A FAST AND MEMORY-EFFICIENT ALGORITHM FOR ROBUST PCA (MEROP)

Praneeth Narayanamurthy and Namrata Vaswani

{pkurpadn, namrata} @iastate.edu,
Department of Electrical and Computer Engineering,
Iowa State University, Ames, IA

ABSTRACT

Robust PCA (RPCA) is the problem of separating a given data matrix into the sum of a sparse matrix and a low-rank matrix. We propose an algorithm, MERoP, (Memory-Efficient Robust PCA) based on the Recursive Projected Compressed Sensing (ReProCS) framework to solve the RPCA problem. We demonstrate that we can provably recover the low-rank and sparse components, and MERoP enjoys nearly-optimal memory complexity. We also show that the algorithm is nearly-online and fast. We validate our theoretical claims through extensive numerical experiments.

1. INTRODUCTION

Principal Components Analysis (PCA) is a widely used dimension reduction technique in a variety of scientific applications. Given a set of data vectors, PCA tries to finds a smaller dimensional subspace that best approximates a given dataset. According to its modern definition [1], robust PCA (RPCA) is the problem of decomposing a given data matrix into the sum of a low-rank matrix (true data) and a sparse matrix (outliers). The column space of the low-rank matrix then gives the desired principal subspace (PCA solution). In recent years, the RPCA problem has been extensively studied, e.g., [1, 2, 3, 4, 5, 6]. A common application of RPCA is in video analytics in separating video into a slow-changing background image sequence and a foreground image sequence consisting of moving objects or people [7]. In this work, we propose an algorithm called MERoP which is an online, and fast algorithm to solve the Robust PCA problem under weaker assumptions as compared to existing algorithms. In particular, we show that the running time of our algorithm is nearly the same as computing a vanilla SVD on the data matrix. Furthermore, we also show that the number of samples required to obtain an ε-accurate estimate of the subspace in which the true data lies is nearly-optimal.

Problem Statement. At each time t we observe data vectors $\boldsymbol{y}_t \in \mathbb{R}^n$ that satisfy

$$\boldsymbol{y}_t = \boldsymbol{\ell}_t + \boldsymbol{x}_t \tag{1}$$

where $\boldsymbol{x}_t$ is the sparse outlier vector and $\boldsymbol{\ell}_t$ is the true data vector that lies in a fixed or slowly changing low-dimensional subspace of $\mathbb{R}^n$. To be precise, $\boldsymbol{\ell}_t = \boldsymbol{P}\boldsymbol{a}_t$ where $\boldsymbol{P}$ is an $n \times r$ *basis matrix*[1] with $r \ll n$. Here and below, $'$ denotes matrix transpose and $\|\cdot\|$ refers to the l_2 norm of a vector or the induced l_2 norm of a matrix. We use $\mathcal{T}_t$ to denote the support set of $\boldsymbol{x}_t$ and assume that $|\mathcal{T}_t| \le s$ for all t. Define the $n \times d$ data matrix $\boldsymbol{Y} := [\boldsymbol{y}_1, \boldsymbol{y}_2, \dots \boldsymbol{y}_d] = \boldsymbol{L} + \boldsymbol{X}$ where $\boldsymbol{L}$, $\boldsymbol{X}$ are similarly defined.

Given an initial subspace estimate, $\hat{\boldsymbol{P}}_{\text{init}}$, the goal is to estimate $\operatorname{span}(\boldsymbol{P})$ within any given ε-accuracy quickly and provably. A by-product of doing this is that the true data vectors $\boldsymbol{\ell}_t$, the sparse outliers $\boldsymbol{x}_t$, and their support sets $\mathcal{T}_t$ can also be tracked on-the-fly. The initial subspace estimate, $\hat{\boldsymbol{P}}_{\text{init}}$, can be computed by applying any static (batch) RPCA technique, e.g., PCP [1] or AltProj [4], to the first t_{train} data frames, $\boldsymbol{Y}_{[1,t_{\text{train}}]}$. Here and below, $[a, b]$ refers to all integers between a and b, inclusive, $[a, b) := [a, b-1]$, and $\boldsymbol{M}_{\mathcal{T}}$ denotes a sub-matrix of $\boldsymbol{M}$ formed by columns indexed by entries in $\mathcal{T}$. For basis matrices $\hat{\boldsymbol{P}}$, $\boldsymbol{P}$ of rank r we use

