Online Compressed Robust PCA

Pingbo Pan[1] Jiashi Feng[2] Ling Chen[1] Yi Yang[1]
[1]Centre for Artificial Intelligence, School of Software, Faculty of Engineering & Information Technology, University of Technology Sydney
[2] Department of ECE, National University of Singapore
lighnt001@gmail.com elefjia@nus.edu.sg {Ling.Chen, Yi.Yang}@uts.edu.au

***Abstract*—In this work, we consider the problem of robust principal component analysis (RPCA) for streaming noisy data that has been highly compressed. This problem is prominent when one deals with high-dimensional and large-scale data and data compression is necessary. To solve this problem, we propose an online compressed RPCA algorithm to efficiently recover the low-rank components of raw data. Though data compression incurs severe information loss, we provide deep analysis on the proposed algorithm and prove that the low-rank component can be asymptotically recovered under mild conditions. Compared with other recent works on compressed RPCA, our algorithm reduces the memory cost significantly by processing data in an online fashion and reduces the communication cost by accepting sequential compressed data as input.**

I. Introduction

In recent years, many algorithms have been proposed to robustify standard principal component analysis (PCA) [4], [14] to deal with gross noise or outliers in realistic data [2], [6], [3], [22], [17], [27]. Among them, a popular robust PCA (RPCA) algorithm for recovering a low-dimensional subspace underlying the noisy data is to factorize the data matrix into a low-rank component plus a sparse component – through solving the following principal component pursuit (PCP) problem [3]:

$$\begin{aligned} \min_{L,E} \quad & \|L\|_* + \lambda\|E\|_1, \\ \text{s.t.} \quad & X = L + E. \end{aligned}$$

Here each column in the matrix X represents one sample of the observations. The matrices L and E are respectively encouraged to be low-rank (accounting for the low-dimensional subspace structure) and sparse (explaining the gross noise on the observations) by the imposed matrix nuclear norm $\|\cdot\|_*$ and the ℓ_1 norm $\|\cdot\|_1$.

Studies on the RPCA problems have achieved great success in theories as well as practices on simple datasets [28], [18]. However, most of existing algorithms require to access the whole set of raw data *directly* to solve the RPCA problems. Thus they are usually computationally expensive when dealing with large-scale and high-dimensional data such as video sequence and high-resolution images.

To address the challenge of high memory cost for RPCA, one possible solution is to compress the large-scale data first and then analyze size-reduced data in a compressed domain (*e.g.*, randomly project the raw data into a low-dimensional space) instead of in the original domain. This is motivated by the practical observations that realistic high-dimensional data is often intrinsically redundant, and thus analyzing properly compressed data directly can be much more efficient.

In fact, several methods have been developed to solve the problems of RPCA in the compressed domain. [25] gives a proof that when the number of observations exceeds the number of intrinsic degree of freedom of the component signals by a polylogarithmic factor, the low-rank and sparse terms can be exactly recovered by convex optimization. SpaRCS [23] is a loop-based greedy algorithm which combines CoSaMP [16] and Admira [11]. In each iteration, SpaRCS adopts CoSaMP to recover the sparse term and Admira to recover the low-rank term.

With the spirit of RPCA, they try to solve the following optimization problem to recover the row rank and sparse components from the compressed data directly:

$$\begin{aligned} \min_{L,E} \quad & \|L\|_* + \lambda\|E\|_1 \\ \text{s.t.} \quad & Y = \mathcal{A}(L + E). \end{aligned}$$

Here $\mathcal{A}$ is a known data compression operator (*e.g.*, random projection, random sampling) and Y is the observed compressed data. Although compressing data before RPCA analysis indeed reduces memory cost, those methods still fail to handle large-scale data sets as they need to load all the compressed data into memory for computation.

In this work, we propose an *online* compressed RPCA algorithm – termed as OCRPCA – to more efficiently solve compressed PCA problems, with *provable* performance guarantee. OCRPCA specifically considers *linear* compression operators $\mathcal{A}$ for compressing streaming data, such as noiselet [5], random sampling and random projection [20], [12], [7]. Given a sequence of compressed data $Y = \mathcal{A}(X) = [y_1, y_2, \ldots, y_T]$, OCRPCA directly recovers the low rank component of raw data X from compressed data Y in an online manner. Here $X = [x_1, x_2, \ldots, x_T]$ is a sequence of the raw high-dimensional data and $y_t = \mathcal{A}(x_t), t = 1, \ldots, T$.

The proposed OCRPCA offers several appealing advantages in both practice and theory. First of all, compared with existing online RPCA algorithms such as ORPCA [8], [9], [13], OCRPCA only needs to access the compressed data, whose size has been reduced significantly. Therefore, the data communication cost that is prominent in modern data processing systems can be reduced dramatically. Secondly, compared with batch compressed RPCA method, OCRPCA does not need to revisit all the former samples when a new sample is revealed. Thus, OCRPCA can efficiently deal with sequential and dynamic inputs, such as video sequences. Thirdly, rather than loading all the samples into memory, OCRPCA only needs to load one sample at each time instance. This property of OCRPCA offers greater efficiency benefits for dealing with big data. We also provide sharp analysis on the statistical performance of OCRPCA and demonstrate that the estimation offered by OCRPCA converges to the global optimum asymptotically.

II. Problem Formulation

A. Notations

Throughout the paper, we use capital letters to denote matrices. In particular, let $X \in \mathbb{R}^{d\times T}$ denote the collection of T noisy raw data in $\mathbb{R}^d$ without compression and x_i denote the ith column of the matrix X. The problem of RPCA is to find an underlying low rank component L and a sparse noise component E of X. We use $\mathcal{A}$ to denote the linear compression operator and $\mathcal{A}^\top$ to denote its adjoint. $Y \in \mathbb{R}^{p\times T}$ is the observed compressed data, for which we have $Y = \mathcal{A}(X) = [y_1, y_2, ..., y_T]$, and specifically $y_i = \mathcal{A}(x_i)$. Given $a' = \mathcal{A}(a)$, we define the *compression ratio* of the operator $\mathcal{A}$ as $s \triangleq \frac{\dim(a')}{\dim(a)}$, where $\dim(a)$ denotes the dimension of a data point a and the smaller compression ratio means more significant compression. In analysis on the recovery performance, the incoherence of the low rank component L is involved, which is defined as follows.

Definition 1 (μ-incoherence [25]). *A low-rank matrix $L \in \mathbb{R}^{m\times n}$ has rank-reduced singular value decomposition $L = U\Sigma V^\top$, then L is μ-incoherent if*

$$\forall i,j, \|U^\top e_i\|_2^2 \le \frac{\mu r}{m}, \|V^\top e_j\|_2^2 \le \frac{\mu r}{n}, \|UV^\top\|_\infty \le \sqrt{\frac{\mu r}{mn}}$$

Here r is the rank of L and e_i is the ith canonical basis vector in Euclidean space (the vector with all the entries equal to 0 but the ith equal to 1).

