significantly improved.

Our SVT computation iterates the following two steps: 1) Estimating an orthonormal column matrix $\mathbf{Q}$, and 2) Computing SVD of $\mathbf{B}$ for SVT. For SVT computation, a partial (or truncated) SVD is frequently used to reduce the complexity in many prior arts. Our method similarly finds a rank-k approximation ($k < n$) of the original matrix $\mathbf{A}$ as $\mathbf{A} \approx \hat{\mathbf{A}}_k = \mathbf{QB}$. It saves the computation of the first step as well as the second step, because the size of matrices $\mathbf{Q}$ and $\mathbf{B}$ are reduced to $m \times k$ and $k \times n$, respectively. By exploiting the observation that the major orthonormal k bases evolve slowly over iterations, our method efficiently initializes matrix $\mathbf{Q}$ at each iteration by bypassing expensive random range estimation (Range propagation in Sec. 3.1). In addition, by avoiding direct SVD computation, we can further reduce the computation as described in Sec. 3.2. Also, we describe a target rank prediction technique for further improving speed in Sec. 3.3.

[2]Without loss of generality, we assume $m \geq n$ in this paper. In the case $m < n$, the proposed method is analogous.

3.1. Finding Approximate Range

Inspired by Halko *et al*. [11], we first estimate the orthonormal bases $\mathbf{Q} = [\mathbf{q}_1, \cdots, \mathbf{q}_l]$ (where $l \geq k$) such that $\mathrm{span}(\mathbf{Q}) \subseteq \mathrm{Range}(\mathbf{A})$ from a matrix compressed by random projection. Intuition of their randomized range finding algorithm is as follows. By multiplying a random vector ω_j, a random linear combination $\mathbf{y}_j$ of the column vectors of $\mathbf{A}$ is generated, which encodes the partial range of $\mathbf{A}$. Suppose $\mathbf{A} = \mathbf{A}_k + \mathbf{E}$, where $\mathbf{A}_k$ is the rank-k projection of $\mathbf{A}$, of which range is the target to capture, and $\mathbf{E}$ represents small perturbation, then sample vector $\mathbf{y}_j$ can be obtained by

$$\mathbf{y}_j = \mathbf{A}_k\omega_j + \mathbf{E}\omega_j. \tag{5}$$

Even though unwanted $\mathbf{E}$ may be included in $\{\mathbf{y}_j\}$, since the action of $\mathbf{A}_k$ (*i.e*., magnitudes of spectrum) is larger than $\mathbf{E}$, the range of $\mathbf{A}_k$ is dominant to be captured in $\{\mathbf{y}_j\}$. However, if only k vectors $\{\mathbf{y}_j\}$ are sampled, $\{\mathbf{y}_j\}$ could not span the entire $\mathrm{Range}(\mathbf{A}_k)$. By increasing the sampling rate, most of $\mathrm{Range}(\mathbf{A}_k)$ can be captured; therefore, we oversample l sample vectors, so that $\mathrm{Range}(\mathbf{A}_k)$ can be captured as much as possible.

Given the sample matrix $\mathbf{Y} = [\mathbf{y}_1, \cdots \mathbf{y}_l]$, where $l = k + p$, $\mathbf{Q}$ can be obtained by orthonormalizing $\mathbf{Y}$. When $r = \mathrm{rank}(\mathbf{A}) < l$, r bases are enough to span the entire $\mathrm{Range}(\mathbf{A})$. Indeed, when $\mathrm{rank}(\mathbf{A}) < l$, the range finding algorithm is fairly accurate and close to the exact method as we will see the theoretical analysis in Sec. 4. Thus, we can reduce the dimension of $\mathbf{Q}$ for almost free by estimating the rank and dominant bases by the QR decomposition with Column Pivoting (QR-CP) to $\mathbf{Y}$. While Halko *et al*. also proposed a complex algorithm to adaptively and approximately determine the number of bases by different randomization for sampling, orthogonalization, and re-orthogonalization, our method is based on an exact method with the same complexity using QR-CP, and we adaptively predict the sampling rate in a simpler manner (we will see in Sec. 3.3). Fortunately, in LAPACK, an efficient QR-CP routine using level-3 BLAS (`dgeqp3`) is available. Moreover, our method does not require the upper triangle matrix from QR, but only the orthonormal basis $\mathbf{Q}$; therefore, we can avoid extracting the whole triangular matrix but only compute rank and $\mathbf{Q}$ with `dgeqp3` and `orgqr` routines, respectively. After obtaining $\mathbf{Q} \in \mathbb{R}^{m\times s}$, where $s = \min(l, r)$, we can compute a small matrix $\mathbf{B}$ by $\mathbf{B} = \mathbf{Q}^\top\mathbf{A} \in \mathbb{R}^{s\times n}$.