$$\operatorname{dist}(\hat{\boldsymbol{P}}, \boldsymbol{P}) := \left(\sum_{i=1}^{r} \sin^2 \theta_i(\hat{\boldsymbol{P}}, \boldsymbol{P})\right)^{1/2}$$

to quantify the distance betwen them. Here, θ_i is the i-th principal angle[2], and can be computed as $\sin\theta_i(\hat{\boldsymbol{P}}, \boldsymbol{P}) = \sigma_i((\boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}')\boldsymbol{P})$. This is popularly known as the chordal distance [8]. This is one of the various metrics used to measure distances between subspaces.

Contributions. We propose an Algorithm, MERoP (Memory-Efficient Robust PCA) to solve the Robust-PCA problem based on the recently introduced Recursive Projected Compressive Sensing (ReProCS) framework [9]. This framework was initially to solve the dynamic Robust PCA problem: the problem in which the subspace is assumed to be changing with time. In this work, we show that adapting this technique, and using a coarse initialization we can solve the RPCA problem. The original result was a partial result: it assumed that the intermediate algorithm estimates satisfied certain "incoherence" assumptions. In subsequent work such as [10, 11, 12] a complete guarantee was obtained.

In this work, we extend this line of work to demonstrate that under mild assumptions, RPCA can be analyzed using the machinery that has been developed through this framework. In particular, we show that MERoP (i) has a running time of $O(ndr\log(1/\epsilon))$, which is the cost of performing a rank-r vanilla SVD on the data matrix; (ii) can tolerate an order wise larger fraction of corruptions/outlier per row under mild assumptions on the minimum outlier magnitude; and (iii) has nearly optimal-storage complexity: we need $O(nr\log n\log(1/\epsilon))$ samples to obtain an ϵ-accurate subspace estimate, which is only larger than the information-theoretic optimum of $O(nr)$ by logarithmic factors.

2. ALGORITHM AND MAIN RESULT

Algorithm 1 proceeds as follows. A coarse subspace estimate is obtained using PCP or AltProj applied to the first t_{train} frames

[1] tall matrix with mutually orthonormal columns

[2] When the subspaces $\hat{\boldsymbol{P}}$ and $\boldsymbol{P}$ are of the same dimesion, $\sin\theta_{\max}(\hat{\boldsymbol{P}}, \boldsymbol{P}) = \sin\theta_{\max}(\boldsymbol{P}, \hat{\boldsymbol{P}})$.

Algorithm 1 MERoP and Offline MERoP

```
**Input**: $\hat{\boldsymbol{P}}_0$, $\boldsymbol{y}_t$, **Output**: $\hat{\boldsymbol{x}}_t$, $\hat{\boldsymbol{\ell}}_t$, $\hat{\boldsymbol{P}}$
**Params:** $\omega_{supp}$, $K$, $\alpha$, $\xi$, $r$, $\omega_{evals}$
$\hat{\boldsymbol{P}}_{(t_{\text{train}})} \leftarrow \hat{\boldsymbol{P}}_0$; $k \leftarrow 1$.
**for** $t > t_{\text{train}}$ **do**
    $\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$;
    $\tilde{\boldsymbol{y}}_t \leftarrow \boldsymbol{\Psi}\boldsymbol{y}_t$.
    $\hat{\boldsymbol{x}}_{t,cs} \leftarrow \arg\min_{\tilde{\boldsymbol{x}}} \|\tilde{\boldsymbol{x}}\|_1$ s.t $\|\tilde{\boldsymbol{y}}_t - \boldsymbol{\Psi}\tilde{\boldsymbol{x}}\| \leq \xi$.
    $\hat{\mathcal{T}}_t \leftarrow \{i : |\hat{\boldsymbol{x}}_{t,cs}| > \omega_{supp}\}$.
    $\hat{\boldsymbol{x}}_t \leftarrow \boldsymbol{I}_{\hat{\mathcal{T}}_t}(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t})^{-1}\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\tilde{\boldsymbol{y}}_t$.
    $\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$.
    **if** $t = t_{\text{train}} + u\alpha$ for $u = 1,\ 2,\ \cdots,\ K$ **then**
        $\hat{\boldsymbol{P}}_k \leftarrow SVD_r[\hat{\boldsymbol{L}}_{t;\alpha}]$, $k \leftarrow k+1$.
    **else**
        $\hat{\boldsymbol{P}}_{(t)} \leftarrow \hat{\boldsymbol{P}}_{(t-1)}$
    **end if**
    **if** $t = t_{\text{train}} + K\alpha$ **then**
        $\hat{\boldsymbol{P}} \leftarrow \hat{\boldsymbol{P}}_k$
        $\boldsymbol{\Psi} \leftarrow \boldsymbol{I} - \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}'$
    **end if**
**end for**
**for** $t > t_{\text{train}}$ **do**                                   }
    $\hat{\boldsymbol{x}}_t \leftarrow \boldsymbol{I}_{\hat{\mathcal{T}}_t}(\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t})^{-1}\boldsymbol{\Psi}_{\hat{\mathcal{T}}_t}{}'\boldsymbol{y}_t$   }
    $\hat{\boldsymbol{\ell}}_t \leftarrow \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$.                  } Offline MERoP.
**end for**                                                    }
```