B. Online Compressed Robust PCA

We focus on solving the problem of RPCA in the compressed domain. The compression operator $\mathcal{A}$ is known and only the compressed data is observed. Formally, the optimization problem we are going to solve is:

$$\min_{L,E} \frac{1}{2}\|Y - \mathcal{A}(L+E)\|_F^2 + \lambda_1\|L\|_* + \lambda_2\|E\|_1, \tag{1}$$

with λ_1 and λ_2 be two pre-defined trade-off parameters. We will explain how to choose the values of λ_1 and λ_2 in the experiments.

Most of existing works, such as SpaRCS [23], solve the above compressed RPCA problem through explicit recovering the raw high-dimensional data and then performing SVDs on it, which restricts their scalability to big data. In this work, we develop an online learning algorithm to solve the above problem. Performing the online learning is not trivial for the above optimization problem, as the involved nuclear norm tightly couples the sampels together, so they cannot be processed individually. Inspired by the recent work of online RPCA [8], we propose a new algorithm that avoids this problem by reformulating the above optimization problem in a bilinear form, based on the property of nuclear norm: $\|L\|_* = \min_{L=UV^\top}\left(\|U\|_F^2 + \|V\|_F^2\right)/2$ [19]. Here $U \in \mathbb{R}^{d\times r}$ and $V \in \mathbb{R}^{T\times r}$, and r is the rank of L. Then it is clear that the optimization problem is equivalent to:

$$\min_{U,V,E} \quad \frac{1}{2}\|Y - \mathcal{A}(UV^\top + E)\|_F^2 + \frac{\lambda_1}{2}(\|U\|_F^2 + \|V\|_F^2) + \lambda_2\|E\|_1.$$

In order to simplify the optimization, we here introduce an auxiliary variable $X = UV^\top + E$ that serves as a surrogate of the raw data. Then the above optimization problem can be written as:

$$\begin{aligned}\min_{U,V,E} \quad & \frac{1}{2}\|Y - \mathcal{A}(X)\|_F^2 + \frac{\lambda_1}{2}\left(\|U\|_F^2 + \|V\|_F^2\right) + \lambda_2\|E\|_1 \\ & + \frac{\lambda_3}{2}\|X\|_F^2 \\ \text{s.t.} \quad & X = UV^\top + E.\end{aligned} \tag{2}$$

Here the extra regularization term $\frac{\lambda_3}{2}\|X\|_F^2$ is used to prevent the magnitude of raw data from being arbitrarily large. Note that the re-formulated optimization problem is actually non-convex as there is a bilinear term of $UV^\top$. Fortunately, we can provide a performance guarantee that our algorithm can successfully solve the above non-convex problem and asymptotically recover the low-rank and sparse components from the compressed data. Details of the proof are provided in Section III of Proof.

We now proceed to explain the details of our proposed OCRPCA algorithm. The algorithm actually solves an equivalent optimization problem with penalty as follows:

$$\begin{aligned}L(U,V,E,X) = & \frac{1}{2}\|Y - \mathcal{A}(X)\|_F^2 + \frac{\lambda_1}{2}\left(\|U\|_F^2 + \|V\|_F^2\right) \\ & + \lambda_2\|E\|_1 + \frac{\mu}{2}\|X - UV^\top - E\|_F^2 + \frac{\lambda_3}{2}\|X\|_F^2.\end{aligned} \tag{3}$$

Compared with the objective function in (2), it has an additional penalty term: $\frac{\mu}{2}\|X - UV^\top - E\|_F^2$, where μ is the penalty parameter.

Let $Y = [y_1, y_2, ..., y_T]$, $X = [x_1, x_2, ..., x_T]$, $V^\top = [v_1, v_2, ..., v_T]$ and $E = [e_1, e_2, ..., e_T]$, then the loss function in the above problem can be written as $L(U,V,E,X) = \sum_{i=1}^T h(y_i, U, v_i, e_i, x_i) + \frac{\lambda_1}{2}\|U\|_F^2$, where:

$$\begin{aligned}h(y_i, U, v_i, e_i, x_i) = & \frac{1}{2}\|y_i - \mathcal{A}(x_i)\|_2^2 + \frac{\lambda_1}{2}\|v_i\|_2^2 \\ & + \lambda_2\|e_i\|_1 + \frac{\lambda_3}{2}\|x_i\|_2^2 + \frac{\mu}{2}\|x_i - Uv_i - e_i\|_2^2.\end{aligned}$$

Then, v_i can be updated by solving the following quadratic programming problem:

$$\begin{aligned}v_i = & \arg\min_{v_i} \frac{\mu}{2}\|x_i - Uv_i - e_i\|_2^2 + \frac{\lambda_1}{2}\|v_i\|_2^2 \\ = & (\lambda_1 I + \mu U^\top U)^{-1}U^\top(\mu x_i - \mu e_i),\end{aligned} \tag{4}$$

and e_i can be updated by solving the following Lasso type problem:

$$e_i = \arg\min_{e_i} \frac{\mu}{2}\|x_i - Uv_i - e_i\|_2^2 + \lambda_2\|e_i\|_1 = \mathcal{S}_{\frac{2\lambda_2}{\mu}}(x_i - Uv_i). \tag{5}$$

Here $\mathcal{S}_\alpha(x)$ is a shrinkage operator over x with a parameter $\alpha : \mathcal{S}_\alpha(x) \triangleq \mathrm{sgn}(x) \odot \max\{|x| - \alpha, 0\}$.

Since $\mathcal{A}$ is a linear operator, x_i can be updated by solving a quadratic programming problem:

$$x_i = \arg\min_{x_i} \|\mathcal{A}(x_i) - y_i\|_2^2 + \frac{\mu}{2}\|x_i - Uv_i - e_i\|_2^2 + \frac{\lambda_3}{2}\|x_i\|_2^2 \tag{6}$$

The update of U is quite different from that of v_i, e_i, x_i. U is the basis matrix which needs to be updated according to the sequential data and we need to minimize the following empirical cost function.

$$f_T(U) = \frac{1}{T}\sum_{i=1}^{T} l(y_i, U) + \frac{\lambda_1}{2T}\|U\|_F^2, \tag{7}$$

where the *loss function* for each sample is:

$$l(y_i, U) = \min_{v,e,x} h(y_i, U, v, e, x). \tag{8}$$

In the stochastic optimization, one is usually interested in the minimization of the *expected loss* of the overall samples [15].

$$f(U) = \mathbb{E}_x[l(y, U)] = \mathbb{E}_y[(l(y, U)] = \lim_{T\to\infty} f_T(U), \tag{9}$$

where the first equation is based on the fact that y is deterministically dependent on x via $y = \mathcal{A}(x)$.