Range Propagation (RP) for Fast Range Finding Based on the observation that $\mathrm{Range}(\mathbf{A}_{(i)})$ at the i-th iteration of NNM related problems is similar to the one at the $(i-1)$-th iteration, our method uses the singular vectors at the $(i-1)$-th step as an initial approximation of range bases $\mathbf{Q}_{(i)}$ at the i-th step. To capture the change of the range space more, additional p sample vectors $\{\mathbf{y}\}$ are

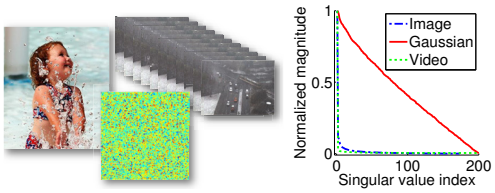


Figure 1. Illustration of singular value decaying. [Left] Gaussian random, video and image samples. [Right] Decaying graphs of singular values. The *Red*, *Green*, *Blue* lines represent the graphs of a Gaussian random matrix, video, and image, respectively. The spectrum of visual data decays significantly fast.

newly sampled. We append $\{\mathbf{y}\}$ to the previous singular vector matrix $\tilde{\mathbf{Q}}_{(i-1)}$ as $\mathbf{Q}_{(i)} = [\tilde{\mathbf{Q}}_{(i-1)}, \mathbf{y}_1, \cdots, \mathbf{y}_p]$, and apply partial orthogonalization only for newly added $\{\mathbf{y}\}$ by the modified Gram-Schmidt procedure [8]. The number of bases can subsequently be reduced by checking the rank with QR-CP in the first step of the power iteration.

Power Iteration Among the overall process in our algorithm, since the only approximation step is to estimate the orthonormal column matrix $\mathbf{Q}$ only, the accuracy of our algorithm depends only on this step. In Eq. (5), if the magnitude of the action (*i.e.*, spectrum) of $\mathbf{A}_k$ is not dominant against $\mathbf{E}$, the directions of sample vectors are biased and may be included in $\mathrm{Range}(\mathbf{A}_k)^\perp$. This introduces accuracy degradation to the rest of the process. To resolve this issue, Halko *et al*. [11] proposed the power iteration scheme, which makes the spectrum difference between $\mathbf{A}_k$ and $\mathbf{E}$ larger by estimating $\mathbf{Q}$ on $(\mathbf{A}\mathbf{A}^\top)^\eta\mathbf{A}$. It improves the chance of better capturing the range of $\mathbf{A}_k$ from $\mathbf{Y} = (\mathbf{A}\mathbf{A}^\top)^\eta\mathbf{A}\mathbf{\Omega}$, while the singular vectors remain unchanged. Halko *et al*. also showed that $\eta = 2$ or 4 power iterations are sufficient for usual data of interest, and highly accurate range finding can be achieved. As shown in Fig. 1, decaying singular values of visual data is much faster than Gaussian random matrix. Our empirical tests also show that $\eta = 2$ is enough and it is used in all our experiments.

3.2. Computing the Singular Values (Vectors)

The NNM problem is now reduced to SVT on a smaller matrix $\mathbf{B}$. In this section, we further reduce the computation time of SVT on $\mathbf{B}$. The SVT operator can be computed by SVD and shrinkage on its singular values. For positive semi-definite matrices, SVD can be more efficiently computed by Eigen decomposition (ED), which is generally faster than SVD at least twice in our empirical tests. To apply ED to a general matrix, we form a positive semi-definite matrix by the following decomposition:

Definition 2 (**Polar decomposition** [12])**.** *Let* $\mathbf{X} \in \mathbb{C}^{m\times n}$, $m \geq n$. *There exists a matrix* $\mathbf{W} \in \mathbb{C}^{m\times n}$ *and a unique Hermitian positive semi-definite matrix* $\mathbf{P} \in \mathbb{C}^{n\times n}$ *such that*

$$\mathbf{X} = \mathbf{W}\mathbf{P}, \qquad \mathbf{W}^*\mathbf{W} = \mathbf{I},$$

where $\mathbf{I}$ *is the identity matrix. If* $\mathrm{rank}(\mathbf{X}) = n$*, then* $\mathbf{P}$ *is positive definite and* $\mathbf{W}$ *is uniquely determined.*

Note that the existence of polar decomposition is equivalent to the existence of SVD.

We use a Newton based polar decomposition suggested by Higham *et al*. [12], which has a quadratic convergence behavior. In our experiment, it converges at a small number of iterations (typically 7) with various different data, which is consistent with the result of [2, 12]. Due to the requirement of the inverse operator in Newton iterations, it is only applicable to non-singular square matrix. Since $\mathbf{B}^\top \in \mathbb{R}^{n\times s}$ is a full column rank matrix, the non-singular square matrix can be simply obtained from $\mathbf{B}^\top = \mathbf{H}\mathbf{C}$ by QR decomposition, where we call $\mathbf{C} \in \mathbb{R}^{s\times s}$ a core matrix that is always non-singular and square. Contrary to Sec. 3.1, no column pivoting is required.

We sequentially apply the polar decomposition and ED on the core matrix to be $\mathbf{C} = \mathbf{W}\mathbf{P} = \mathbf{W}\mathbf{V}\mathbf{D}\mathbf{V}^\top$, where $\mathbf{D}$ and $\mathbf{V}$ are the eigenvalue and eigenvector matrices of $\mathbf{P}$, respectively. Since the matrices $\mathbf{H}$, $\mathbf{W}$, and $\mathbf{V}$ are orthonormal column matrices, the diagonal matrix $\mathbf{D}$ is equivalent to the singular value matrix of $\mathbf{B}$. Finally, $\mathbb{S}_\tau(\mathbf{A})$ can be approximated by

$$\mathbb{S}_\tau(\mathbf{A}) \approx \mathbb{S}_\tau(\hat{\mathbf{A}}_s) = (\mathbf{Q}\mathbf{V})\,\mathcal{S}_\tau(\mathbf{D})\,(\mathbf{H}\mathbf{W}\mathbf{V})^\top. \quad (6)$$

For the range propagation, the singular vector matrix $\tilde{\mathbf{Q}}$ is stored as $\tilde{\mathbf{Q}} = \mathbf{Q}\mathbf{V}$ or $\mathbf{H}\mathbf{W}\mathbf{V}$ (according to either side of random matrix multiplication). Overall algorithm is summarized in Algorithm 1.

