Efficient Robust Principal Component Analysis via Block Krylov Iteration and CUR Decomposition

Shun Fang[1,2] Zhengqin Xu[3] Shiqian Wu[1,2,4*] Shoulie Xie[5]
[1]School of Information Science and Engineering, Wuhan University of Science and Technology, China
[2]Institute of Robotics and Intelligent Systems, Wuhan University of Science and Technology, China
[3]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China
[4]Hubei Province Key Laboratory of Intelligent Information Processing and Real-Time Industrial Systems
[5]Signal Processing, RF & Optical Dept. Institute for Infocomm Research A*STAR, Singapore
fangshun97@wust.edu.cn, fate311@sjtu.edu.cn, shiqian.wu@wust.edu.cn, slxie@i2r.a-star.edu.sg

Abstract

Robust principal component analysis (RPCA) is widely studied in computer vision. Recently an adaptive rank estimate based RPCA has achieved top performance in low-level vision tasks without the prior rank, but both the rank estimate and RPCA optimization algorithm involve singular value decomposition, which requires extremely huge computational resource for large-scale matrices. To address these issues, an efficient RPCA (eRPCA) algorithm is proposed based on block Krylov iteration and CUR decomposition in this paper. Specifically, the Krylov iteration method is employed to approximate the eigenvalue decomposition in the rank estimation, which requires $O(ndrq + n(rq)^2)$ for an $(n \times d)$ input matrix, in which q is a parameter with a small value, r is the target rank. Based on the estimated rank, CUR decomposition is adopted to replace SVD in updating low-rank matrix component, whose complexity reduces from $O(rnd)$ to $O(r^2n)$ per iteration. Experimental results verify the efficiency and effectiveness of the proposed eRPCA over the state-of-the-art methods in various low-level vision applications.

1. Introduction

Robust principal component analysis (RPCA) aims to recover a low-rank matrix $\boldsymbol{L}$ and a sparse matrix $\boldsymbol{S}$ from the corrupted observation matrix $\boldsymbol{D} \in \mathbb{R}^{n\times d}$: $\boldsymbol{D} = \boldsymbol{L} + \boldsymbol{S}$. The RPCA can be formulated as the following optimization problem [1]:

$$\min_{\boldsymbol{L},\boldsymbol{S}} \operatorname{rank}(\boldsymbol{L}) + \lambda\|\boldsymbol{S}\|_0 \quad \text{s.t. } \boldsymbol{D} = \boldsymbol{L} + \boldsymbol{S} \tag{1}$$

where ℓ_0-norm is the number of nonzero elements in the matrix, the paramter $\lambda > 0$ provides the trade-off between the rankness and sparsity.

RPCA has been widely studied and applied in computer vision. For example, "background" in a video clip captured by a static camera has a low-rank property, which can be considered in background modeling [2]. The requirement of detecting sparse outliers from the observed imagery data leads to RPCA applications in image or video processing [3]. In industry, RPCA is also applicable to point cloud filtering [4], surface defects detection [5], shock sensing [6], etc.

It is noted that the optimization problem (1) is NP-hard. The convex relaxation of RPCA has been studied to achieve an exact recovery in [1, 7–12], where the RPCA problem was relaxed as the sum of the nuclear norm and ℓ_1-norm. But the convex methods always have a rate of sublinear convergence and high computation in practice [13]. It is necessary to exploit the structure of the underlying data and develop more efficient algorithms for RPCA. Zhou et al. [14] extended the equality constraint of RPCA to inequality in order to deal with noisy data, thus the RPCA is reformulated as the following constrained non-convex optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{L},\boldsymbol{S}} \|\boldsymbol{D} - \boldsymbol{L} - \boldsymbol{S}\|_F^2 \\ &\text{s.t. } \operatorname{rank}(\boldsymbol{L}) \le r \text{ and } \|\boldsymbol{S}\|_0 \le s \end{aligned} \tag{2}$$

with the target rank r and target sparse number s.

The key idea of Eq. (2) is to formulate a constrained non-convex optimization problem. Among the nonconvex optimization mthods of RPCA, Zhou et al. [15] substituted the hard thresholding of Eq. (2) with a soft ℓ_1 regularization to reduce the complexity of $\boldsymbol{S}$. Netrapalli et al. [16] added the deterministic sparsity assumption and devoted themselves

*Shiqian Wu is corresponding author.

to recovering a low-rank matrix from the sparse corruptions that were of unknown value and support. The RPCA algorithm via gradient descent on the factorized space in [17] achieved linear convergence with proper initialization and step size, whose sparse estimator is to guarantee that the fraction of nonzero entries in each column and row of $\boldsymbol{S}$ is bounded above. Inspired by these methods, GoDec+ [18] made the model become an ordinary low-rank projection problem based on the correntropy of noise and half-quadratic optimization theory. In [19], RPCA problem is considered for the first time without heuristics, such as loss functions, convex and surrogate constraints, which provides a new direction for potential research on online algorithms. In addition, Ornhag et al. [20] used second-order methods to convert the original objectives to differentiable equivalents, benefitting from faster convergence.

Meanwhile, the efficient algorithms for RPCA have been widely investigated for large-scale matrices. In [16], the developed algorithm involved alternating projections between a set of low-rank matrices and a set of sparse matrices, whose projection idea was also widely studied. Inspired by [16], a proximal block coordinate descent method was proposed in [21] to find an ϵ-stationary solution in $O(1/\epsilon^2)$ iterations. Furthermore, an accelerated alternating projection strategy was studied in [22] for RPCA, which project a matrix onto a specific low-dimensional subspace before obtaining a new estimate of the low-rank matrix via truncated SVD.

Most of RPCA and its variants contain SVD, which requires significant computational cost for large-scale matrices. Hintermüller [23] directly considered a least-squares problem subject to rank and cardinality constraints based on matrix manifolds, which favorably avoids singular value decompositions in full dimension. Phan et al. [24] proposed an accelerated algorithm with iteratively reweighted nuclear norm to ensure that every limit point is a critical point, which leads to small singular values and obtains fast result. Some different computing strategies were employed to substitute SVD so that the computation of RPCA was significantly reduced. Cai et al. [25] introduced CUR decomposition at each iteration, which only required $O(r^2 n)$ flops per iteration and preserved more information than SVD.

Generally, the aforementioned algorithms achieved efficient and effective performance in different computer vision tasks. It is highlighted that these methods require the rank of a low-rank matrix to be known a prior, which is inappropriate in most practical applications. To bridge this gap, Xu et al. [26] proposed a rank estimation method based on Gerschgorin disk theorem (GDE), whose computational complexity is $O((d-1)^3)$ at each iteration. Furthermore, an adaptive weighting RPCA was developed based on iteratively estimated rank, which outperforms the state-of-the-art RPCA methods in various computer vision applications. More specifically, the adaptive weighting RPCA in [26] was formulated as the following optimization problem:

$$\min_{\boldsymbol{L},\boldsymbol{S}} \|\boldsymbol{L}\|_W + \lambda\|\boldsymbol{S}\|_1 \quad \text{s.t.}\, \boldsymbol{D} = \boldsymbol{L} + \boldsymbol{S} \tag{3}$$

where $\|\boldsymbol{L}\|_W = \Sigma_i \omega_i \sigma_i(\boldsymbol{L})$ and ω_i are non-negative weights. Based on the estimated rank of $\boldsymbol{L}$, the weights can be updated iteratively.