We develop our OCRPCA method based on the above analysis. The basic idea is to develop a stochastic optimization algorithm to minimize the empirical cost function (7) for streaming data. The coefficients v_i, e_i, x_i are alternatively optimized according to Equations (4), (5) and (6) respectively. At the t-th time instance, we use the previously estimated coefficients $\{v_i\}_{i=1}^t$, sparse noise $\{e_i\}_{i=1}^t$ and recovered high-dimensional data $\{x_i\}_{i=1}^t$ to establish a surrogate cost function and then minimize it to calculate the basis U_t. The surrogate loss function we are going to work with is defined as:

$$g_t(U) \triangleq \frac{1}{t}\sum_{i=1}^{t} h(y_i, U, v_i, e_i, x_i) + \frac{\lambda_1}{2t}\|U\|_F^2. \tag{10}$$

It is easy to verify that this function renders an upper bound for $f_t(U)$: $g_t(U) \geq f_t(U)$.

Combining the above optimization steps gives our proposed OCRPCA algorithm, with details provided in Algorithm 1. Here we consider the linear compression operator $\mathcal{A}$, and thus x_i is updated via a simple quadratic programming. To improve the convergence, we update U by adopting the block-coordinate descent with warm restarts [1]. It is described in Algorithm 2.

Algorithm 1: Online Compressed Robust PCA (OCRPCA)

```
Input : [$y_1, y_2, ..., y_T$] (a sequence of observed data), $\mathcal{A}, \mathcal{A}^\top$
        (the compression operator and its inverse operator),
        $\lambda_1, \lambda_2, \mu \in \mathbb{R}$ (regularization parameters)
Initialize: $U_0$ is a random matrix ;
for $i = 1$ to $T$ do
    initialize $v_i = 0, e_i = 0, x_i = \mathcal{A}(y_i)$ ;
    while not converge do
        $v_i \leftarrow (\lambda_1 I + \mu U_{i-1}^\top U_{i-1})^{-1} U_{i-1}^\top(\mu x_i - \mu e_i)$ ;
        $e_i \leftarrow \mathcal{S}_{\frac{2\lambda_2}{\mu}}(x_i - U_{i-1}v_i)$ ;
        $x_i \leftarrow \arg\min_{x_i} \frac{1}{2}\|\mathcal{A}(x_i) - y_i\|_F^2 + \frac{\mu}{2}\|x_i - U_{i-1}v_i - e_i\|_F^2 + \frac{\lambda_3}{2}\|x_i\|_2^2$;
    end
    $A_i \leftarrow A_{i-1} + (\mu x_i - \mu e_i)v_i^\top$ ;
    $B_i \leftarrow B_{i-1} + \mu v_i v_i^\top$ ;
    Compute $U_i$ with $U_{i-1}$ as warm restart using Algorithm 2 ;
end
Output: $U, V, E$
```

Algorithm 2: The Basis Update

```
Input : $U = [u_1, u_2, ..., u_r] \in \mathbf{R}^{p\times r}, A = [a_1, a_2, ..., a_r] \in$
        $\mathbb{R}^{p\times r}, B = [b_1, b_2, ..., b_r] \in \mathbb{R}^{r\times r}$
$\tilde{B} = B + \lambda_1 I$;
for $j = 1$ to $r$ do
    $u_j \leftarrow \frac{1}{\tilde{B}_{j,j}}(a_j - U\tilde{b}_j) + u_j$ ;
end
```

C. Performance Guarantees

In this work, we propose deep analysis on the solution presented in Algorithm 1, and provide a guarantee in the following theorem that the final solution will converge to the optimum of the batch problem.

$$\begin{aligned} \min_{L,E} \quad & \frac{1}{2}\|Y - \mathcal{A}(X)\|_F^2 + \frac{\lambda_1}{2}\|L\|_F^2 + \lambda_2\|E\|_1 + \frac{\lambda_3}{2}\|X\|_F^2 \\ \text{s.t.} \quad & X = L + E. \end{aligned}$$

Theorem 1 (convergence guarantee)**.** *Assume the magnitudes of observations are always upper bounded and the compression operator $\mathcal{A}$ is a linear operator. Suppose the rank of the optimal solution U of the problem in* (9) *is r, and the solution $U_t \in \mathbb{R}^{p\times r}$ provided by Algorithm 1 is full rank, then U_t converges to the optimal solution of* (9) *asymptotically as $t \to \infty$.*

Applying the above convergence result, along with the recovery condition for batch algorithm in [25], guarantees that OCRPCA can recover the low-rank components from compressed data with a high probability.

Theorem 2 (main result)**.** *Let $X = L_0 + E_0 \in \mathbb{R}^{d\times T}$, with $d \geq T$, and suppose that $L_0 \neq 0$, with a rank r, which is a μ-incoherent matrix with*

$$r \leq \frac{c_r T}{\mu \log^2 d},$$

and $\mathrm{sgn}(E_0)$ *is iid Bernoulli-Rademacher with nonzero probability* $\rho < c_\mathrm{e}$. *Let* $Q \subset \mathbb{R}^{d\times T}$ *be a random subspace of dimension*

$$\dim(Q) \geq C_d \cdot (\rho dT + dr) \cdot \log^2 d$$

distributed according to the Haar measure [10], probabilistically independent of $\mathrm{sgn}(E_0)$. *Then with probability at least* $1 - Cd^{-9}$ *in* $(sign(E_0), Q)$, *the asymptotic solution by OCRPCA with* $\lambda = 1/\sqrt{d}$ *is unique, and equal to* L_0. *Above,* c_r, c_e, C_d *and* C *are positive constants.*

III. Proof

In this section, we provide the proof of our main result in Theorem 2.

The compression operator $\mathcal{A}$ used in our algorithm is linear operator, so we write $\mathcal{A}(X)$ as AX in this section for notational convenience. Here A is the corresponding projection matrix of the compression operator $\mathcal{A}$.

The proof of Theorem 1 proceeds in the following four steps: (I) we first prove the surrogate function $g_t(U)$ converges almost surely; (II) we then prove the solution difference behaves as $\|U_t - U_{t-1}\|_F = O(1/t)$; (III) based on (II) we show $f(U_t) - g_t(U_t) \to 0$ almost surely, and the gradient of f becomes zero at solution U_t when $t \to \infty$; (IV) finally we prove that U_t asymptotically converges to the optimum solution of (9), which recovers the ground truth with high probability under mild conditions.