3.3. Adaptive Rank Prediction (AP) Heuristic

For SVT, only singular vectors corresponding to the singular values that are greater than a certain threshold are needed, and full SVD is unnecessary. Since the rank of $\mathbf{A}_{(i)}$ is unknown before SVD, predicting its rank can avoid unnecessary computation. We observe that, in many NNM related problems, the rank of $\mathbf{A}_{(i)}$ is monotonically increasing or decreasing over iterations, and the rank is stabilized as the number of iteration increases. As we shall see in the theorem of error bound in Sec. 4, over-sampling is always useful to reduce the expected error bound of FRSVT. Thus, optimistically predicting rank allows to achieve both computational efficiency and stability.

The speed advantage of our method will be lost with a higher sampling rate. In such a case, we resort to the truncated SVT by upper bounding the target rank. As shown in natural image statistics of Fig. 1, the rank of $\mathbf{A}_{(i)}$ is generally stabilized at low-rank in many computer vision application. Usually, the final accuracy is not harmed, as seen

in the successes of the truncated SVD in the NNM related problems [17, 18, 19, 21].

Based on these observations, we define over-sampling rate p as:

$$p_{i+1} = \begin{cases} a, & \text{if } r_i < l_i, \\ \lceil \rho n \rceil, & \text{otherwise,} \end{cases} \tag{7}$$

where $m \geq n$ is assumed, a is a constant (set to $a = 2$), $\rho \in (0, 1]$ is a constant parameter to rapidly follow the real rank r_i (set to $\rho = 0.05$), and $\lceil \cdot \rceil$ denotes the `ceil` operation. The prediction rule for sampling rate is defined as $l_{i+1} = \min(r_i + p_{i+1}, b)$, where $b = \lceil \gamma n \rceil$ is the maximum bound of sampling rate, $\gamma \in (0, 1]$ is the proportion parameter. The sampling rate l_i can be regarded as the predicted rank at the i-th iteration, and r_i is the number of singular values of $\mathbf{A}_{(i)}$ that are larger than the threshold, *i.e.*, the estimated rank of $\mathbb{S}_\tau(\mathbf{A}_{(i)})$. Initially, we set $l_0 = 0.1b$. In the case of the range propagation, the number of columns in $\tilde{\mathbf{Q}}_{(i-1)}$ becomes r_i, and $p_i = l_i - r_i$.

When $r_i < l_i$, Eq. (7) slightly over-samples, otherwise it optimistically predicts the rank of the next iteration by the a larger over-sampling rate. By virtue of low-rankness of visual data shown in Fig. 1, the optimistic rule leads to accurate estimate.

3.4. Computational Complexity

The computation of the sample matrix $\mathbf{Y}$ by random projection requires $O(mnl)$, but we can reduce it to $O(mn)$ by the range propagation technique. The random matrix can be multiplied to either left or right hand side of $\mathbf{A}$. We multiply the random matrix to the longer side (left side, when $n < m$) of $\mathbf{A}$, so that the complexity of power iteration can be further reduced. Under $m \geq n$, the power scheme consists of $O(mns)$ matrix multiplication and $O(ns^2)$ QR. The polar decomposition and ED takes $O(s^3)$. Finally, the solution matrix composition takes $O(mns)$. Most expensive computations with $O(mns)$ are simple matrix multiplications, so it can be easily parallelized.

4. Approximation Bound

To analyze the behavior of the proposed algorithm, it is meaningful to see the approximation error bound. Except for the randomized step in sampling $\mathbf{Y}$, the other steps of our algorithm is based on exact methods. Thus, the accuracy is only affected by the randomized range estimation, which computes a rank-k approximation of a matrix. Since there is the only one approximation step, the proposed method shares the same theorems with Halko *et al.* [11] as:

Theorem 1 (Average Frobenius[3] error bounds by randomization [11]). *Let $\hat{\mathbf{A}}_k$ be a rank-k approximation matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$. For a target rank $k \geq 2$ and an over-sampling parameter $p \geq 2$, where $k + p \leq \min(m, n)$, draw an standard Gaussian matrix $\mathbf{\Omega} \in \mathbb{R}^{n \times (k+p)}$. Then, the theoretical minimum error and the upper bound satisfy*

$$\sum\nolimits_{j>k} \sigma_j^2(\mathbf{A}) \leq \mathbb{E}\|\mathbf{A} - \hat{\mathbf{A}}_k\|_F^2 \leq \mathrm{poly}(\mathbf{v}) \cdot \sum\nolimits_{j>k} \sigma_j^2(\mathbf{A}),$$

where $\mathbb{E}$ denotes expectation with respect to the random matrix, $\mathrm{poly}(\mathbf{v})$ is a function for $\mathbf{v} = \{k, p\}$ (without the power iteration) or $\{k, p, \eta\}$ (with the power iteration).

Halko *et al.* show that the upper bound of the error is close to theoretical minimum error with high probability in conjunction with some improving techniques (*e.g.*, over-sampling, power iteration), and the bound is pessimistic.