$\boldsymbol{Y}_{[1,t_{\text{train}}]}$. For $t > t_{\text{train}}$, the algorithm proceeds as follows. At time t, let $\hat{\boldsymbol{P}}_{t-1}$ denote the subspace estimate from $(t-1)$. If this estimate is accurate enough, projecting $\boldsymbol{y}_t := \boldsymbol{x}_t + \boldsymbol{\ell}_t$ onto its orthogonal complement will nullify most of $\boldsymbol{\ell}_t$. We compute $\tilde{\boldsymbol{y}}_t := \boldsymbol{\Psi}\boldsymbol{y}_t$ where $\boldsymbol{\Psi} := \boldsymbol{I} - \hat{\boldsymbol{P}}_{(t-1)}\hat{\boldsymbol{P}}_{(t-1)}{}'$. Thus, $\tilde{\boldsymbol{y}}_t = \boldsymbol{\Psi}\boldsymbol{x}_t + \boldsymbol{b}_t$ where $\boldsymbol{b}_t := \boldsymbol{\Psi}\boldsymbol{\ell}_t$ and $\|\boldsymbol{b}_t\|$ is small. Recovering $\boldsymbol{x}_t$ from $\tilde{\boldsymbol{y}}_t$ is thus a regular compressive sensing (CS) / sparse recovery problem in small noise [13]. We compute $\hat{\boldsymbol{x}}_{t,cs}$ using l_1 minimization followed by thresholding based support estimation to get $\hat{\mathcal{T}}_t$. A Least Squares (LS) based debiasing step on $\hat{\mathcal{T}}_t$ returns the final $\hat{\boldsymbol{x}}_t$. We then estimate $\boldsymbol{\ell}_t$ as $\hat{\boldsymbol{\ell}}_t = \boldsymbol{y}_t - \hat{\boldsymbol{x}}_t$. The $\hat{\boldsymbol{\ell}}_t$'s are used to update the subspace estimate. This is done using K steps of r-SVD, each done with a new set of α frames of $\hat{\boldsymbol{\ell}}_t$. Here r-SVD means compute the top r left singular vectors of $\hat{\boldsymbol{L}}_{t;\alpha} := [\hat{\boldsymbol{\ell}}_{t-\alpha+1}, \hat{\boldsymbol{\ell}}_{t-\alpha+2}, \ldots, \hat{\boldsymbol{\ell}}_t]$.

Using the result of [14] we show that with high probability, the subspace estimation error decreases exponentially after every α frames, and thus, after K-SVD steps, we obtain an ε-accurate estimate of the subspace. This is then used to re-estimate all the previous $\hat{\boldsymbol{x}}_t$'s and $\hat{\boldsymbol{\ell}}_t$'s which can also shown to be ε-accurate estimates.

Assumption on principal subspace coefficients $\boldsymbol{a}_t$. We assume that the $\boldsymbol{a}_t$'s are zero mean, mutually independent, and *element-wise bounded* random variables (r.v.) with diagonal covariance matrix $\boldsymbol{\Lambda}$. Since the $\boldsymbol{a}_t$'s are element-wise bounded, there exists an $\eta < \infty$, such that $\max_{j=1,2,\ldots r} \max_t \frac{(\boldsymbol{a}_t)_j^2}{\lambda_j(\boldsymbol{\Lambda})} \leq \eta$. For most bounded distributions, η is a little more than one, e.g., if the entries of $\boldsymbol{a}_t$ are zero mean uniform, then $\eta = 3$. The bounded-ness assumption on the $\boldsymbol{a}_t$'s is similar to the right singular vectors' incoherence assumption needed by all the other RPCA solutions [3, 4]. There are minor differences since we impose statistical assumptions on the $\boldsymbol{a}_t$'s.