Although the adaptive weighting RPCA [26] achieves top performance in recovering a low-rank matrix and a sparse matrix, both the rank estimation and RPCA optimization algorithm contain SVD, which takes significant computational costs for large-scale matrices. Recently, randomized block Krylov iteration [27] was introduced to approximate the singular value decomposition in fewer iterations with better accuracy guarantees. This motivates us to use the block Krylov iteration method to accelerate the GDE-based rank estimation in this work. The computational complexity of the proposed Krylov GDE (KGDE)-based rank estimation is reduced to $O(ndrq + n(rq)^2)$, where q is a parameter with a small value. On the other hand, CUR decomposition is also adopted to replace SVD in the updates of a low-rank matrix, thus the computational complexity of RPCA is significantly reduced. Since the rank of a matrix is required to be known for CUR decomposition, it is natural that the proposed KGDE is used to adaptively estimate the rank of the low-rank matrix within the iterative RPCA computing. Furthermore, a new non-convex low-rank regularized term is used to replace the weighted nuclear norm in (3) which can improve the low-rank matrix approximation. Compared with the state-of-the-art RPCA approaches, the proposed efficient RPCA (eRPCA) algorithm are fast with better accuracy guarantees. The main contributions of this paper are as follows:

1) An efficient rank estimation method based on Gerschgorin disks with block Krylov iteration is proposed to accelerate the rank estimate of low-rank matrices.
2) CUR decomposition is adopted to reduce the computation of SVD on a large-scale matrix in the update of the low-rank matrix.
3) An efficient non-convex RPCA method with a non-convex weighted regularizer is proposed to achieve better recovery of the low-rank matrix and the sparse matrix.
4) The proposed eRPCA algorithm has been applied to various computer vision scenarios and outperforms the state-of-the-art methods on large-scale data.

In the following section, the improved eRPCA method, which consists of efficient rank estimation and new weight is presented in Section 2. The experimental results are demonstrated in Section 3, and the conclusions are drawn in Section 4.

2. Proposed Method

2.1. Efficient Rank Estimation via Block Krylov Iteration

In [26], GDE was used to estimate the rank of a low-rank matrix, which involves SVD and has the computational complexity $O((d-1)^3)$ for a matrix $\boldsymbol{D} \in \mathbb{R}^{n\times d}$. Hence it is time-consuming to estimate the rank of large-scale matrix $\boldsymbol{D}$. Recently it has been shown that block Krylov iteration (BKI) method [27] gives nearly optimal approximation of singular values and singular vectors with few interations. In this subsection, an efficient rank estimation is proposed by combining BKI and GDE, which is named as KGDE.

Now we give a brief review on the block Krylov iteration method. For a matrix $\boldsymbol{D} \in \mathbb{R}^{n\times d}$, the Krylov subspace is obtained by

$$\boldsymbol{K} := [\boldsymbol{v}, \boldsymbol{D}\boldsymbol{v}, \boldsymbol{D}^2\boldsymbol{v}, \cdots, \boldsymbol{D}^{q-1}\boldsymbol{v}] \tag{4}$$

where $\boldsymbol{v}$ is a vector of unit norm, $\|\boldsymbol{v}\| = 1$, and q is a scalar. In [28], the Krylov subspace was extended to randomized block Krylov subspace:

$$\boldsymbol{K} := [\boldsymbol{V}, \boldsymbol{D}\boldsymbol{V}, \boldsymbol{D}^2\boldsymbol{V}, \cdots, \boldsymbol{D}^{q-1}\boldsymbol{V}] \tag{5}$$

where $\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_b] \in \mathbb{R}^{d\times b}$ is a block of b random vectors.

After constructing the block Krylov subspace, the Block Krylov Iteration (BKI) [27] can be used to generate closely approximate eigenpairs within fewer iterations by projecting the matrix onto the randomized block Krylov subspace. Its idea is to take a randomized starting matrix $\boldsymbol{V} = \boldsymbol{D}\boldsymbol{\Omega}$ as the initial matrix, where $\boldsymbol{\Omega}$ is a Gaussian random matrix. BKI captures an accurate range space:

$$\boldsymbol{K} := (\boldsymbol{D}\boldsymbol{D}^T)^q \boldsymbol{D}\boldsymbol{\Omega} \tag{6}$$

where $q := \Theta(\frac{\log d}{\sqrt{\varepsilon}})$. Orthonormalize the columns of $\boldsymbol{K}$ to obtain $\boldsymbol{Q}$ then compute $\boldsymbol{M} := \boldsymbol{Q}^T\boldsymbol{D}\boldsymbol{D}^T\boldsymbol{Q}$. Set $\overline{\boldsymbol{U}}_k$ to the top k singular vectors of $\boldsymbol{M}$. At last, return the low-dimension matrix $\boldsymbol{Z}$:

$$\boldsymbol{Z} = \boldsymbol{Q}\overline{\boldsymbol{U}}_k \tag{7}$$

The following theorem shows that the randomized BKI can get strong relative-error bound.

Theorem 1 *[27] Given data matrix $\boldsymbol{D} \in \mathbb{R}^{n\times d}$, $n < d$ with eigenvalues $\lambda_i, i = 1, 2, \cdots, n$. Let $\{\theta_i, \boldsymbol{\varphi}_i\}_{i=1}^k$ be the k eigenpair computed using q steps of Block Krylov Iteration (using the orthonormal basis of $\boldsymbol{K}$ for $\boldsymbol{V} \in \mathbb{R}^{d\times b}$). If $q = \frac{\log d}{\sqrt{\varepsilon}}$ for some $0 < \varepsilon < 1$, then have*

$$|\theta_i - \lambda_i| \le \varepsilon\lambda_{k+1}, i = 1, \cdots, k.$$

In the rest of this subsection, we present a fast rank estimation of low-rank matrices based on the BKI and GDE. Given a low-rank matrix $\boldsymbol{D} \in \mathbb{R}^{n\times d}, n < d$ with rank r. The BKI algorithm outputs a low-dimension matrix $\boldsymbol{Z} \in \mathbb{R}^{n\times k}$, $k < d$, which aligns well with the top k sigular vectors of $\boldsymbol{D}$. If $r \le k$, the ranks of $\boldsymbol{D}$ and $\boldsymbol{Z}$ are equal. Hence it reduces the computational complexity when GDE can also be applied to $\boldsymbol{Z}$ to estimate the rank instead of the large-scale matrix $\boldsymbol{D}$.