From Theorem 1, the following theorem is our main result for the bound of the approximate SVT.

Theorem 2 (Average error bound of the approximate SVT). *Let $\mathbb{S}_\tau(\cdot)$ be the SVT operator. Then, the average error satisfies the following inequality.*

$$\mathbb{E}\|\mathbb{S}_\tau(\mathbf{A}) - \mathbb{S}_\tau(\hat{\mathbf{A}}_k)\|_F^2 \leq \mathrm{poly}(\mathbf{v}) \cdot \Big(\sum\nolimits_{j>k} \sigma_j^2(\mathbf{A})\Big) - G(\mathbf{A}),$$

where $G(\mathbf{A}) = \sum_{j>k} \min(\sigma_j(\mathbf{A}), \tau)^2 \geq 0$.

Proof. The proof can be found in the appendix. □

In Theorems 1 and 2, in the case without power iteration, $\mathrm{poly}(\cdot)$ is defined as $\mathrm{poly}(k, p) = (1 + k/(p-1))$. The polynomial bound with power iteration with the Frobenius norm representation has no simple form, so instead we can observe the behavior of the error bound with power iteration by referring to the spectral bound of Halko *et al.*

Consequently, our Theorem 2 asserts that the bound of the approximate SVT is tighter than the error by rank-k approximation in Theorem 1 on average. Thus, we can see that FRSVT can be safely used in practice. From Theorem 2, we see the following properties:

- $\sigma_{k+1}(\mathbf{A}) \to \epsilon$ and $\|\mathbb{S}_\tau(\mathbf{A}) - \mathbb{S}_\tau(\hat{\mathbf{A}}_k)\|_F^2$ may approach to close to zero when $k \geq \mathrm{rank}(\mathbf{A})$, which means it is almost exact.
- Fortunately, the bound becomes rapidly tighten as p or η increase in both practice and theory, similar to Halko *et al.*.
- The bound is independent of the size of the matrix.

5. Experimental Results

In this section, we first evaluate the efficiency of the proposed method in comparison with other methods using simulation data. The evaluation is conducted by examining the performance of a single SVT computation and also by assessing the performance of RPCA computation, which is arguably the most relevant NNM problem in computer vision today. We then show NNM applications in computer

[3]For easy derivation, we provide the squared versions (the square of the Frobenius norm) for Theorems 1 and 2. The original error bound (the Frobenius norm) can be obtained by taking the square root.

vision using real-world data: subspace clustering, semi-online weather artifact removal, and simultaneous multi-image alignment and rectification. All the experiments are conducted on a PC with Intel i7-3.4GHz and 16GB RAM. The same shared parameters were used among algorithms.

5.1. Evaluation using Simulation Data

We quantitatively evaluate our method in comparison to other methods with synthetic matrices sampled from a standard Gaussian distribution. We evaluate the computational times on Matlab 2010a and 2014a 64bits. Since the recent Matlab has been intensively optimized on Intel CPU (mainly due to improvement of Intel MKL), the computation efficiency has been noticeably improved. Therefore, it is worth reporting performance difference on these two versions of Matlab, because most related works have been assessed with Matlab older than 2011a [2, 19, 21, 22, 26]. For a fair comparison, we turn off multi-threading functions including `maxNumCompThreads(1)` in Matlab. We describe the implementation details of each method in the supplementary material. Also, more complete testings and comparisons can be found in the supplementary material.

Single SVT test Figure 2 compares speed and accuracy of SVT computation using SVT methods, such as Matlab built-in SVD[4] (baseline), Lanczos [16], FSVT [2], LTSVD [5] and our FRSVT (with / without RP). Except for the baseline SVD, the others produce the truncated SVD of the input matrix, and it is used for SVT computation. For a rank-k approximation, we compute rank-$(k+p)$ approximations by setting the over-sampling rate p to 2 for Lanczos, 5 for LTSVD, 5 for FRSVT. While LTSVD in Fig. 2-(e) is faster than ours, the approximation error is significantly higher than other truncated SVD, and it results in slower convergence in RPCA as we will see in the following.

Robust PCA test To see the convergence behavior of the SVT methods for RPCA [3], in Table 1, we compare our method to various SVT methods, such as SVD, LTSVD, BLWS [19], FSVT, RSVD [11], using an inexact augmented Lagrange multiplier method [17] (iALM, or called alternating directional multiplier method)[5]. We apply the proposed adaptive rank prediction in Sec. 3.3 to LTSVD, RSVD and our FRSVT. LTSVD shows convergence degradation to achieve comparable accuracy with others due to rough approximation on both bases and singular values. Other methods including our FRSVT show similar numbers of iterations and accuracy, but our method has a considerably lower computation time for a single iteration.

[4]We apply either the economic size or full SVD according to the matrix shape and report faster one, but we denote them as econSVD only.

[5]We report the result of Mu *et al*. [26] in the supplementary material as we could not reproduce their result.