Incoherence left singular vectors of $\boldsymbol{L}$. In order to separate the $\boldsymbol{\ell}_t$'s from the sparse outliers $\boldsymbol{x}_t$, we need to assume that the $\boldsymbol{\ell}_t$'s are themselves not sparse (thus we sometimes refer to this property as "denseness".). This is ensured if we can assume that column vectors of $\boldsymbol{P}$ are dense enough. To quantify this, we define μ as the smallest real number that satisfies

$$\max_{i=1,2,\ldots,n} \|\boldsymbol{I}_i{}'\boldsymbol{P}\| \leq \sqrt{\frac{\mu r}{n}} \tag{2}$$

The above assumption is similar to the left incoherence assumption needed by all the other RPCA solutions [3, 4].

Bound on outlier fractions. Similar to earlier RPCA works, we also need outlier fractions to be bounded. However, we need different bounds on this fraction per column and per row. The row bound can be much larger. Let max-outlier-frac-col $:= \max_t |\mathcal{T}_t|/n$ denotes the maximum outlier fraction in any column of $\boldsymbol{Y}$. Because MERoP is an online algorithm we need the fraction of outliers per row of a sub-matrix of $\boldsymbol{Y}$ with α consecutive columns to be bounded. To quantify this, for a time interval, $\mathcal{J}$, define

$$\gamma(\mathcal{J}) := \max_{i=1,2,\ldots,n} \frac{1}{|\mathcal{J}|} \sum_{t \in \mathcal{J}} \mathbf{1}_{\{i \in \mathcal{T}_t\}}. \tag{3}$$

where $\mathbf{1}_S$ is the indicator function for event S. Thus $\gamma(\mathcal{J})$ is the maximum outlier fraction in any row of the sub-matrix $\boldsymbol{Y}_{\mathcal{J}}$ of $\boldsymbol{Y}$. Let $\mathcal{J}^\alpha$ denote a time interval of duration α. We will bound

$$\text{max-outlier-frac-row} := \max_{\mathcal{J}^\alpha \subseteq [t_{\text{train}}, d]} \gamma(\mathcal{J}^\alpha). \tag{4}$$

Initialization. Assume that the initial data, $\boldsymbol{Y}_{[1,t_{\text{train}}]}$, satisfies PCP(H) [3] or AltProj [4] assumptions: (i) let $\boldsymbol{L}_{\text{init}} \overset{\text{SVD}}{=} \boldsymbol{P}\boldsymbol{\Sigma}\boldsymbol{V}'$; $\boldsymbol{P}$ and $\boldsymbol{V}$ satisfy incoherence with parameter μ, and (ii) outlier fractions per row and per column are both upper bounded by $c/(\mu r)$; and, (iii) we run enough iterations of AltProj so that the output subspace estimate satisfies $\sin\theta_{\max}(\hat{\boldsymbol{P}}_{\text{init}}, \boldsymbol{P}) \leq 0.05/\sqrt{r}$.

2.1. Main Result

We use λ^+ and λ^- to denote the maximum and minimum eigenvalues of $\boldsymbol{\Lambda}$ and let $f := \lambda^+/\lambda^-$ denote the condition number.

Theorem 2.1 (RPCA)**.** *Consider the data matrix $\boldsymbol{Y} = \boldsymbol{L} + \boldsymbol{X}$ where each matrix is of size $n \times d$ and let $\operatorname{rank}(\boldsymbol{L}) = r$. Pick an $\varepsilon_{\text{dist}} > 0$ and set $\varepsilon = \varepsilon_{\text{dist}}/\sqrt{r}$. For $t \geq t_{\text{train}}$, assume*