The covariance matrix $\boldsymbol{R}_{\boldsymbol{Z}} \in \mathbb{R}^{k\times k}$ of the matrix $\boldsymbol{Z}$ with a rank r can be defined as:

$$\boldsymbol{R}_{\boldsymbol{Z}} = \boldsymbol{Z}^T\boldsymbol{Z} \tag{8}$$

Then the eigenvalue decomposition of $\boldsymbol{R}_{\boldsymbol{Z}}$ is given by

$$\boldsymbol{R}_{\boldsymbol{Z}} = \boldsymbol{U}_{\boldsymbol{R}_{\boldsymbol{Z}}}\boldsymbol{\Sigma}_{\boldsymbol{R}_{\boldsymbol{Z}}}\boldsymbol{U}_{\boldsymbol{R}_{\boldsymbol{Z}}}^H \tag{9}$$

where $\boldsymbol{U}_{\boldsymbol{R}_{\boldsymbol{Z}}} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_k]$ is the eigenvector matrix, and $\boldsymbol{\Sigma}_{\boldsymbol{R}_{\boldsymbol{Z}}} = diag(\sigma_1, \sigma_2, \cdots, \sigma_k)$ represents the eigenvalue matrix. According to the Gerschgorin disk theorem [29], $\boldsymbol{R}_{\boldsymbol{Z}}$ can be transformed as follows:

$$\boldsymbol{R}_{\boldsymbol{Z}} = \begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1k} \\ R_{21} & R_{22} & \cdots & R_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ R_{k1} & R_{k2} & \cdots & R_{kk} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{\boldsymbol{Z1}} & \boldsymbol{R} \\ \boldsymbol{R}^H & R_{kk} \end{bmatrix} \tag{10}$$

where $\boldsymbol{R}_{\boldsymbol{Z1}} \in \mathbb{R}^{(k-1)\times(k-1)}$ is the square matrix by the first $(k-1)$ rows and the first $(k-1)$ columns of $\boldsymbol{R}_{\boldsymbol{Z1}}$, and $\boldsymbol{R} = [R_{1k}, R_{2k}, \cdots, R_{(k-1)k}]^T$. Then the eigenvalue decomposition on $\boldsymbol{R}_{\boldsymbol{Z1}}$ is obtained as:

$$\boldsymbol{R}_{\boldsymbol{Z1}} = \boldsymbol{U}_{\boldsymbol{Z1}}\boldsymbol{\Sigma}_{\boldsymbol{1}}\boldsymbol{U}_{\boldsymbol{Z1}}^H \tag{11}$$

where $\boldsymbol{U}_{\boldsymbol{Z1}} = [\boldsymbol{q}'_1, \boldsymbol{q}'_2, \cdots, \boldsymbol{q}'_{k-1}]$ is an $(k-1)\times(k-1)$ unitary matrix composed of the eigenvectors of $\boldsymbol{R}_{\boldsymbol{Z1}}$, and $\boldsymbol{\Sigma}_{\boldsymbol{1}} = diag(\sigma'_1, \sigma'_2, \cdots, \sigma'_{k-1})$. Similar to Eq. (10), a unitary transformed matrix $\boldsymbol{U} \in \mathbb{R}^{k\times k}(\boldsymbol{U}\boldsymbol{U}^H = \boldsymbol{I})$ is defined as:

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{U}_{\boldsymbol{Z1}} & \boldsymbol{0} \\ \boldsymbol{0}^T & \boldsymbol{1} \end{bmatrix} \tag{12}$$

The transformed covariance matrix is obtained by:

$$\begin{aligned} \boldsymbol{R}_{\boldsymbol{T}} = \boldsymbol{U}^H\boldsymbol{R}_{\boldsymbol{Z}}\boldsymbol{U} &= \begin{bmatrix} \boldsymbol{U}_{\boldsymbol{Z1}}^H\boldsymbol{R}_{\boldsymbol{Z1}}\boldsymbol{U}_{\boldsymbol{Z1}} & \boldsymbol{U}_{\boldsymbol{Z1}}^H\boldsymbol{R} \\ \boldsymbol{R}^H\boldsymbol{U}_{\boldsymbol{Z1}} & R_{kk} \end{bmatrix} \\ &= \begin{bmatrix} \sigma'_1 & 0 & 0 & \cdots & 0 & \rho_1 \\ 0 & \sigma'_2 & 0 & \cdots & 0 & \rho_2 \\ 0 & 0 & \sigma'_3 & \cdots & 0 & \rho_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma'_{k-1} & \rho_{k-1} \\ \rho_1^* & \rho_2^* & \rho_3^* & \cdots & \rho_{k-1}^* & R_{kk} \end{bmatrix} \end{aligned} \tag{13}$$

where $\rho_i = \boldsymbol{q}_i'^H\boldsymbol{R}$. The eigenvalues of $\boldsymbol{R}_{\boldsymbol{T}}$ can be estimated using the Gerschgorin disk theorem [29]. Then, the radii of the first $(k-1)$ Gerschgorin's disk can be written as:

$$r_i = |\rho_i| = |\boldsymbol{q}_i'^H\boldsymbol{R}| \tag{14}$$

where the radius r_i of the ith Gerschgorin's disk depends on the size of $\boldsymbol{q}_i'^H\boldsymbol{R}$.

To further improve the accuracy of the rank estimation, the radii of the Gerschgorin's disk are shrank in [26]. Then

Algorithm 1 KGDE

Input: $\boldsymbol{D} \in \mathbb{R}^{n\times d}$, error $\varepsilon \in (0,1)$, $k=2$, the $(1+\varepsilon)$ relative-error bound for BKI $q = \frac{\log d}{\sqrt{\varepsilon}}$

Output: Estimated rank r

```
Set gde = zeros(d, 1)
for t = 1 to d do
    Z ∈ R^{n×k} ← BKI(D ∈ R^{n×d}, ε, q)
    Covariance matrix R_Z = Z^T Z
    Obtain the transformed unitary matrix according to Eqs. (10)-(12)
    Calculate the radii r_i of the Gerschgorin's disk according to Eqs. (14)-(15)
    Calculate gde(t) according to Eqs. (16)-(17)
    if (t > 1 && gde(t) < 0) then
        break;
    else
        k = k + 1
    end if
end for
r = t − 1
return r
```

a new diagonal matrix $\boldsymbol{\Sigma_T} = diag(\sigma_1', \sigma_2', \cdots, \sigma_{k-1}', \sigma_k')$ is constructed, where $\sigma_k' = \sqrt{\sum_{i=1}^{k-1} \sigma_i'^2}$. Thus, the new transformed matrix $\boldsymbol{R_{T\Sigma_T}}$ is constructed as follows:

$$\boldsymbol{R_{T\Sigma_T}} = \boldsymbol{\Sigma_T R_T \Sigma_T^{-1}} = \begin{bmatrix} \sigma_1' & 0 & \cdots & 0 & \frac{\sigma_1'}{\sigma_k'}\rho_1 \\ 0 & \sigma_2' & \cdots & 0 & \frac{\sigma_2'}{\sigma_k'}\rho_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \sigma_{k-1}' & \frac{\sigma_{k-1}'}{\sigma_k'}\rho_{k-1} \\ \frac{\sigma_1'}{\sigma_k'}\rho_1^* & \frac{\sigma_2'}{\sigma_k'}\rho_2^* & \cdots & \frac{\sigma_{k-1}'}{\sigma_k'}\rho_{k-1}^* & R_{kk} \end{bmatrix} \tag{15}$$

$\boldsymbol{R_{T\Sigma_T}}$ and $\boldsymbol{R_T}$ are similar matrices and have the same eigenvalues. In Eq. (15), the radii are compressed to various degrees. Finally, the estimated rank is obtained by the improved heuristic decision rule:

$$gde(t) = \frac{1}{\sqrt{\sum_{i=1}^{k-1}\sigma_i'^2}}[|\sigma_t'|r_t - \frac{H_D^{(t)}(M)}{k-1}\sum_{i=1}^{k-1}|\sigma_i'|r_i] \tag{16}$$

where $t = 1, 2, \cdots, k-2$, and $0 < H_D^{(t)}(M) < 1$ is the adjustment factor:

$$H_D^{(t)}(M) = \frac{2|\sigma_{t+1}'|}{\sqrt{\sum_{i=t}^{k-1}\sigma_i'^2}} \tag{17}$$

As a result, the rank $r = t-1$ if the first negative value of (16) is reached at t.