Lemma 1. *Assume* A *is the linear compression matrix.* $v^* \in \mathbb{R}^r$, $e^* \in \mathbb{R}^d$ *and* $x^* \in \mathbb{R}^d$ *is a solution of* (8) *if and only if*

$$\begin{aligned}
&x^* = (A^\top A + Q + \lambda_3 I)^{-1}(A^\top y + Qe^*)\\
&v^* = PU^\top((A^\top A + Q)^{-1}(A^\top y + Qe^*) - e^*)\\
&C_\Lambda e^*_\Lambda + b_\Lambda = \lambda_2 \mathrm{sign}(e^*_\Lambda)\\
&C_{\Lambda^c} e^*_{\Lambda^c} + b_{\Lambda^c} \leq \lambda_2
\end{aligned}$$

where $P = (U^\top U + \lambda_1/\mu I)^{-1}$, $Q = (1/\mu I + 1/\lambda_1 UU^T)^{-1}$, $C = ((A^\top A + Q)^{-1}Q - I)Q$ $b = \mu(A^\top A + Q)^{-1}A^\top y - \mu UPU^\top(A^\top A + Q)^{-1}A^\top y$, $u \in \partial\|e^*\|_1$, *and* C_Λ *denotes the elements of matrix* C *indexed by* $\Lambda \times \Lambda, \Lambda = \{j | e^*[j] \neq 0\}$. *Moreover, the optimal solution is unique.*

The proof of Lemma 1 is quite straightforward. We first calculate the partial derivative of the objective function (8) and then the first order optimal condition on the objective function is applied. In addition the uniqueness of the optimal solution, as guaranteed in Lemma 1, offers the following theorems and lemmas.

Lemma 2. *Assume the observation* y *always has upper bounded magnitude and* A *is a linear compression operator. Define*

$$\{v^*, e^*, x^*\} = \underset{v,e,x}{\arg\min} \frac{1}{2}\|y - Ax\|_F^2 + \frac{\lambda_1}{2}\|v\|_F^2 + \lambda_2\|e\|_1 + \frac{\mu}{2}\|x - Uv - e\|_F^2 + \frac{\lambda_3}{2}\|x\|_2^2.$$

Then, we have 1) the function defined in (8) *is continuously differentiable and*

$$\nabla_U l(y, U) = \mu(Uv^* + e^* - x^*)v^{*\top}$$

2) $\nabla f(U) = \mathbb{E}_y[\nabla_U l(y, U)]$; *3)* $\nabla f(U)$ *is Lipschitz.*

Proof. The function $h(y, U, v, x, e)$ is continues, and for all $v \in \mathbb{R}^r, x \in \mathbb{R}^d, e \in \mathbb{R}^d$, the function $h(., ., v, x, e)$ is differentiable, and the derivative $\nabla_U h(., ., v, x, e)$ is continuous. According to Lemma 1, $h(., ., v, x, e)$ has unique minimizer (v^*, e^*, x^*). We directly apply Lemma 2 and obtain that $l(y, U)$ is differentiable in U and

$$\nabla_U l(y, U) = \nabla_U h(y, U, v^*, x^*, e^*) = \mu(Uv^* + e^* - x^*)v^{*\top}.$$

Proof of the second claim

According to the first claim, function $l(y, U)$ is continuously differentiable, thus

$$\nabla_U f(U) = \nabla_U \mathbb{E}_y[l(y, U)] = \mathbb{E}_y[\nabla_U l(y, U)]$$

Proof of the third claim

To prove $\nabla f(U)$ is Lipschitz, we will show that for all bounded observations y, $v^*(y, .), x^*(y, .)$ and $e^*(y, .)$ are Lipschitz with constants independent of y. The loss function $l(y, U)$ is continues in y, U, v, e, x and according to Lemma 1 , for fixed y and U, it has a unique minimum, so the optimal solutions v^*, e^*, x^* are continuous in U and y.

Consider a matrix U and a sample y, the corresponding optimal solutions are v^*, x^*, e^*. Assume C and b are defined as those of Lemma 1, and Λ is the set of the indices such that $|C_\Lambda e_\Lambda + b_\Lambda| = \lambda_1$. Since C_Λ is nonsingular, $C_\Lambda e_\Lambda + b_\Lambda$ is continuous in U, y. We consider a small perturbation of (y, U) in their open neighbourhood V, where for all $(y', U') \in V$, we have if $j \notin \Lambda$, $|C'_j(y'[j] - e^{*\prime}[j])| < \lambda_1$ and $e^*[j] = 0$. Thus the support set of e^* will not change in set V. Moreover, v^*, e^*, x^* are continuous in the set V.

We consider the function $h(y, U, v, x, e)$. The Hessian matrix of the function $h(y, U, ., ., .)$ w.r.t. x is $A^\top A + \mu I + \lambda_3 I$, the Hessian matrix of the function $h(y, U, ., ., .)$ w.r.t. v is $L^\top L + \lambda_1 I$, the Hessian matrix of the function $h(y, U, ., ., .)$ w.r.t. e is I. So all these Hessian matrices are positive definite. We have the function $h(y, U, ., ., .)$ is strictly convex and

$$\begin{aligned}
&h(y, U, v^{*\prime}, x^{*\prime}, e^{*\prime}) - h(y, U, v^*, x^*, e^*)\\
\geq\ &\lambda_1\|v^{*\prime} - v^*\|_2^2 + \lambda_3\|x^{*\prime} - x^*\|_2^2 + \|e^{*\prime} - e^*\|\\
\geq\ &\min(\lambda_1, \lambda_3, 1)(\|v^{*\prime} - v^*\|_2^2 + \|x^{*\prime} - x^*\|_2^2 + \|e^{*\prime} - e^*\|),
\end{aligned} \tag{11}$$

We then show the function $h(y, U, ., ., .) - h(y', U', ., ., .)$ is Lipschitz continuous. Define a matrix $H = [I, -U, -I]$, a vector $d = [x; v^\top; e]$. Assume matrix A is the projection matrix of compression operator $\mathcal{A}$. The difference of these two functions is

$$\begin{aligned} & (h(y,U,v,x,e) - h(y',U',v,x,e)) \\ & - (h(y,U,v',x',e') - h(y',U',v',x',e')) \\ = \; & \frac{\mu}{2}(\|Hd\|_2^2 - \|H'd\|_2^2) - \frac{\mu}{2}(\|Hd'\|_2^2 - \|H'd'\|_2^2) \\ & + \frac{1}{2}(\|y - Ax\|_2^2 - \|y' - Ax\|_2^2) \\ & - \frac{1}{2}(\|y - Ax'\|_2^2 - \|y' - Ax'\|_2^2) \end{aligned}$$