Algorithms	#NM	SIT	TT	SPG	ERR
$\text{iALM}_{\text{econSVD10}}$	23	947.9	21668	–	1.8e-7
$\text{iALM}_{\text{econSVD14}}$	23	46.5	1069	20×	1.8e-7
$\text{iALM}_{\text{LTSVD}}$ [5]	41	1.7	70	312×	4.6e-7
$\text{iALM}_{\text{BLWS}}$ [19]	24	2.2	53	405×	4.8e-7
$\text{iALM}_{\text{FSVT}}$ [2]	23	110.0	2527	9×	1.8e-7
$\text{iALM}_{\text{RSVD}}$ [11]	23	3.6	81	268×	1.8e-7
$\text{iALM}_{\text{FRSVT}}$ (ours)	23	1.5	33	665×	1.8e-7
$\text{iALM}_{\text{FRSVT-RP}}$ (ours)	23	1.3	30	716×	1.8e-7

- #NM: The number of NNM, - SIT: Elapsed time (sec) of a single iteration,
- TT: Total elapsed time (sec), - SPG: Speed-up gain against the baseline,
- ERR: $\|\mathbf{A}_{GT} - \hat{\mathbf{A}}\|_F / \|\mathbf{A}_{GT}\|_F$.

Table 1. Quantitative Comparisons on RPCA. 4000×4000 matrices are used. The results of other sizes and convergence graphs are shown in the supplementary material. In iALM procedure, #NM is the number of iteration to solve the NNM subproblem, which corresponds to the total number of iterations.

Type	Objective function	Constraint
A	$\underset{\mathbf{L},\mathbf{S}}{\text{argmin}} \|\mathbf{L}\|_* + \lambda\|\mathbf{S}\|_{2,1}$	$\mathbf{O} = \mathbf{ZL} + \mathbf{S}$
B	$\underset{\mathbf{L},\mathbf{S}}{\text{argmin}} \sum_{i=k+1}^{n} \sigma_i(\mathbf{L}) + \lambda\|\mathbf{S}\|_1$	$\mathbf{O} = \mathbf{L} + \mathbf{S}$
C	$\underset{\mathcal{L},\mathcal{E},\Gamma}{\text{argmin}} \sum_{i=1}^{3} \alpha_i \|\mathcal{L}_{(i)}\|_* + \lambda\|\mathcal{E}\|_1$	$\mathcal{O} \circ \Gamma = \mathcal{L} + \mathcal{E}$

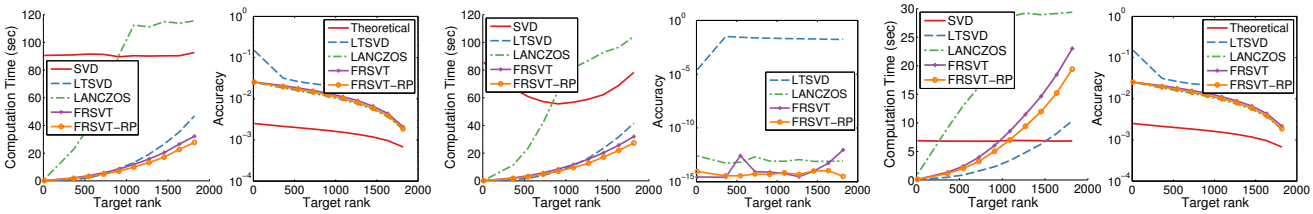
Table 2. Examples of NNM related objective functions. (a) Low-rank representation. (b) RPCA based on the non-convex truncated nuclear norm, where k is the target rank to be encouraged. (c) Low-rank and sparse 3-order tensor decomposition with alignment. Here, $\|\cdot\|_{2,1}$ is $l_{2,1}$ norm, $\{\alpha_i\}$ are the balance parameters among the unfolding matrices $\mathcal{L}_{(i)}$ and $\sum \alpha_i = 1$ is assumed. We refer the basic tensor algebra notations denoted in [37].

5.2. More Applications

We show more general applications of our FRSVT method to various types of low-rank optimization problems summarized in Table 2, such as affine constrained NNM (Type A), non-convex truncated NNM (Type B) and NNM on tensor structure (Type C). We show them with typical computer vision applications in the following.

Type A - Robust Subspace Clustering by Low-Rank Representation (LRR) Many visual data is often characterized by a mixture of multiple subspaces. One of the recent promising methods is LRR, which effectively performs subspace clustering and noise correction simultaneously. While both noise correction and subspace clustering are known to be challenging, LRR has been shown robust against large corruptions. The robust LRR can be formulated by Type A, which can be efficiently solved by iALM. In this experiment, we use the same parameter settings and evaluation metrics suggested in [20].

We apply FRSVT to the robust LRR for motion segmentation on Hopkins155 [35] and for face recognition on a part of Extended Yale Database B [15] respectively. We use all 156 sequences in Hopkins155, and only the first 10 classes in Yale, in which each class contains 64 face images captured under various illumination conditions – given 42×48 resized images, we construct a 2016×640 data matrix by



(a) Computation time Full rank 2000 × 2000 Matlab 2010a (b) Accuracy Full rank 2000 × 2000 (c) Computation time Low rank 2000 × 2000 Matlab 2010a (d) Accuracy Low rank 2000 × 2000 (e) Computation time Full rank 2000 × 2000 Matlab 2014a (f) Accuracy Full rank 2000 × 2000