When $r \le k$, $\boldsymbol{D}$ is transformed to $\boldsymbol{Z}$ by BKI, then the rank of $\boldsymbol{Z}$ is estimated by GDE. It is noted that while $r > k$, GDE can not estimate the true rank of $\boldsymbol{D}$ from $\boldsymbol{Z}$. To adaptively determine an appropriate k, KGDE is proposed by adopting iteration, and GDE is served as a stopping criterion for such Krylov subspace approximation of $\boldsymbol{Z}$. The KGDE for rank estimation based on BKI and GDE is summarized in Algorithm 1.

Remark 1: The computational cost of KGDE is related to BKI. If the estimated r of the input matrix $\boldsymbol{D} \in \mathbb{R}^{n\times d}$ is exact, the computational complexity will be $O(ndrq + n(rq)^2)$, where $n < d$. For sparse matrix, the computational cost of KGDE will be $O(nnz(\boldsymbol{D})rq + n(rq)^2)$, where $nnz(\boldsymbol{D})$ is the number of nonzeros in $\boldsymbol{D}$. Since $r \le n$ and $q = \frac{\log d}{\sqrt{\varepsilon}}$ are small, the KGDE will be inexpensive, especially for sparse matrices.

Remark 2: The accuracy of the KGDE mainly depends on the BKI. In light of Theorem 1, $\boldsymbol{Z}$ is obtained within a high-quality principal components approximation of $\boldsymbol{D}$, both its eigenvalues and eigenvectors are highly close to the actual ones of $\boldsymbol{D}$.

2.2. Efficient RPCA based on KGDE and CUR Decomposition

In this subsection, an efficient RPCA based on the above rank estimation and CUR decomposition is proposed for solving the following optimization problem:

$$\min_{\boldsymbol{L},\boldsymbol{S}} \|\boldsymbol{D}-\boldsymbol{L}-\boldsymbol{S}\|_F^2 + \lambda\|\boldsymbol{L}\|_\gamma \quad s.t.\ \|\boldsymbol{S}\|_0 \le s \tag{18}$$

where

$$\|\boldsymbol{L}\|_\gamma = \begin{cases} \sum_{i=1} \frac{e^\gamma \sigma_i}{\gamma+\sigma_i}, & i \le r \\ 0, & i > r \end{cases}$$

and $\sigma_i, i = 1, 2, \cdots, r$, is the singular value, $\gamma > 0$, and λ is the regularized parameter which provides a trade-off between the recovery of low-rank matrix and sparse matrix.

The optimization model (18) adds a new non-convex low-rank regularization term to Eq. (2), which adopts the estimated rank to achieve better recovery accuracy of the low-rank matrix than the weighted nuclear norm in (3).

In this work, the alternating projection method is used to solve the following two sub-problems until convergence:

$$\boldsymbol{L}_{\ell+1} = \arg\min_{\boldsymbol{L}} \|\boldsymbol{D}-\boldsymbol{L}-\boldsymbol{S}_\ell\|_F^2 + \lambda\|\boldsymbol{L}\|_\gamma \tag{19}$$

$$\boldsymbol{S}_{\ell+1} = \arg\min_{\boldsymbol{S}:\ \|\boldsymbol{S}\|_0 \le s} \|\boldsymbol{D}-\boldsymbol{L}_{\ell+1}-\boldsymbol{S}\|_F^2 \tag{20}$$

Update for L: The traditional solution of (19) often involves SVD, which requires significant computational resource for a large-scale matrix. To efficiently solve the model (19), we replace SVD with CUR decomposition [30], which has been proven to be efficient and effective in solving classical RPCA problem [31]. Mathematically, the CUR

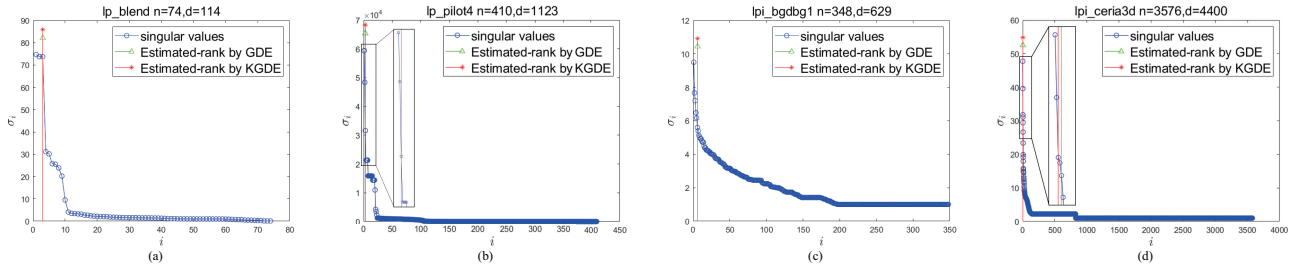

Figure 1. The numerical rank estimation of sparse data matrices.

decomposition of (19) can be rewritten as follows:

$$\boldsymbol{C}_{\ell+1} = \arg\min_{\boldsymbol{L}_{:,\mathcal{J}}} \|[\boldsymbol{D}-\boldsymbol{L}-\boldsymbol{S}_\ell]_{:,\mathcal{J}}\|_F^2 + \lambda\|\boldsymbol{L}_{:,\mathcal{J}}\|_\gamma \quad (21)$$

$$\boldsymbol{U}_{\ell+1} = \mathcal{TS}([\boldsymbol{D}-\boldsymbol{L}-\boldsymbol{S}_\ell]_{\mathcal{I},\mathcal{J}}) \quad (22)$$

$$\boldsymbol{R}_{\ell+1} = \arg\min_{\boldsymbol{L}_{\mathcal{I},:}} \|[\boldsymbol{D}-\boldsymbol{L}-\boldsymbol{S}_\ell]_{\mathcal{I},:}\|_F^2 + \lambda\|\boldsymbol{L}_{\mathcal{I},:}\|_\gamma \quad (23)$$

where $\boldsymbol{C}_{\ell+1} \in \mathbb{R}^{n\times\mathcal{I}}$ is $\mathcal{I}$ columns of (19) via uniform sampling, $\boldsymbol{R}_{\ell+1} \in \mathbb{R}^{\mathcal{J}\times d}$ is $\mathcal{J}$ rows of (19) in the same way, $\boldsymbol{U}_{\ell+1} \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ is the overlap of the column and row indices of $\boldsymbol{C}_{\ell+1}$ and $\boldsymbol{R}_{\ell+1}$. $\mathcal{TS}(\cdot)$ denotes the truncated SVD.

In addition, the rank value r is needed for $\mathcal{TS}(\cdot)$ and CUR decomposition, which can be exactly estimated by our proposed KGDE method.