1. *assumptions on $\boldsymbol{a}_t$'s hold,*
2. *The initial assumptino on $\hat{\boldsymbol{P}}_{init}$ holds,*
3. $0.35\sqrt{\eta\lambda^+} \leq x_{\min}/15$,
4. *max-outlier-frac-row $\leq b_0/f^2$ where $b_0 = 0.02$, and max-outlier-frac-col $\leq 0.09/(\mu r)$,*
5. *algorithm parameters are set as $K = c\log(1/\varepsilon_{\text{dist}})$, $\alpha = Cf^2(r\log n)$, $\xi = x_{\min}/15$, $\omega_{supp} = x_{\min}/2$*

Then, with probability at least $1 - 10dn^{-10}$, the output of Algorithm 1 satisfies $\sin\theta_{\max}(\hat{\boldsymbol{P}}, \boldsymbol{P}) \leq \varepsilon$, $\|\hat{\boldsymbol{\ell}}_t - \boldsymbol{\ell}_t| \leq \varepsilon\|\boldsymbol{\ell}_t\|$ and $\hat{\mathcal{T}}_t = \mathcal{T}_t$ for all t.

Remark 2.2. *The lower bound on $x_{\min}$ seems counter-intuitive since small magnitude corruptions should not be problematic. With simple changes to the proof of Theorem 2.1, it is possible to show the more intuitive result stated here. Define $\zeta_k := 0.3^k(\Delta_{dist} + \varepsilon)$. Let $\mathcal{J}_k := [t_{\text{train}} + (k-1)\alpha, t_{\text{train}} + k\alpha)$.*

1. *Pick $\varepsilon < \min_t \min_{i \in \mathcal{T}_t} |(\boldsymbol{x}_t)_i|/30$*

Table 1: Comparing assumptions, time and memory complexity. For simplicity, we ignore all dependence on condition numbers.

Algorithm	Outlier tolerance, rank of $(\boldsymbol{L})$	Assumptions	Memory, Time,	# params.
PCP(C)[1] (offline)	max-outlier-frac-row $= O(1)$ max-outlier-frac-col $= O(1)$ $r \leq \frac{c \min(n,d)}{\log^2 n}$	strong incoh, unif. rand. support,	Mem: $O(nd)$ Time: $O(nd^2 \frac{1}{\epsilon})$	zero
AltProj[4], (offline)	max-outlier-frac-row $= O\left(1/r\right)$ max-outlier-frac-col $= O\left(1/r\right)$		Mem: $O(nd)$ Time: $O(ndr^2 \log \frac{1}{\epsilon})$	2
RPCA-GD [5] (offline)	max-outlier-frac-row $= O(1/r^{3/2})$ max-outlier-frac-col $= O(1/r^{3/2})$		Mem: $O(nd)$ Time: $O(ndr \log \frac{1}{\epsilon})$	5
PG-RMC [6] (offline)	max-outlier-frac-row $= O\left(1/r\right)$ max-outlier-frac-col $= O(1/r)$	$d = O(n)$	Mem: $O(nd)$ Time: $O(nr^3 \log n \log \frac{1}{\epsilon})$	4
MERoP **(this work)** (online and offline)	**max-outlier-frac-row** $= O(1)$ **max-outlier-frac-col** $= O(1/r)$	$\boldsymbol{a}_t$'s independent, init data: AltProj assu's, outlier mag. lower bounded	**Mem:** $O(nr \log n \log \frac{1}{\epsilon})$ **Time:** $O(ndr \log \frac{1}{\epsilon})$	4

The table assumes an $n \times d$ data matrix $\boldsymbol{Y} := \boldsymbol{L} + \boldsymbol{X}$, where $\boldsymbol{L}$ has rank r and the outlier matrix $\boldsymbol{X}$ is sparse. It compares ReProCS for solving the original robust PCA problem with other methods for solving the same problem. Thus the subspace that ReProCS recovers is also of dimension $r = r$. With only a mild extra assumption on outlier magnitudes, ReProCS is able to achieve a significant gain in outlier tolerance per row. We also note that all the algorithms require *left and right* incoherence, and thus we do not list this in the third column.