It is easy to show that the function $\|Hd\|_2^2 - \|H'd\|_F^2$ is Lipschitz with constant as $c_1\|H-H'\|_2$, where c_1 is a constant independent of H, H'. It is not difficult to show function $\|y - Ax\|_2^2 - \|y' - Ax\|_2^2$ is also Lipschitz with constant as $c_2\|y - y'\|_2$, where c_2 is a constant independent of y, y'. Thus we have,

$$\begin{aligned} & (h(y,U,v,x,e) - h(y',U',v,x,e)) \\ & - (h(y,U,v',x',e') - h(y',U',v',x',e')) \\ \leq \; & c_1\|H - H'\|_2\|d - d'\|_2 + c_2\|y - y'\|_2\|x - x'\|_2 \\ \leq \; & (c_1\|U - U'\|_F + c_2\|y - y'\|_2) \\ & * (\|x - x'\|_2 + \|v - v'\|_2 + \|e - e'\|_2) \end{aligned}$$

Then according to the (11) of this section, and considering $(v^{*\prime}, e^{*\prime}, x^{*\prime})$ minimizes $h(y', U', ., ., .)$, we have

$$\begin{aligned} & \min(\lambda_1, \lambda_3, 1)(\|v^{*\prime} - v^*\|_2^2 + \|x^{*\prime} - x^*\|_2^2 + \|e^{*\prime} - e^*\|_2^2) \\ \leq \; & h(y,U,v^{*\prime},x^{*\prime},e^{*\prime}) - h(y,U,v^*,x^*,e^*) \\ \leq \; & h(y,U,v^{*\prime},x^{*\prime},e^{*\prime}) - h(y',U',v^{*\prime},x^{*\prime},e^{*\prime}) \\ & + h(y',U',v^*,x^*,e^*) - h(y,U,v^*,x^*,e^*) \\ \leq \; & (c_1\|U - U'\|_F + c_2\|y - y'\|_2) \\ & * (\|v^{*\prime} - v^*\|_2 + \|x^{*\prime} - x^*\|_2 + \|e^{*\prime} - e^*\|_2) \end{aligned}$$

So we have

$$\begin{aligned} & (\|v^{*\prime} - v^*\|_2 + \|x^{*\prime} - x^*\|_2 + \|e^{*\prime} - e^*\|_2) \\ \leq \; & \frac{1}{\min(\lambda_1, \lambda_3, 1)}(c_1\|U - U'\|_F + c_2\|y - y'\|_2) \end{aligned}$$

Combining the second claim, we can conclude the third claim. The proof of the convergence of f can be derived from Lemma 2. □

Theorem 3 (convergence of the surrogate function g_t)**.** *Let g_t denote the surrogate function defined in* (10)*. Then, $g_t(U_t)$ converges almost surely when the solution U_t is given by Algorithm 1.*

The convergence of the surrogate function g_t provides a guarantee for the convergence of the function f.

Theorem 4 (difference of the solution U_t)**.** *For the two successive solutions obtained from the Algorithm 1, we have*

$$\|U_{t+1} - U_t\|_F = O(1/t)$$

Theorem 4 actually provides a guarantee of the convergence of U_t. Since the purpose of our algorithm is to recover the basis matrix U from the compressed data.

Theorem 5 (convergence of f)**.** *Let g_t denote the surrogate function defined in* (10)*. Then, 1) $f(U_t) - g_t(U_t)$ converges almost surely to 0; and 2) $f(U_t)$ converges almost surely, when the solution U_t is given by Algorithm 1.*

The convergence of f is proved by first showing $g_t(U_t)$ converges to $f(U_t)$ as $t \to \infty$, then combining Theorem 3 and 4 gives Theorem 5.

Theorem 6. *The first order optimal condition for minimizing the objective function in* (9) *is satisfied by U_t, the solution provided by Algorithm 1, when t tends to infinity.*

According to Theorem 6, the U_∞ obtained form Algorithm 1 satisfies the first order optimality condition for minimizing the expected cost $f(U)$. Our OCRPCA algorithm can calculate a solution that converges to a stationary point of the expected loss (9). Until now we finish the convergence analysis of our algorithm.

Details of the proofs for Theorem 3, Theorem 4, Theorem 5 and Theorem 6 can be found in [8].

Theorem 7. *When solution U satisfies the first order condition for minimizing the objective function in* (9)*, the obtained solution U is the optimal solution of the problem* (9) *if U is full rank.*

Proof. The minimizer of the objective function in (9) is

$$\min_U \lim_{T\to\infty} \frac{1}{T}\sum_{i=1}^{T} l(y_i, U)$$

which is equivalent to

$$\begin{aligned} \min_{X,U,V,E} \; & \frac{1}{2}\|Y - AX\|_F^2 + \frac{\lambda_1}{2}(\|U\|_F^2 + \|V\|_F^2) + \lambda_2\|E\|_1 \\ & + \frac{\lambda_3}{2}\|X\|_F^2 + \frac{\mu}{2}\|X - UV^\top - E\|_F^2. \end{aligned} \tag{12}$$

Where $Y = [y_1, y_2, ..., y_T], X = [x_1, x_2, ..., x_T], E = [e_1, e_2, ..., e_T], V^\top = [v_1^\top, v_2^\top, ..., v_T^\top]$.

We can calculate the following equations

$$(A^\top A + \mu I + \lambda_3 I)X - \mu UV^\top - \mu E = 0 \tag{13}$$
$$\mu(VU^\top - Z^\top)U + \lambda_1 V = 0 \tag{14}$$
$$\mu(UV^\top - Z)V + \lambda_1 U = 0 \tag{15}$$
$$\mu(UV^\top - Z) \in \lambda_2 \partial\|E\|_1 \tag{16}$$

Here $Z = X - E$. Note that for any invertible matrix Q, the pair $(UQ, VQ^{-1^\top})$ provides a factorization equivalent to (U, V). According to [21] any solution (U, V) can be orthogonalized to an equivalent orthogonal solution $\bar{U} = UQ, \bar{V} = VQ^{-1^\top}$ such that $\bar{U}^\top\bar{U} = \Lambda_U$ and $\bar{V}^\top\bar{V} = \Lambda_V$ are diagonal matrices. When we replace U, V by $\bar{U}, \bar{V}$ in (14) and (15), it is easy to find that $\Lambda_U = \Lambda_V = \Lambda$.