As Eqs. (21) and (23) are the combination of concave and convex functions, the Difference of Convex programming [32] is used to solve these problems. Thus, $\boldsymbol{C}_{\ell+1}$ can be obtained as follows [33]:

$$\boldsymbol{C}_{\ell+1} = [\boldsymbol{a}diag\{\sigma^*\}\boldsymbol{b}^T]_{:,\mathcal{J}} \quad (24)$$

where $\boldsymbol{a}$ and $\boldsymbol{b}$ are the left and right singular vectors of $(\boldsymbol{D}-\boldsymbol{S}_{\ell+1})$, respectively, and σ^* has a closed-form solution at each iteration:

$$\sigma^{i+1} = \max((\sigma_{\boldsymbol{D}-\boldsymbol{S}_{\ell+1}} - \frac{\lambda\omega_i}{\mu^\ell}), 0) \quad (25)$$

where $\sigma_{\boldsymbol{D}-\boldsymbol{S}_{\ell+1}}$ is the singular value of $(\boldsymbol{D}-\boldsymbol{S}_{\ell+1})$, $\mu > 0$, and $\omega_i = \begin{cases} \frac{\gamma e^\gamma}{(\gamma+\sigma_i(\gamma))^2}, & i \le r \\ 0, & others \end{cases}$ is the gradient of $\|\boldsymbol{L}\|_\gamma$ at σ^i. Obviously, ω_i is dependent on r, which can be estimated by KGDE at each iteration.

Similarly,

$$\boldsymbol{R}_{\ell+1} = [\boldsymbol{a}diag\{\sigma^*\}\boldsymbol{b}^T]_{\mathcal{I},:} \quad (26)$$

Hence, the updated for $\boldsymbol{L}$ is:

$$\boldsymbol{L}_{\ell+1} = \boldsymbol{C}_{\ell+1}\boldsymbol{U}_{\ell+1}^\dagger\boldsymbol{R}_{\ell+1} \quad (27)$$

where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudoinverse.

Update for S: Similarly, the optimization of (20) can be divided into as:

$$[\boldsymbol{S}_{\ell+1}]_{:,\mathcal{J}} = \arg\min_{\boldsymbol{S}_{:,\mathcal{J}}:\ \|\boldsymbol{S}\|_0\le s} \|[\boldsymbol{D}-\boldsymbol{L}_\ell-\boldsymbol{S}]_{:,\mathcal{J}}\|_F^2 \quad (28)$$

$$[\boldsymbol{S}_{\ell+1}]_{\mathcal{I},:} = \arg\min_{\boldsymbol{S}_{\mathcal{I},:}:\ \|\boldsymbol{S}\|_0\le s} \|[\boldsymbol{D}-\boldsymbol{L}_\ell-\boldsymbol{S}]_{\mathcal{I},:}\|_F^2 \quad (29)$$

Algorithm 2 eRPCA based on KGDE

Input: $\boldsymbol{D} \in \mathbb{R}^{n\times d}$ with estimated rank r; δ: error; λ,μ: penalty parameter;ζ_0: initial thresholding value; $|\mathcal{I}| = O(r)$,$|\mathcal{J}| = O(r)$: sampling number of rows and columns; γ: non-convex regularizer parameter.

Output: $\boldsymbol{L}$,$\boldsymbol{S}$

Uniformly sample row indices $\mathcal{I}$ and column indices $\mathcal{J}$.
$\boldsymbol{L}_0 = 0$, $\boldsymbol{S}_0 = 0$, $k = 0$
while not converged **do**
  Resample $\mathcal{I}$ and $\mathcal{J}$
  Update $\zeta_{k+1} = \eta\zeta_0$
  Update $\boldsymbol{S}_{\ell+1}$ according to Eqs. (28)-(29)
  Estimate and update r by KGDE
  Update $\boldsymbol{C}_{\ell+1}$ according to (21)
  Update $\boldsymbol{R}_{\ell+1}$ according to (23)
  Update $\boldsymbol{U}_{\ell+1}$ according to (22)
  $\boldsymbol{L}_{\ell+1} = \boldsymbol{C}_{\ell+1}\boldsymbol{U}_{\ell+1}^\dagger\boldsymbol{R}_{\ell+1}$
  $\ell = \ell + 1$
end while

For $\boldsymbol{S}$, the hard thresholding operator $\mathcal{HT}$ is adopted as:

$$[\mathcal{HT}(\boldsymbol{S})]_{i,j} = \begin{cases} \boldsymbol{S}_{i,j}, & if\ |\boldsymbol{S}_{i,j}| > \zeta, \\ 0, & otherwise. \end{cases} \quad (30)$$

where ζ is the thresholding value. The hard thresholding is employed for projections on to $(\boldsymbol{D}-\boldsymbol{L}_\ell)$.

The iteration is terminated when

$$\frac{\|[\boldsymbol{D}-\boldsymbol{L}_\ell-\boldsymbol{S}_\ell]_{:,\mathcal{J}}\|_F + \|[\boldsymbol{D}-\boldsymbol{L}_\ell-\boldsymbol{S}_\ell]_{\mathcal{I},:}\|_F}{\|\boldsymbol{D}_{:,\mathcal{J}}\|_F + \|\boldsymbol{D}_{\mathcal{I},:}\|_F} < \delta \quad (31)$$

The entire procedure to solve the problem (18) is summarized in Algorithm 2.

Remark 3: The computational complexity of $\boldsymbol{U}$ in CUR decomposition is $O(r^3)$. Since the sampling numbers of rows and columns are $|\mathcal{I}| = O(r)$ and $|\mathcal{J}| = O(r)$, respectively, updating $\boldsymbol{L}$ requires $O(r^2n)$ flops per iteration. In contrast, computing the SVD usually requires $O(rnd)$. In addition, the runtime of updating $\boldsymbol{S}$ is actually a simple projection, whose computational complexity is further less than that of $\boldsymbol{L}$. Thus, the computational complexity of the alternating projection procedure is $O(\frac{d^2r}{\epsilon} + d\cdot poly(r, log(r/\epsilon), 1/\epsilon))$, where $0 < \epsilon < 1$ is an

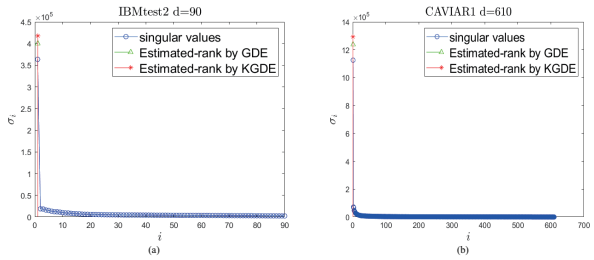

Figure 2. The numerical rank estimation of video data matrices.

accuracy parameter.

3. Experimental Results

This section reports the experimental results of our proposed eRPCA and compares it with state-of-the-art algorithms on synthetic datasets and two practical computer vision scenarios. The parameters in our method are listed as follows: $\gamma = 0.02$, $\delta = 10^{-7}$, $\lambda = 10^{-3}$, $\mu = \frac{1}{2}$, $\zeta_0 = \max(|\boldsymbol{D}|)$, and $\varepsilon = 0.05$. The parameters in other methods use their default settings. All the experiments are conducted on a laptop with MATLAB 2019 equipped with Windows 10 based on AMD Ryzen 5 CPU with 32G RAM.