Figure 2. SVT comparisons across SVD methods. [Left] X-axis: Target rank, Y-axis: Elapsed time (Sec). [Right] X-axis: Target rank, Y-axis: $\|\mathbf{A}^* - \hat{\mathbf{A}}_k\|_F / \|\mathbf{A}^*\|_F$, where $\mathbf{A}^* = \mathbb{S}_\tau(\mathbf{D}_{GT})$, $\hat{\mathbf{A}}_k = \mathbb{S}_\tau(\mathbf{D}_k)$, $\mathbf{D}_{GT}$ is the input data, and $\mathbf{D}_k$ is approximated by each method. For the low-rank matrix test, we generate input data matrices of which rank correspond to the target rank by multiplying two Gaussian random matrices with $m \times r$ and $r \times n$, while in other tests full rank matrices are used. In (b,d,f), the theoretical minimum error bound by the rank truncation in $\mathbb{S}_\tau(\mathbf{D}_k)$ is provided for guidance, and it is defined as $\sum_{j>k} \sigma_i(\mathbf{A}^*)$, where $\sigma_i(\mathbf{A}^*)$ is computed by the built-in SVD of Matlab with the assumption that the SVD is exact. For the low-rank case in (d), the theoretical minimum error is 0, so we omit the theoretical bound. For other results with other sizes, refer to the supplementary material.

	Computational Time (s)		
	LRR+SVD Matlab 2010a	LRR+SVD Matlab 2014a	LRR+Ours
Time per Motion	1.230	1.016	0.452
Time for 640 Faces	419.440	75.216	44.590

Table 3. Comparisons of the subspace segmentation algorithms on Hopkins155 [35] (Motion) and Extended Yale Database B [15] (Face). Motion Segmentation Errors on the Hopkins155 of both LRR and LRR+Ours are 1.59%. The segmentation accuracies (%) on the Yale data are 79.06 for both LRR and LRR+Ours. These results show the improvement of the computation time with retaining the same accuracy. Details on the computational time are shown in the supplementary material.

vectorizing each image. While Hopkins155 is only weakly corrupted, Yale data is heavily corrupted by shadows, specularity and noise. As shown in Table 3, our method speeds up LRR without degrading of the accuracy.

Type B - Semi-Online Weather Artifact Removal We consider a weather artifacts (*e.g.* snow or rain) removal problem in a video sequence. Since the weather artifacts have non-deterministic appearance and sparse yet random distribution in the spatio-temporal domain, we model the artifacts as sparse outlier and the scene as low-rank, while we want to retain moving objects. We apply the low-rank and sparsity decomposition to n frames in a sliding window manner to leave moving objects in the latent images.

Unfortunately, since finding low-rank solution by the nuclear norm is based on the blessing of high dimensionality [3], it could be degenerated with small n frames as reported in Oh *et al*. [30]. With the assumption that the object motions are small during few frames, we can encourage the low-rank solution to be rank-1 and decompose sparse corruptions by Type B optimization with $k = 1$, where $\mathbf{O}$ is constructed by stacking vectorized n images. It can be effectively solved by alternating Partial SVT (PSVT) [30] and l_1 minimization based on iALM. We replace PSVT by

Figure 3. Qualitative results for the weather artifact removal. (a) Sample input images $\mathbf{O}$. (b) Low-rank images $\mathbf{Z}$.

FRSVT with the partial thresholding. We transfer the previous basis estimation to the next n frame optimization, so it can be regarded as semi-online algorithm.

Our algorithm produces n results simultaneously for n images. With an $n = 5$ sliding window, the method based on SVD takes $158.9ms$ per a single channel of a 384×288 image on Matlab 2014a ($318.3ms$ on Matlab 2010a), while our method only takes $65.52ms$. The qualitative results of our method can be found in Fig. 3, which shows plausible snow removal effects.

Type C - Simultaneous Multi-Image Alignment and Rectification Simultaneous multi-image alignment and rectification problem is Type C optimization suggested by Zhang *et al*. [37] (called SRALT). The method combines the ideas of TILT [38] and RASL [37] on the 3rd-order tensor structure, which exploits low-rank texture property and nuclear norm-based misalignment error respectively. With showing the applicability of FRSVT to NNM on a 3rd-order

(a) (b) (c) (d)

Figure 4. Qualitative comparison with RASL [31] and SRALT [37] + Our FRSVT. (a) Samples of input images. (b) Results obtained by RASL [31]. (c) Results obtained by SRALT [37] + Our FRSVT. (d) The average images. (Top: for all input images, Middle: for RASL, Bottom: for SRALT+Ours). Due to the angular bias shown in (a), RASL aligns the samples to be consistent at the biased angle, while the SRALT based method in (c) not only align the images, but also correct upright poses.

tensor, we present a new application of SRALT to gait data.

Since gait is a biometric signal spanned on not only spatial domain (2D), but also temporal domain, so it is natural to represent the data by 3[rd]-order tensor [24]. In gait recognition, the exact alignment of acquired data is frequently assumed, but in real situation, the accuracy of the gait recognition could be varied according to the pivot angle of the camera and locations of moving persons of both training and test data. Therefore, SRALT framework is useful to align many gait data and to find vertical angles as a preprocessing step. We observe that the human silhouettes have the low-rank texture property, and as observed in Lu *et al*. [24], cyclic gait motions are spanned by a few number of bases.