Since orthogonalization operation is always performed on our algorithm, we focus on the orthogonal solution, where $U^\top U = \Lambda \in \mathbb{R}^{r\times r}$ and $V^\top V = \Lambda \in \mathbb{R}^{r\times r}$. Since U and R are full rank, the elements in the diagonal of matrix Λ are non-zero.

From (15) we can obtain

$$U = ZV(V^\top V + \frac{\lambda_1}{\mu} I)^{-1} = ZV(\Lambda + \frac{\lambda_1}{\mu} I)^{-1} \tag{17}$$

Substituting back into (14), we have

$$V\Lambda - Z^\top U + \frac{\lambda_1}{\mu} V = 0$$

Then,

$$V\Lambda - Z^\top ZV(\Lambda + \frac{\lambda_1}{\mu} I)^{-1} + \frac{\lambda_1}{\mu} V = 0$$

$$V(\Lambda + \frac{\lambda_1}{\mu} I)^2 = Z^\top ZV. \tag{18}$$

Define $V' = V(\sqrt{\Lambda})^{-1}$, then we have ${V'}^\top V' = (\sqrt{\Lambda})^{-1} V^\top V (\sqrt{\Lambda})^{-1} = I$. Namely, the matrix V' is an orthogonal matrix. From the above equation, we conclude that

$$V'\sqrt{\Lambda}(\Lambda + \frac{\lambda_1}{\mu} I)^2 = Z^\top Z V' \sqrt{\Lambda}$$

$$V'(\Lambda + \frac{\lambda_1}{\mu} I)^2 = Z^\top Z V'$$

Therefore, the columns of the matrix V' are the eigenvectors of the matrix $Z^\top Z$. Thus the columns of the matrix V are the eigenvectors of the matrix $Z^\top Z$ scaled by the square root of the matrix Λ. And the eigenvalues of the matrix $Z^\top Z$ are the elements in the diagonal of matrix $(\Lambda + \lambda_1 I)^2$.

From (17) we have

$$ZZ^\top U = ZZ^\top ZV(\Lambda + \frac{\lambda_1}{\mu} I)^{-1} \overset{(18)}{=} ZV(\Lambda + \frac{\lambda_1}{\mu} I) \overset{(17)}{=} U(\Lambda + \frac{\lambda_1}{\mu} I)^2.$$

Thus similar to V, the columns of matrix U correspond to the eigenvector of the matrix $ZZ^\top$ scaled by the square root of the matrix Λ.

Performing SVD on the matrix Z provides $Z = P\Sigma D^\top = P_1 \Sigma_1 D_1^\top + P_2 \Sigma_2 D_2^\top$. Here $P_1^\top P_2 = 0$, $D_1^\top D_2 = 0$ and $\Sigma_1 \in \mathbb{R}^{r \times r}$, $\Sigma_2 \in \mathbb{R}^{(n-r)\times(n-r)}$.

From the above results, we can obtain $U = P_1\sqrt{\Lambda}$ and $V = D_1\sqrt{\Lambda}$.

$$Z^\top Z = D\Sigma^2 D^\top.$$

Thus, we have

$$\Sigma_1 = \Lambda + \frac{\lambda_1}{\mu} I.$$

Since matrix U is full rank, $U^\top U = \Lambda$ is positive definite. Thus $\Sigma_1 \succ \frac{\lambda_1}{\mu} I$.

The obtained solution $L = UV^\top = P_1 \Lambda D_1^\top = P_1(\Sigma_1 - \frac{\lambda_1}{\mu} I) D_1^\top$. We can obtain that

$$\begin{aligned} Z - L = {} & P\Sigma D^\top - P_1(\Sigma_1 - \frac{\lambda_1}{\mu} I) D_1^\top \\ = {} & \frac{\lambda_1}{\mu} P_1 D_1^\top + P_2 \Sigma_2 D_2^\top = \frac{\lambda_1}{\mu}(U_1 V_1^\top + W), \end{aligned}$$

where $W = \frac{\mu}{\lambda_1} P_2 \Sigma_2 D_2^\top$.

Thus it is easy to verify that

$$\begin{aligned} & Z - L = X - E - L \in \frac{\lambda_1}{\mu} \partial \|L\|_* \\ & = \left\{ \frac{\lambda_1}{\mu}(P_1 D_1^\top + W) | P_1^\top W = 0, W D_1 = 0, \|W\|_2 \leq 1 \right\}. \end{aligned} \tag{19}$$

The problem in (12) is equivalent to the following convex optimization problem

$$\begin{aligned} \min_{X,L,E} \frac{1}{2}\|Y - AX\|_F^2 + \frac{\lambda_1}{2}\|L\|_F^2 + \frac{\lambda_3}{2}\|X\|_F^2 + \lambda_2 \|E\|_1 \\ + \frac{\mu}{2}\|X - L - E\|_F^2 \end{aligned}$$

The first order optimal condition is satisfied by the obtained solution shown in (13), (16) and (19). Since the problem is convex, we can conclude that the solution is also global optimal. □

The results of Theorem 2 are directly implied from applying Theorem 2.1 of [25] with the convergence guarantee in Theorem 7.

IV. Simulations

We devote this section to simulation evaluation of the proposed OCRPCA algorithm on the synthetic data.

For benchmarking the performance of OCRPCA, we develop a batch counterpart of online compressed robust PCA by applying the ADM technique [26] to solve the problem (2). Here, instead of directly solving the complex optimization problem: $\min_{U,V,E,X} L(U, V, E, X, \beta)$, ADM separates the problem into four subproblems and solves them alternatively. Details of the batch compressed robust PCA algorithm are given in Algorithm 3.

Algorithm 3: Batch Compressed Robust PCA

```
Input : $Y$ (the observed data matrix), $\mathcal{A}, \mathcal{A}^\top$ (the compressed
        operator and its inverse operator), $\lambda_1, \lambda_2, \mu \in \mathbb{R}$
        (regularization parameters)
Initialize: $U_0$ is a random matrix, $E_0 = 0$, $V_0 = 0$, $\beta_0 = 0$,
$X_0 = \mathcal{A}(Y)$;
for $i$ = 0, 1, 2, ... do
    $V_{i+1} \leftarrow V_i, E_{i+1} \leftarrow E_i, X_{i+1} \leftarrow X_i$;
    while not converge do
        $V_{i+1} \leftarrow (\mu X_{i+1}^\top - \mu E_{i+1}^\top - \beta_i^\top) U (\lambda_1 I + \mu U_i^\top U_i)^{-1}$ ;
        $E_{i+1} \leftarrow \mathcal{S}_{\frac{2\lambda_2}{\mu}} (X_{i+1} - U_i V_{i+1}^\top - \beta_i/\mu)$ ;
        $X_{i+1} \leftarrow \arg\min_{X_{i+1}} \frac{1}{2}\|\mathcal{A}(X_{i+1}) - Y\|_F^2 +$
        $\frac{\mu}{2}\|X_{i+1} - (\beta_i/\mu + U_i V_{i+1}^\top + E_{i+1})_F^2 + \frac{\lambda_3}{2}\|X_{i+1}\|_F^2$;
    end
    $\beta_{i+1} \leftarrow \beta_i - \mu_k (X_{i+1} - U_i V_{i+1}^\top - E_{i+1})$;
    $U_{i+1} \leftarrow (\mu X_{i+1} - \mu E_{i+1} - \beta_{i+1}) V_{i+1} (\lambda_1 I + V_{i+1}^\top V_{i+1})^{-1}$
    ;
end
Output: $U, V, E$
```