3.1. The Validation of Rank Estimation Method

3.1.1 Public real datasets

To validate the efficiency and accuracy of KGDE, the general data matrices are tested to estimate the rank. SuiteSparse Database[1] [34] provides general data matrices with low numerical rank, whose Gaussian-type distribution assumptions for the data and noise may not hold. In addition, the video datasets from Scene Background Initialization (SBI) dataset[2] [35], which are converted to the full matrix are also tested.

Fig.1 shows the numerical rank estimation for the SVD curve of sparse data matrices. The singular values of Fig.1(a) and Fig.1(b) have apparent distribution, and KGDE and GDE obtain the exact rank estimation. The singular values of Fig.1(c) and Fig.1(d) are very close, it is therefore hard to bind the rank. The result of Fig.1(c) is the same, and the one of Fig.1(d) is slightly different, while the gap can be ignored in practice.

Fig.2 shows the rank estimation results of video data matrices. The video datasets have a stable low-rank, and the rank is usually equal to 1. Thus, the distribution of their singular values has a noticeable gap.

The comparisons of runtime and estimated rank for different matrices are shown in Table 1. When the matrix scale is small, the execution time is close, but KGDE is still faster than GDE. When the matrix becomes more extensive, the efficiency of KGDE performs better than GDE. Because the

[1]https://sparse.tamu.edu/

[2]https://sbmi2015.na.icar.cnr.it/SBIdataset.html

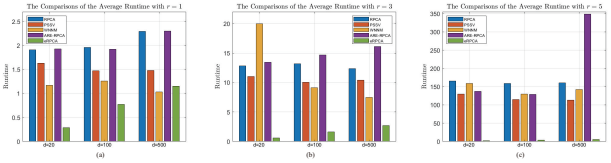

Figure 3. The comparison of the average runtime between different algorithms.

Table 1. The results of runtime and estimated rank using GDE and KGDE

Dataset	GDE		KGDE	
	Runtime(s)	Rank	Runtime(s)	Rank
lp_blend	0.156	3	0.094	3
lp_pilot4	1.488	3	0.250	3
Ipi_bgdbg1	0.274	6	0.266	6
Ipi_ceria3d	91.180	5	0.234	3
IBMtest2	0.176	1	0.134	1
CAVIAR1	2.235	1	0.757	1

computational complexity of KGDE is not too much related to the dimension of the matrix, KGDE can maintain a fast result.

3.1.2 Synthetic Datasets

For synthetic datasets, our algorithm was compared with the state-of-the-art RPCA-based approaches: RPCA [1], PSSV [36], WNNM [37], and ARE-RPCA [26].

The recovery ability and execution time of the algorithms are investigated. The input matrix $\boldsymbol{D} \in \mathbb{R}^{n\times d}$ is corrupted by sparse noise with the corruption rate $\alpha = \{0.1, 0.3, 0.5, 0.7\}$. The parameters are set: $d = 10000$, $n = \{20, 100, 500\}$ and different ranks $r = \{1, 3, 5\}$. The iteration stop criterion is 10^{-7}. Let $\boldsymbol{L}_{sol}$ be the reconstruction of $\boldsymbol{L}$ and the reconstruction error is defined as $\|\boldsymbol{L}_{sol} - \boldsymbol{L}\|_F / \|\boldsymbol{L}\|_F$.

The comparisons of the average runtime with different ranks r and dimensions d are shown in Fig.3. The eRPCA is faster than others for the average runtime in Fig.3. When the input matrix has a larger scale or rank, the time gap between our method and others is more prominent.

The simulation results are shown in Fig.4. It can be seen that eRPCA algorithm is stubborn when the corruption rate is low and has similar results with other algorithms. When the corruption rate rises, the reconstruction error of eRPCA algorithm is larger. eRPCA performs worse when the corruption rate is 0.7 with a large rank.

Table 2 shows the desired rank of the low-rank matrix decomposed by different algorithms under different corruption rates. It can be observed that the proposed eRPCA can obtain correct ranks of low-rank matrices in all cases. Other algorithms can get the desired rank correctly only if the corruption rate is low.

According to the above analysis, it is observed that eR-

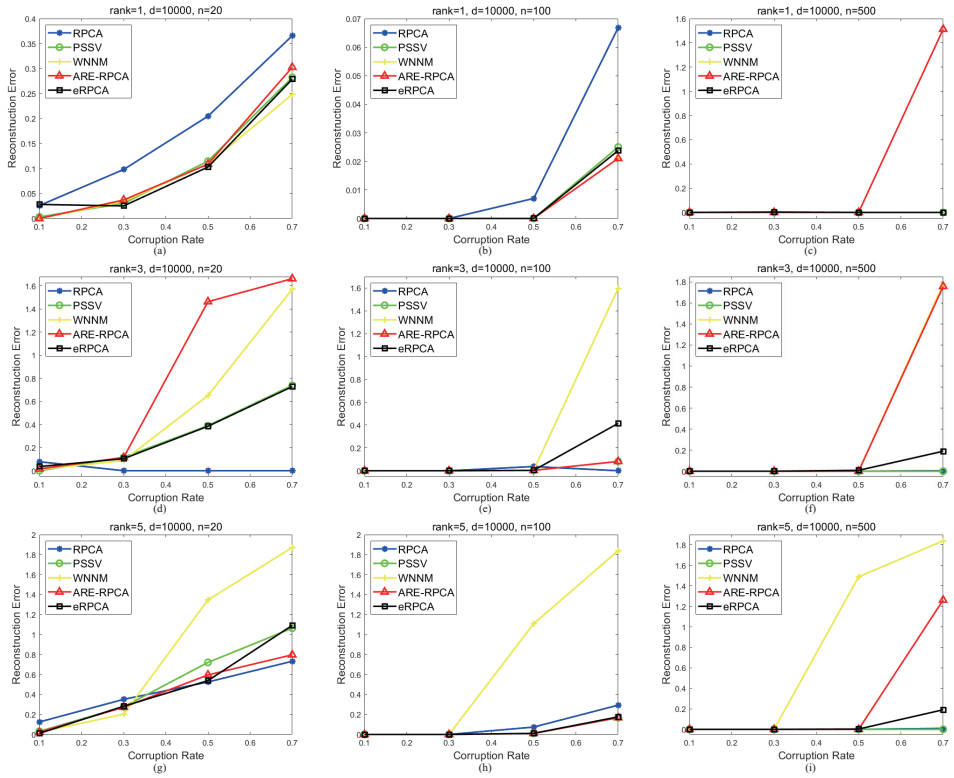

Figure 4. Comparison of various algorithms with different dimensions n, matrix ranks r, and corruption rates α.