We conduct the experiments on the Gait Challenge data sets[6] [34], whose size is $\mathcal{O} \in \mathbb{R}^{32\times 22\times 3000}$ (3000 unaligned images with the size 22×32). The set of the geometric transformations $\Gamma = \{g_1, \cdots, g_{3000}\}$, where $g_j \in \mathbb{R}^p$ is a p-group parameterization, and is parameterized by a 3-DoF Euclidean transformation in our experiments. We randomly apply translation and rotation to introduce misalignments. Our method is implemented on the top of [37] by replacing their SVT method with FRSVT. While the method of [37] is converged at the objective value 206.69 and takes about 6 hours 51 min. ($24652s$), our method takes only about 1 hour 45 min. ($6280s$) and reaches the even smaller objective score 206.46. The qualitative and the detailed computation time comparisons are reported in Figs. 4 and 5, respectively.

6. Conclusions

We have presented a fast approximate SVT method that exploits the property of iterative NNM procedures by range propagation and adaptive rank prediction. The approximation error bound shows our method can produce reliable approximation. The proposed method has been assessed using the problems of affine constrained NNM, non-convex NNM and NNM on tensor structure as well as the original NNM. The empirical evaluations showed the consistent result with the theoretical analysis, and our approach can reduce the computational time of low-rank applications without losing accuracy and hurting convergence behavior. The major bottleneck of our method is in the power scheme, and we are interested in further reducing the computational complexity by effectively relaxing this computation block in the future.

[6]The dataset is described in the supplementary material.

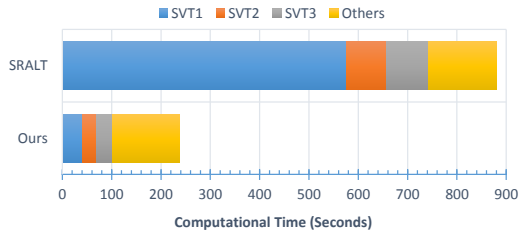


Figure 5. Computational time comparison on the registration and rectification application [37]. The NNM for tensor consists of 3-way SVTs. Computational times are measured for a single outer iteration (almost 100 inner iterations).

Appendices – Proof of Theorem 2

The following propositions are useful for deriving the approximation bound.

Proposition 2 (Dual pseudo-contraction). *Let $\mathbb{P}_\tau(\cdot)$ is a dual operator of $\mathbb{S}_\tau(\cdot)$, such that $\mathbf{X} = \mathbb{S}_\tau(\mathbf{X}) + \mathbb{P}_\tau(\mathbf{X})$. Then,*

$$\|\mathbb{S}_\tau(\mathbf{A}) - \mathbb{S}_\tau(\mathbf{B})\|_F^2 \leq \|\mathbf{A} - \mathbf{B}\|_F^2 - \|\mathbb{P}_\tau(\mathbf{A}) - \mathbb{P}_\tau(\mathbf{B})\|_F^2.$$

Proof. Since the derivation is straightforward based on the pseudo-contraction [32], we omit the details. □

Proposition 3. *Let $\hat{\mathbf{A}}_k$ be a rank-k approximation of $\mathbf{A}$.*

$$\min \|\mathbb{P}_\tau(\mathbf{A}) - \mathbb{P}_\tau(\hat{\mathbf{A}}_k)\|_F^2 = \sum\nolimits_{j>k} \min(\sigma_j(\mathbf{A}), \tau)^2.$$

Proof. This can be directly reached from Proposition 2. □

Proof of Theorem 2. By Propositions 2 and 3,

$$\begin{aligned}
&\|\mathbb{S}_\tau(\mathbf{A}) - \mathbb{S}_\tau(\hat{\mathbf{A}}_k)\|_F^2 \\
&\leq \|\mathbf{A} - \hat{\mathbf{A}}_k\|_F^2 - \|\mathbb{P}_\tau(\mathbf{A}) - \mathbb{P}_\tau(\hat{\mathbf{A}}_\mathbf{k})\|_F^2 \\
&\leq \|\mathbf{A} - \hat{\mathbf{A}}_k\|_F^2 - \sum\nolimits_{j>k} \min(\sigma_j(\mathbf{A}), \tau)^2.
\end{aligned}$$

Take the expectation to the both sides,

$$\begin{aligned}
&\mathbb{E}\|\mathbb{S}_\tau(\mathbf{A}) - \mathbb{S}_\tau(\hat{\mathbf{A}}_k)\|_F^2 \\
&\leq \mathbb{E}\|\mathbf{A} - \hat{\mathbf{A}}_k\|_F^2 - \sum\nolimits_{j>k} \min(\sigma_j(\mathbf{A}), \tau)^2 \\
&\leq \mathrm{poly}(\mathbf{v}) \cdot \sum\nolimits_{j>k} \sigma_j^2(\mathbf{A}) - \sum\nolimits_{j>k} \min(\sigma_j(\mathbf{A}), \tau)^2, \\
&\quad \text{(by Theorem 1)} \\
&= \mathrm{poly}(\mathbf{v}) \cdot \sum\nolimits_{j>k} \sigma_j^2(\mathbf{A}) - G(\mathbf{A}) \\
&\quad \left(G(\mathbf{A}) = \sum\nolimits_{j>k} \min(\sigma_j(\mathbf{A}), \tau)^2\right)
\end{aligned}$$

Obviously, $G(\mathbf{A}) \geq 0$ by the definition. □

***Acknowledgments** We would like thank Zhouchen Lin for helpful comments, Xiaoqin Zhang and Yadong Mu for sharing their source codes. The first author was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2010-0028680).