2. *Use the following to replace (5). For $t \in \mathcal{J}_k$, define the $\tilde{\mathcal{T}}_t := \{i : |(\boldsymbol{x}_t)_i| > 30\zeta_k \sqrt{\eta\lambda^+}\}$. Define $\tilde{\boldsymbol{x}}_t := \boldsymbol{I}_{\tilde{\mathcal{T}}_t} \boldsymbol{I}_{\tilde{\mathcal{T}}_t}{}' \boldsymbol{x}_t$. Let $\boldsymbol{v}_t := \boldsymbol{x}_t - \tilde{\boldsymbol{x}}_t$. Assume that $\|\boldsymbol{v}_t\| \leq \zeta_k \sqrt{\eta\lambda^+}$. (Thus $\tilde{\boldsymbol{x}}_t$ contains the entries of $\boldsymbol{x}_t$ whose magnitude is larger than $30\zeta_k \sqrt{\eta\lambda^+}$. These are the "real" outliers while $\boldsymbol{v}_t$ contains the smaller magnitude corruptions. Our assumption requires that the small magnitude corruptions are small enough so that $15(\zeta_k + \|\boldsymbol{v}_t\|)$ is smaller than the minimum non-zero entry of $\tilde{\boldsymbol{x}}_t$.*

3. *Assume that $\tilde{\mathcal{T}}_t$ satisfies the max-outlier-frac-row and max-outlier-frac-col bounds (Item 4 of Theorem 2.1).*

Then, with the same probability of success, the output of Algorithm 1 satisfies all the conditions of Theorem 2.1 with $\tilde{\mathcal{T}}_t$ replaced with $\mathcal{T}_t$.

3. DISCUSSION

There is a vast body of literature analyzing RPCA. We compare with some of the existing algorithms and note that this is by no means an exhaustive list. One of the first works to provide guarantees for RPCA were [1, 2] who studied a convex relaxation to recover the sparsest $\boldsymbol{X}$ and the "least-rank" $\boldsymbol{L}$ using the l_1 norm, and the nuclear norm, respectively. These works offer an elegant theory, however, in practice, they are very slow. To obtain an ϵ-accurate solution for an $n \times d$ matrix, the time complexity is $O(nd^2/\epsilon)$. Since then, there has been a rich development of faster algorithms to attack this problem. In particular, AltProj [4] was one of the first algorithms to study a non-convex version of the Robust PCA problem. Their method relied on starting with an initial estimate of the sparse outliers, and then alternately projecting the "residuals" on to the highly non-convex sets of sparse, and. low-rank matrices. To circumvent the problem of sensitivity of the Singular Value Decomposition (SVD) the algorithm proceeds in stages; incrementing the "target rank" at each stage. This is significantly faster than the convex approaches and enjoyed a run-time of $O(ndr^2 \log(1/\epsilon))$. Following this, there were attempts to further improve the running time of the algorithms. In particular, the goal is to design an algorithm that runs in "optimal" time, i.e., that is equal to that of an SVD. A recent work, [5] showed that it is indeed possible to design such an algorithm, and demonstrated that their algorithm runs in time $O(ndr \log(1/\epsilon))$. However, this approach required a very stringent bound on the allowed outlier-fractions. However, there are two scenarios which these algorithms do not consider, (i) what is the minimum number of samples needed to guarantee an ϵ-accurate solution?, and (ii) is it possible to work with streaming samples, and not requiring to store the entire $n \times d$ data matrix in memory? These are the questions that our algorithm address, and we will show that we achieve near-optimal memory complexity, and also show that it is possible to work without storing the entire matrix in memory.

Another related work is the recently proposed PG-RMC [6], which is the fastest exisiting batch algorithm for the Robust PCA problem. The reason it is the fastest is that it randomly selects a few entries of the data matrix, and thus needs to work with a fewer number of samples. In particular, the run-time of this algorithm is $O(nr^3 \log(1/\epsilon))$. This is significantly faster than all the existing algorithms, however, the disadvantage of this approach is that (i) one cannot recover the sparse matrix due to the uniform random sampling of the data matrix and (ii) it requires $d = O(n)$ which is a stringent requirement. We provide a detailed discussion in Table 1.