Table 2. The desired rank of low-rank matrix decomposed by different algorithms in different corruption rates and dimensions with different ranks

Algorithm	Dimension	$r=1$				$r=3$				$r=5$			
		$\alpha=0.1$	$\alpha=0.3$	$\alpha=0.5$	$\alpha=0.7$	$\alpha=0.1$	$\alpha=0.3$	$\alpha=0.5$	$\alpha=0.7$	$\alpha=0.1$	$\alpha=0.3$	$\alpha=0.5$	$\alpha=0.7$
RPCA	$n=20$	1	1	6	6	4	8	6	5	5	6	5	6
	$n=100$	1	1	1	2	3	3	3	55	5	5	5	52
	$n=500$	1	1	1	7	3	3	3	12	5	5	5	16
PSSV	$n=20$	1	1	1	2	3	3	5	15	5	5	15	19
	$n=100$	1	1	1	1	3	3	3	3	5	5	5	8
	$n=500$	1	1	1	6	3	3	3	10	5	5	5	13
ARE-RPCA	$n=20$	1	1	1	1	3	3	3	5	5	5	5	5
	$n=100$	1	1	1	1	3	3	3	3	5	5	5	5
	$n=500$	1	1	1	1	3	3	3	7	5	5	5	5
WNNM	$n=20$	1	1	1	1	3	3	5	11	5	5	15	20
	$n=100$	1	1	1	1	3	3	3	100	5	5	5	100
	$n=500$	1	1	1	1	3	3	3	500	5	5	5	500
eRPCA	$n=20$	1	1	1	1	3	3	3	3	5	5	5	5
	$n=100$	1	1	1	1	3	3	3	3	5	5	5	5
	$n=500$	1	1	1	1	3	3	3	3	5	5	5	5

PCA has excellent robustness and the fastest excutation time.

3.2. Foreground-Background Separation

In this subsection, we evaluate the proposed eRPCA with the following approaches: ARE-RPCA [26], IRCUR [25], PETRELS [38], and ADW-RPCA [39].

Foreground-background separation is an important application of RPCA. Scene Background Initialization (SBI) dataset[3] [35] is considered. The visual comparisons of the sub-dataset CAVIAR1 are shown in Fig.5, containing 610 frames with size 384×256. The dataset can be represented by a matrix, where each column is a vectorized video frame. Then each algorithm can be applied to decompose the matrix into low-rank parts and sparse ones; that is, this scene can be separated from the moving foreground object from the static background.

There are five frames selected in Fig.5, and the upper and

[3]https://sbmi2015.na.icar.cnr.it/SBIdataset.html

Figure 5. The comparison of visual results on CAVIAR1 between different algorithms.

Table 3. Performance of one frame from CAVIAR1 dataset among different algorithms

	AGE	pEPs%	pCEPS%	MSSSIM	PSNR	CQM	TIME(s)
ARE-RPCA	5.2637	4.0741	3.4871	0.8564	25.4414	24.7876	10.78593
IRCUR	3.1489	0.2828	0.1821	**0.9917**	34.4689	33.2714	**0.007564**
PETRELS	4.9972	5.0161	4.0222	0.9329	29.8017	29.0773	1.14708
ADW-RPCA	2.64	0.3499	0.2482	0.9905	34.4853	33.3402	6.356669
eRPCA	**2.6394**	**0.2686**	**0.1689**	**0.9917**	**34.8626**	**33.5725**	0.034198

lower parts represent the background and foreground. It can be shown that ARE-RPCA and PETRELS yield artifacts in some backgrounds. IRCUR, ADW-RPCA, and eRPCA obtain more distinctive visual results.

In addition, objective evaluations, i.e, AGE, pEP, pCEP, PSNR, MS-SSIM, CQM, are performed on SBI database where the smaller the first three values, the better the separation results, and the last three ones and vice versa. Table 3 shows the performance evaluation of the above six metrics and runtime. For the first six metrics, eRPCA algorithm obtains better results. Due to the time of estimating the rank, eRPCA algorithm is slower than IRCUR for each frame. However, IRCUR needs to determine the rank in advance, leading to its results being slightly inferior to ours. Moreover, only ARE-RPCA and eRPCA algorithm can obtain the rank prior, but ours are better than ARE-RPCA in all aspects.

3.3. Shadow Removal from Face Images

Face images taken under different lighting conditions always introduce errors, for example, uneven lighting, shadow etc, which yields challenge for face recognition. Fig.6 (first column) shows two typical examples with size 192×168 from the Extended Yale B database[4] [40], a benchmark database in face recognition. Each image is converted to a column vector. The comparisons of these methods are given: RPCA [1], PSSV [36], WNNM [37], and ARE-RPCA [26]. In addition, since the images are aligned, the rank is set to 1, and ARE-RPCA and eRPCA can adaptively estimate the rank.

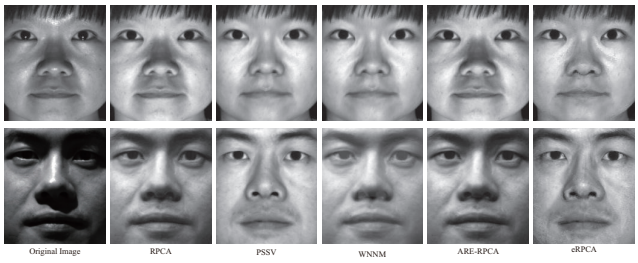

Figure 6. The comparison of different algorithms on two images.

Table 4. The comparison of runtime between different algorithms

	RPCA	PSSV	WNNM	ARE-RPCA	eRPCA
yale05	3.535s	0.973s	1.113s	3.648s	**0.071s**
yaleB02	3.499s	0.962s	2.424s	3.650s	**0.081s**

Fig.6 shows the shadow removal results of different algorithms. In the visual aspect, the proposed eRPCA algorithm removes the shadow well and does not need to determine the rank. The runtime of different algorithms is shown in Table 4 and ours is the fastest.

Due to the space limitation, more experimental results are shown in the Supplementary Material.

4. Conclusion

In this paper, an efficient non-convex RPCA method is proposed for recovering a low-rank matrix and a sparse matrix, especially with large sizes. Specifically, a fast rank estimation method KGDE has been proposed by using Gerschgorin disk theorem with the block Krylov iteration. Based on the estimated rank, a new non-convex regularizer is proposed to achieve better recovery of the low-rank matrices with less computation due to CUR decomposition involved in the iterations. Experimental results demonstrate that the proposed eRPCA algorithm outperforms state-of-the-art algortihms in various vision applications.

In future, our algorithm will be extended in other applications, such as image recovery and video denoising. However, the accuracy of eRPCA for high corruption rate is still a big challenge. Therefore, we will devote ourselves to improving the robustness of the algorithm. Meanwhile, the parameter adaptability is considered in our future work.