We follow the data generation scheme of PCP as in [3] to generate our synthetic dataset. We first generate a low-rank matrix L as a product of U and V, which is $L = UV^\top$. Here the sizes of U and V are $d \times r$ and $T \times r$ respectively. r

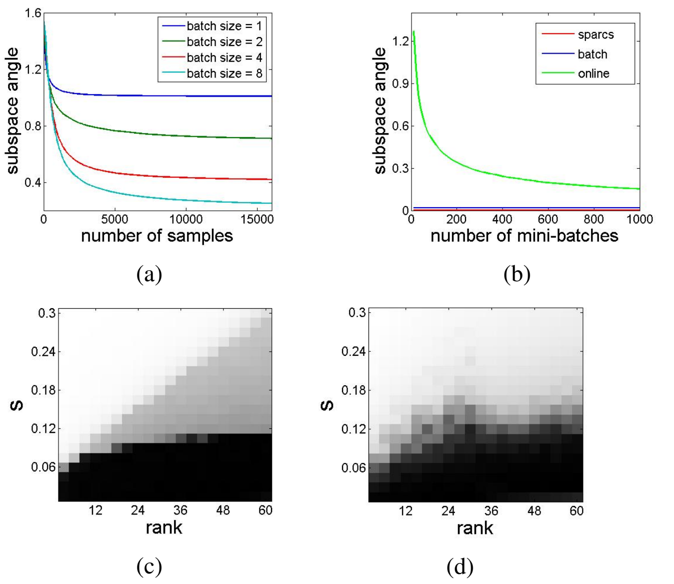

Fig. 1: (a) convergence curves of OCRPCA with different batch sizes. (b) convergence curve of the proposed OCRPCA. (c) and (d) show the performance of subspace recovering under different compression ratio s (vertical axis) and low-rank matrix rank (horizontal axis). (c) the result of batch compressed RPCA algorithm and (d) the result of OCRPCA. Brighter color means smaller subspace angle between the recovered one and the groundtruth.

is the intrinsic rank of the low-rank data matrix L, d is the dimension of each sample and T is the number of samples. U is the basis of the subspace spanned by the columns of L. The elements of both U and V are i.i.d. sampled from a normal distribution $\mathcal{N}(0, 1/T)$. The raw data before compression is then generated by $X = L + E$, where E is a sparse noise matrix with a ρ_s fraction of non-zero element. The non-zero elements in E is uniformly distributed over the interval of $[-10, 10]$. The observed compressed data is $Y = \mathcal{A}(X)$, where the compression operator $\mathcal{A}$ is the permuted noiselets proposed in [5]. A nice property of this compression operator is that we do not need to store any projection matrix explicitly. The compression operation is implemented by C++ to reduce both time and memory cost.

The performance is evaluated by the principal subspace angle θ [24] between the subspace recovered by Algorithm 1 and the groundtruth. More concretely, let U_T be the output basis by OCRPCA, and U be the basis of the groundtruth subspace. Then the subspace angle θ is calculated as

$$\theta = \arcsin(\min\{1, \|U_T - U(U^\top U_T)\|_F\}).$$

Smaller subspace angle indicates better recovery of the subspace. If the matrices U_T and U are not orthogonal ones, we first calculate the orthonormal basis of their range space through SVD, and then use the corresponding basis to calculate the subspace angle.

In the experiments, we empirically found that the convergence rate of Algorithm 1 was slow when it was only allowed to access one sample at each time instance. This is because a single sample carries very limited information when the compression ratio s is small. Thus, we improve Algorithm 1 by allowing it to process a mini-batch of samples at each time instance. The size of mini-batch n is controlled to be small and thus the extra memory cost is also light. The results in Figure 1a demonstrate the relationship between the size of mini-batch and the convergence rate of Algorithm 1. In the experiments, we use grid search to tune the parameters λ_1, λ_2 and λ_3. The results are obtained in the following settings: $d = 2048$, $r = 2$, $\rho_s = 0.01$, $s = 0.3$. The parameter λ_3 is fixed to be a small value 1×10^{-3}. The same experiments are performed for five times and the average is taken. We report corresponding subspace angles produced by OCRPCA with different mini-batch sizes when they see the same number of samples. The results demonstrate that slightly increasing the batch size from 1 to 4 significantly improves the convergence rate. For example, there is a large performance difference (around 0.8) after the algorithms see 5000 samples. Note that these empirical results on dependence of convergence rate on batch size do not conflict with our theoretical results in Theorem 2, as the theoretical guarantees hold for asymptotic cases.

Figure 1b illustrates how the performance of OCRPCA converges to the one of its batch counterpart when OCRPCA visits increasing number of mini-batches of very high-dimensional samples under the following setting: $n = 32$, $d = 2048$, $T = 1024$, $s = 0.3$, $r = 2$, $\rho_s = 0.01$. Here, we set the trade-off parameter of OCRPCA as: $\lambda_1 = 10$, $\lambda_2 = 0.01$, $\mu = 1$. We allow OCRPCA to revisit mini-batches to avoid introducing new samples. We run the batch algorithm on the same dataset as baseline with following parameter setting: $\lambda_1 = 0.1, \lambda_2 = 0.01, \mu = 1$. The result of the batch algorithm is plotted as the blue curve in Figure 1b. As another baseline, we also apply SpaRCS proposed in [23] on the same dataset. The final result is plotted as the red curve in the Figure 1b. We observe that along with OCRPCA visiting more mini-batches, the subspace angle smoothly decreases. This verifies the convergence of OCRPCA. On this dataset, both batch compressed RPCA and SpaRCS can correctly recover the subspace (the subspace angle is less than 0.1), while the subspace angle of the proposed OCRPCA gets really close to 0.1 after dealing with 1000 batches.