References

[1] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 2010. 1

[2] J.-F. Cai and S. Osher. Fast singular value thresholding without singular value decomposition. *Methods and Applications of Analysis*, 2013. 2, 4, 6

[3] E. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 2011. 1, 2, 6, 7

[4] Y.-L. Chen and C.-T. Hsu. A generalized low-rank appearance model for spatio-temporally correlated rain streaks. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2013. 1, 2

[5] P. Drineas, R. Kannan, and M. W. Hahoney. Fast monte carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. *SIAM Journal of Computing*, 2006. 2, 6

[6] A. Eriksson and A. van den Hengel. Efficient computation of robust weighted low-rank matrix approximations using the l_1 norm. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012. 2

[7] A. Ganesh, Z. Lin, J. Wright, L. Wu, M. Chen, and Y. Ma. Fast algorithms for recovering a corrupted low-rank matrix. In *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, 2009. 1

[8] G. H. Golub and C. F. V. Loan. Matrix computations 3rd Ed. *Johns Hopkins University Press*, 1996. 2, 4

[9] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with application to image denoising. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2014. 1, 2

[10] E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for l_1-minimization: Methodology and convergence. *SIAM Journal on Optimization*, 2008. 2, 1

[11] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 2011. 2, 3, 4, 5, 6

[12] N. J. Higham. Computing the polar decomposition-with applications. *SIAM Journal on Scientific and Statistical Computing*, 1986. 4

[13] Y. Hu, D. Zhang, J. Ye, X. Li, and X. He. Fast and accurate matrix completion via truncated nuclear norm regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013. 1, 2

[14] S. Ji and J. Ye. An accelerated gradient method for trace norm minimization. In *ICML*. ACM, 2009. 1

[15] K.-C. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2005. 6, 7, 2

[16] Z. Lin. Some software packages for partial svd computation. In *arXiv:1108.1548*, 2011. 6, 2

[17] Z. Lin, M. Chen, and Y. Ma. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. Technical Report UILU-ENG-09-2215, UIUC, 2009. 1, 5, 6, 3

[18] Z. Lin, R. Liu, and Z. Su. Linearized alternating direction method with adaptive penalty for low-rank representation. In *Advances in Neural Information Processing Systems (NIPS)*, 2011. 1, 5

[19] Z. Lin and S. Wei. A block lanczos with warm start technique for accelerating nuclear norm minimization algorithms. In *arXiv:1012.0365*, 2010. 5, 6, 2

[20] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2013. 1, 2, 6

[21] G. Liu and S. Yan. Active subspace: Toward scalable low-rank learning. *Neural Computation*, 2012. 2, 5, 6

[22] R. Liu, Z. Lin, Z. Su, and J. Gao. Linear time principal component pursuit and its extensions using l_1 filtering. *Neurocomputing*, 2014. 1, 2, 6

[23] C. Lu, J. Tang, S. Y. Yan, and Z. Lin. Generalized nonconvex nonsmooth low-rank minimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2014. 1

[24] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. MPCA: Multilinear principal component analysis of tensor objects. *IEEE Transactions on Neural Networks,*, 2008. 8

[25] S. Ma, D. Goldfarb, and L. Chen. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 2011. 1, 2

[26] Y. Mu, J. Dong, X. Yuan, and S. Yan. Accelerated low-rank visual recovery by random projection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2011. 1, 2, 6, 3

[27] T.-H. Oh, H. Kim, Y.-W. Tai, J.-C. Bazin, and I. S. Kweon. Partial sum minimization of singular values in rpca for low-level vision. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2013. 1, 6

[28] T.-H. Oh, J.-Y. Lee, and I. S. Kweon. High dynamic range imaging by a rank-1 constraint. In *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2013. 1

[29] T.-H. Oh, J.-Y. Lee, Y.-W. Tai, and I. S. Kweon. Robust high dynamic range imaging by rank minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2014. 1, 2

[30] T.-H. Oh, Y.-W. Tai, J.-C. Bazin, H. Kim, and I. S. Kweon. Partial sum minimization of singular values in robust pca: Algorithm and applications. *arXiv preprint arXiv:1503.01444*, 2015. 7

[31] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2012. 1, 2, 8, 5

[32] G. Pierra. Decomposition through formalization in a product space. *Mathematical Programming*, 1984. 8

[33] C. Sanderson. Armadillo: An open source c++ linear algebra library for fast prototyping and computationally intensive experiments. *Technical Report, NICTA*, 2010. 2

[34] S. Sarkar, P. J. Phillips, Z. Liu, I. R. Vega, P. Grother, and K. W. Bowyer. The humanid gait challenge problem: Data sets, performance, and analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2005. 8

[35] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2007. 6, 7

[36] L. Wu, A. Ganesh, B. Shi, Y. Matsushita, Y. Wang, and Y. Ma. Robust photometric stereo via low-rank matrix completion and recovery. In *Asian Conference on Computer Vision (ACCV)*, 2010. 1, 2

[37] X. Zhang, D. Wang, Z. Zhou, and Y. Ma. Simultaneous rectification and alignment via robust recovery of low-rank tensors. In *Advances in Neural Information Processing Systems (NIPS)*, 2013. 6, 7, 8, 5

[38] Z. Zhang, A. Ganesh, X. Liang, and Y. Ma. TILT: Transform invariant low-rank textures. *International Journal of Computer Vision (IJCV)*, 2012. 1, 2, 7

[39] Y. Zheng, G. Liu, S. Sugimoto, S. Yan, and M. Okutomi. Practical low-rank matrix approximation under robust l_1-norm. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012. 2