Solutions to RPCA in the online setting has received considerable interest too. Firstly, there are algorithms based on the GRASTA framework which use stochastic Gradient Descent to obtain the subspace estimate, and sparse recovery alternately [15, 16]. Although very fast, this framework does not offer any theoretical guarantees. Another approach is the recent "online-reformulation" of PCP [17], based on stochastic optimization techniques. This offered a partial guarantee: it requires that the intermediate algorithms are full rank, and illustrates asymptotic convergence. We mention this because they do not obtain a precise relation to the minimum number of samples needed for a good reconstruction. Very recently, a streaming algorithm using the ideas of AltProj [4] – but replacing SVD by a block-stochastic power-method was proposed in [18]. The advan-

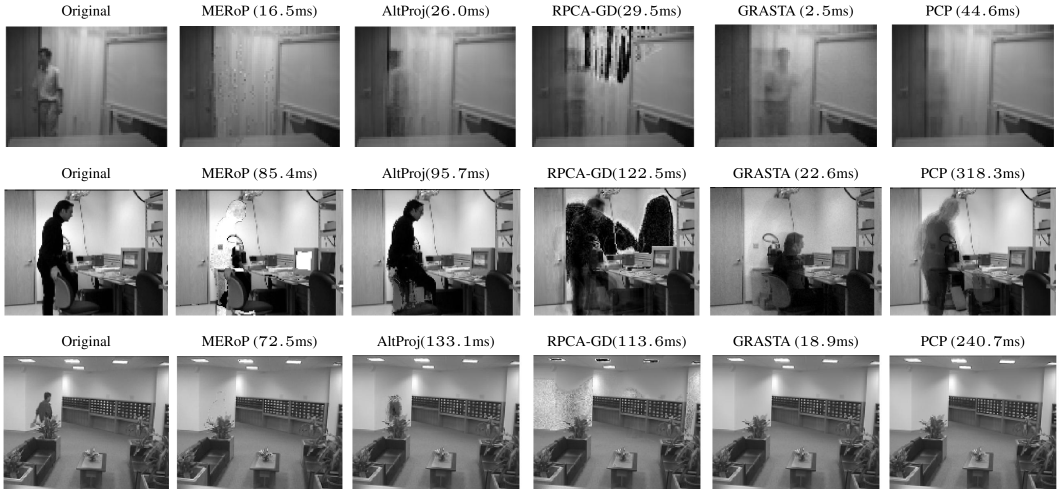

Fig. 1: Background recovery. For MR and SL sequences (first two rows), only MERoP background does not contain the person or even his shadow. All others do. Also MERoP is faster than all except GRASTA. For LB, MERoP is as good as PCP and GRASTA, while others fail. Time taken per frame is shown in parentheses.

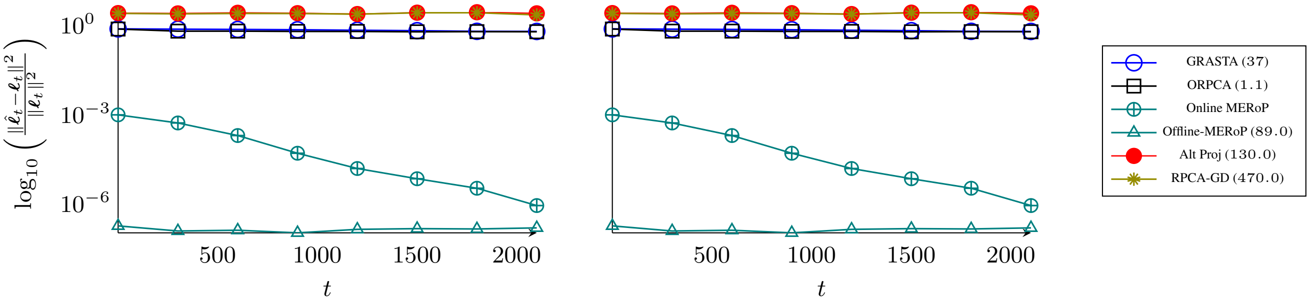

Fig. 2: Relative error in recovering $\boldsymbol{\ell}_t$'s. Left: Moving object model on $\mathcal{T}_t$. Right: Bernoulli model on $\mathcal{T}_t$. Time taken per frame in milliseconds (ms) for the Bernoulli model is shown in parentheses in the legend. The errors are plotted every $k\alpha - 1$ time-frames. Observe the nearly-exponential decay of the subspace error with time.

tage of such an approach is that the algorithm needs only a single pass over the data samples. However, the guarantees only include the case of one-dimensional RPCA, i.e., $r = 1$.

4. EMPIRICAL EVALUATION

In this section we present the results of numerical experiments to compare the performance of MERoP with existing state-of-the-art algorithms on synthetic data and in the task of Foreground-Background separation in real videos. All experiments are performed on a Desktop Computer with Intel® Xeon E3-1240 8-core CPU @ 3.50GHz and 32GB RAM.