We also plot the averaged subspace angle of batch compressed RPCA and OCRPCA under different settings in a matrix form in Figure 1c and Figure 1d. All the results in Figure 1c and Figure 1d are produced under the following setting: $d = 256, T = 1024, n = 32$($n$ is set as a trade-off between the performance and overall convergence). Since there are only 1024 samples that are not sufficient for OCRPCA to converge, OCRPCA goes over all the mini-batches for 10 times without requiring new samples. This is also a common practice for implementing online learning algorithms. The fraction of non-zero elements in E is set as $\rho_s = 0.01$; the intrinsic rank r of the low-rank component varies from 3 to 60; the compression ratio s varies from very small 0.015 (a quite difficult case) to relatively large 0.3 (easier case). All the results in Figure 1c and Figure 1d are generated by averaging the results from implementing the algorithms for 5 times.

The results demonstrate that under relatively low intrinsic rank (*e.g.*, $r \leq 12$) and large compression ratio (*e.g.*, $s \geq 0.1$), OCRPCA can almost correctly recover the subspace (the subspace angle less than 0.1). From the plots, we can also observe that the performance of OCRPCA is very close to its batch counterpart. This verifies the guarantee in Theorem 2 that the performance of OCRPCA can converge to the one of the batch method, and both OCRPCA and batch method can provide estimates close to groundtruth under mild conditions. In some difficult settings for signal recovery (the very bottom of Figure 1c and Figure 1d, which means the compression ratio is very small), both the online algorithm and batch algorithm cannot correctly recover the subspace. The reason is that when the compression ratio is small, the observed compressed data cannot provide sufficient information due to the huge information loss.

We compare the efficiency of SpaRCS, batch compressed RPCA and OCRPCA in Table I. SpaRCS, PCP and batch compressed RPCA all have an $O(dT)$ factor in space cost. The dependency on T prevents them from dealing with big data. In contrast, OCRPCA only has a space complexity of $O(dr + dn + r^2)$ where $r \ll T, n \ll T$. Thus OCRPCA has the desired small space complexity to deal with big data.

TABLE I: The space complexity of SpaRCS, PCP, batch compressed RPCA and the proposed OCRPCA.

Algorithm	Space complexity
SpaRCS	$O(\rho_s dT + dT + dr + Tr)$
PCP	$O(dT)$
Batch compressed RPCA	$O(dT + dr + Tr)$
OCRPCA	$O(dr + dn + r^2)$

V. Conclusion

In this work, we developed an online compressed robust principal component analysis method that processed compressed big data directly. Different from former batch algorithms that need to record all the data, the proposed OCRPCA algorithm only needs to record a small batch (or even a single one) of the input sequential samples. Thus OCRPCA reduces the computation cost significantly and is very promising for dealing with big or dynamic data. Though we need to solve a non-convex optimization problem in OCRPCA, we provides an asymptotic guarantee – with a large probability, OCRPCA can successfully find the global optimum under mild conditions.

VI. Acknowledgement

This work was in part supported by the Data to Decisions Cooperative Research Centre www.d2dcrc.com.

References

[1] D. P. Bertsekas. Nonlinear programming. 1999.

[2] N. A. Campbell. Robust procedures in multivariate analysis i: Robust covariance estimation. *Applied statistics*, pages 231–237, 1980.

[3] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

[4] X. Chang, F. Nie, Y. Yang, and H. Huang. A convex sparse PCA for feature analysis. *ACM Transactions on Knowledge Discovery from Data*, 2016.

[5] R. Coifman, F. Geshwind, and Y. Meyer. Noiselets. *Applied and Computational Harmonic Analysis*, 10(1):27–44, 2001.

[6] F. De La Torre and M. J. Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54(1-3):117–142, 2003.

[7] D. Feldman, M. Monemizadeh, C. Sohler, and D. P. Woodruff. Coresets and sketches for high dimensional subspace approximation problems. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 630–649. Society for Industrial and Applied Mathematics, 2010.

[8] J. Feng, H. Xu, and S. Yan. Online robust pca via stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 404–412, 2013.

[9] H. Guo, C. Qiu, and N. Vaswani. An online algorithm for separating sparse and low-dimensional signal sequences from their sum. 2014.

[10] A. Haar. Der massbegriff in der theorie der kontinuierlichen gruppen. *Annals of mathematics*, pages 147–169, 1933.

[11] K. Lee and Y. Bresler. Admira: Atomic decomposition for minimum rank approximation. *Information Theory, IEEE Transactions on*, 56(9):4402–4416, 2010.

[12] E. Liberty, F. Woolfe, P.-G. Martinsson, V. Rokhlin, and M. Tygert. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 104(51):20167–20172, 2007.

[13] B. Lois and N. Vaswani. Online matrix completion and online robust pca. *arXiv preprint arXiv:1503.03525*, 2015.

[14] M. Luo, F. Nie, X. Chang, Y. Yang, A. Hauptmann, and Q. Zheng. Avoiding optimal mean robust pca/2dpca with non-greedy l1-norm maximization. In *International Joint Conference on Artificial Intelligence*, 2016.

[15] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, 11:19–60, 2010.

[16] D. Needell and J. A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

[17] F. Nie, H. Huang, C. Ding, D. Luo, and H. Wang. Robust principal component analysis with non-greedy l1-norm maximization. In *International Joint Conference on Artificial Intelligence*, volume 22, page 1433. Citeseer, 2011.

[18] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma. Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2233–2246, 2012.

[19] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[20] V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009.

[21] N. Srebro, T. Jaakkola, et al. Weighted low-rank approximations. In *International Conference on Machine Learning*, volume 3, pages 720–727, 2003.

[22] S. Wang, D. Liu, and Z. Zhang. Nonconvex relaxation approaches to robust matrix recovery. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 1764–1770. AAAI Press, 2013.

[23] A. E. Waters, A. C. Sankaranarayanan, and R. Baraniuk. Sparcs: Recovering low-rank and sparse matrices from compressive measurements. In *Advances in neural information processing systems*, pages 1089–1097, 2011.

[24] P. Å. Wedin. On angles between subspaces of a finite dimensional inner product space. In *Matrix Pencils*, pages 263–285. Springer, 1983.

[25] J. Wright, A. Ganesh, K. Min, and Y. Ma. Compressive principal component pursuit. *Information and Inference*, 2(1):32–68, 2013.

[26] X. Yuan and J. Yang. Sparse and low-rank matrix decomposition via alternating direction methods. *preprint*, 2009.

[27] H. Zhang, Z. Lin, C. Zhang, and E. Y. Chang. Exact recoverability of robust pca via outlier pursuit with tight recovery bounds. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[28] Z. Zhang, Y. Matsushita, and Y. Ma. Camera calibration with lens distortion from low-rank textures. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2321–2328. IEEE, 2011.