[4] http://vision.ucsd.edu/ leekc/ExtYaleDatabase/ExtYaleB.html

References

[1] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011. 1, 6, 8

[2] Spencer Markowitz, Corey Snyder, Yonina C Eldar, and Minh N Do. Multimodal unrolled robust pca for background foreground separation. *IEEE Transactions on Image Processing*, 2022. 1

[3] Thierry Bouwmans, Sajid Javed, Hongyang Zhang, Zhouchen Lin, and Ricardo Otazo. On the applications of robust pca in image and video processing. *Proceedings of the IEEE*, 106(8):1427–1457, 2018. 1

[4] Xuequan Lu, Scott Schaefer, Jun Luo, Lizhuang Ma, and Ying He. Low rank matrix approximation for 3d geometry filtering. *IEEE Transactions on Visualization and Computer Graphics*, 2020. 1

[5] Junpu Wang, Guili Xu, Chunlei Li, Zhengsheng Wang, and Fuju Yan. Surface defects detection using non-convex total variation regularized rpca with kernelization. *IEEE Transactions on Instrumentation and Measurement*, 70:1–13, 2021. 1

[6] Shudong Ou, Ming Zhao, Sen Li, and Tao Zhou. Online shock sensing for rotary machinery using encoder signal. *Mechanical Systems and Signal Processing*, 182:109559, 2023. 1

[7] Namrata Vaswani and Praneeth Narayanamurthy. Static and dynamic robust pca and matrix completion: A review. *Proceedings of the IEEE*, 106(8):1359–1379, 2018. 1

[8] Wenda Chu. Distributed robust principal analysis. *arXiv preprint arXiv:2207.11669*, 2022. 1

[9] Yunlong Gao, Tingting Lin, Yisong Zhang, Sizhe Luo, and Feiping Nie. Robust principal component analysis based on discriminant information. *IEEE Transactions on Knowledge and Data Engineering*, 2021. 1

[10] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. Weighted nuclear norm minimization with application to image denoising. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2862–2869, 2014. 1

[11] Fanhua Shang, James Cheng, Yuanyuan Liu, Zhi-Quan Luo, and Zhouchen Lin. Bilinear factor matrix norm minimization for robust pca: Algorithms and applications. *IEEE transactions on pattern analysis and machine intelligence*, 40(9):2066–2080, 2017. 1

[12] Paris V Giampouras, Athanasios A Rontogiannis, and Konstantinos D Koutroumbas. Robust pca via alternating iteratively reweighted low-rank matrix factorization. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3383–3387. IEEE, 2018. 1

[13] Shiqian Ma and Necdet Serhat Aybat. Efficient optimization algorithms for robust principal component analysis and its variants. *Proceedings of the IEEE*, 106(8):1411–1426, 2018. 1

[14] Tianyi Zhou and Dacheng Tao. Godec: Randomized low-rank & sparse matrix decomposition in noisy case. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, 2011. 1

[15] Tianyi Zhou and Dacheng Tao. Shifted subspaces tracking on sparse outlier for motion segmentation. In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013. 1

[16] Praneeth Netrapalli, Niranjan UN, Sujay Sanghavi, Animashree Anandkumar, and Prateek Jain. Non-convex robust pca. *Advances in Neural Information Processing Systems*, 27, 2014. 1, 2

[17] Xinyang Yi, Dohyung Park, Yudong Chen, and Constantine Caramanis. Fast algorithms for robust pca via gradient descent. *Advances in neural information processing systems*, 29, 2016. 2

[18] Kailing Guo, Liu Liu, Xiangmin Xu, Dong Xu, and Dacheng Tao. Godec+: Fast and robust low-rank matrix decomposition based on maximum correntropy. *IEEE transactions on neural networks and learning systems*, 29(6):2323–2336, 2017. 2

[19] Aritra Dutta, Filip Hanzely, and Peter Richtárik. A nonconvex projection method for robust pca. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1468–1476, 2019. 2

[20] Marcus Valtonen Ornhag, José Pedro Iglesias, and Carl Olsson. Bilinear parameterization for non-separable singular value penalties. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3897–3906, 2021. 2

[21] Bo Jiang, Tianyi Lin, Shiqian Ma, and Shuzhong Zhang. Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis. *Computational Optimization and Applications*, 72(1):115–157, 2019. 2

[22] HanQin Cai, Jian-Feng Cai, and Ke Wei. Accelerated alternating projections for robust principal component analysis. *The Journal of Machine Learning Research*, 20(1):685–717, 2019. 2

[23] Michael Hintermüller and Tao Wu. Robust principal component pursuit via inexact alternating minimization on matrix manifolds. *Journal of Mathematical Imaging and Vision*, 51(3):361–377, 2015. 2

[24] Duy Nhat Phan and Thuy Ngoc Nguyen. An accelerated irnn-iteratively reweighted nuclear norm algorithm for nonconvex nonsmooth low-rank minimization problems. *Journal of Computational and Applied Mathematics*, 396:113602, 2021. 2

[25] HanQin Cai, Keaton Hamm, Longxiu Huang, Jiaqi Li, and Tao Wang. Rapid robust principal component analysis: Cur accelerated inexact low rank estimation. *IEEE Signal Processing Letters*, 28:116–120, 2020. 2, 7

[26] Zhengqin Xu, Rui He, Shoulie Xie, and Shiqian Wu. Adaptive rank estimate in robust principal component analysis. In

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6577–6586, 2021. 2, 3, 6, 7, 8

[27] Cameron Musco and Christopher Musco. Randomized block krylov methods for stronger and faster approximate singular value decomposition. *Advances in neural information processing systems*, 28, 2015. 2, 3

[28] Gene H Golub and Richard Underwood. The block lanczos method for computing eigenvalues. In *Mathematical software*, pages 361–377. Elsevier, 1977. 3

[29] Hsien-Tsai Wu, Jar-Ferr Yang, and Fwu-Kuen Chen. Source number estimators using transformed gerschgorin radii. *IEEE transactions on signal processing*, 43(6):1325–1333, 1995. 3

[30] Christos Boutsidis and David P Woodruff. Optimal cur matrix decompositions. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 353–362, 2014. 4

[31] HanQin Cai, Keaton Hamm, Longxiu Huang, and Deanna Needell. Robust cur decomposition: Theory and imaging applications. *SIAM Journal on Imaging Sciences*, 14(4):1472–1503, 2021. 4

[32] Pham Dinh Tao and LT Hoai An. Convex analysis approach to dc programming: theory, algorithms and applications. *Acta mathematica vietnamica*, 22(1):289–355, 1997. 5

[33] Zhao Kang, Chong Peng, and Qiang Cheng. Robust pca via nonconvex rank approximation. In *2015 IEEE International Conference on Data Mining*, pages 211–220. IEEE, 2015. 5

[34] Timothy A Davis and Yifan Hu. The university of florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)*, 38(1):1–25, 2011. 6

[35] Lucia Maddalena and Alfredo Petrosino. Towards benchmarking scene background initialization. In *International conference on image analysis and processing*, pages 469–476. Springer, 2015. 6, 7

[36] Tae-Hyun Oh, Yu-Wing Tai, Jean-Charles Bazin, Hyeongwoo Kim, and In So Kweon. Partial sum minimization of singular values in robust pca: Algorithm and applications. *IEEE transactions on pattern analysis and machine intelligence*, 38(4):744–758, 2015. 6, 8

[37] Shuhang Gu, Qi Xie, Deyu Meng, Wangmeng Zuo, Xiangchu Feng, and Lei Zhang. Weighted nuclear norm minimization and its applications to low level vision. *International journal of computer vision*, 121(2):183–208, 2017. 6, 8

[38] Nguyen Viet Dung, Nguyen Linh Trung, Karim Abed-Meraim, et al. Robust subspace tracking with missing data and outliers: Novel algorithm with convergence guarantee. *IEEE Transactions on Signal Processing*, 69:2070–2085, 2021. 7

[39] Zhengqin Xu, Huasong Xing, Shun Fang, Shiqian Wu, and Shoulie Xie. Double-weighted low-rank matrix recovery based on rank estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 172–180, 2021. 7

[40] Kuang-Chih Lee, Jeffrey Ho, and David J Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on pattern analysis and machine intelligence*, 27(5):684–698, 2005. 8