Synthetic Data. We perform an experiment on synthetic data to demonstrate the superiority of MEDRoP over existing algorithms. We generate the data as follows. $\boldsymbol{P}_0$ is generated by ortho-normalizing the columns of an $n \times r$ i.i.d standard normal matrix. We used $n = 1000$, $r = 30$, $d = 3000$. For the low-rank matrix $\boldsymbol{L}$ we generate the coefficients $\boldsymbol{a}_t \in \mathbb{R}^r$ according to $(\boldsymbol{a}_t)_i \overset{i.i.d}{\sim} Unif[-q_i, q_i]$ where $q_i = \sqrt{f} - \sqrt{f}i/2r$ for $i = 1, 2, \cdots, r-1$ and $q_r = 1$. thus the condition number is f and we selected $f = 50$. We used the first $t_{\text{train}} = 300$ frames as the training part, where we generated a smaller fraction of outliers. For the moving object model (see Appendix [12, Model G.24]) with parameters $s/n = 0.01$, $b_0 = 0.01$ and for $t > t_{\text{train}}$ we used $s/n = 0.05$ and $b_0 = 0.3$. For the Bernoulli model we set $\rho = 0.01$ for the first t_{train} frames and $\rho = 0.3$ for the subsequent frames. The sparse outlier magnitudes are generated uniformly at random from the interval $[x_{\min}, x_{\max}]$ with $x_{\min} = 10$ and $x_{\max} = 20$ in both experiments. The results are averaged over 50 independent trials. The results are shown in Fig. 2.

We initialized MERoP and s-ReProCS [12] using AltProj [4] applied to $\boldsymbol{Y}_{[1,t_{\text{train}}]}$. The smaller outlier fraction helped achieve $\sin\theta_{\max}(\hat{\boldsymbol{P}}_{\text{init}}, \boldsymbol{P}_0) \approx 10^{-3}$. For the batch methods used in the comparisons – PCP, AltProj and RPCA-GD, we implement the algorithms on $\boldsymbol{Y}_{[1,t]}$. Further, we set the regularization parameter for PCP $1/\sqrt{n}$ in accordance with [1]. The other known parameters, r for Alt-Proj, outlier-fraction for RPCA-GD, are set using the true values. For online methods we implement ORPCA by [17] and GRASTA by [19]. The regularization parameter for ORPCA was set as with $\lambda_1 = 1/\sqrt{n}$ and $\lambda_2 = 1/\sqrt{d}$ according to [17].

Video Experiments. We also illustrate the efficacy of MERoP on real videos in this section. In particular, we implement several algorithms on three datasets, `MR` (Meeting Room), `SL` (Switch Light) and `LB` (Lobby) which are benchmark datasets in background separation [7]. We show one recovered background frame for each video in Fig. 1. All algorithms used $r = 40$ and default parameters in their code. ReProCS used $\alpha = 60$, $K = 3$, $\xi_t = \|\boldsymbol{\Psi}\hat{\boldsymbol{\ell}}_{t-1}\|$, $\omega_{supp} = \|\boldsymbol{y}_t\|/\sqrt{n}$, $\omega_{evals} = 0.011\lambda^-$. ORPCA failed completely, gave a black background. Hence it is not shown. Time taken per frame is shown above each image.

5. REFERENCES

[1] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?," *Journal of ACM*, vol. 58, no. 3, 2011.

[2] V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky, "Rank-sparsity incoherence for matrix decomposition," *SIAM Journal on Optimization*, vol. 21, 2011.

[3] D. Hsu, S.M. Kakade, and T. Zhang, "Robust matrix decomposition with sparse corruptions," *IEEE Trans. Info. Th.*, Nov. 2011.

[4] P. Netrapalli, U N Niranjan, S. Sanghavi, A. Anandkumar, and P. Jain, "Non-convex robust pca," in *Neural Info. Proc. Sys. (NIPS)*, 2014.

[5] Xinyang Yi, Dohyung Park, Yudong Chen, and Constantine Caramanis, "Fast algorithms for robust pca via gradient descent," in *Neural Info. Proc. Sys. (NIPS)*, 2016.

[6] Yeshwanth Cherapanamjeri, Kartik Gupta, and Prateek Jain, "Nearly-optimal robust matrix completion," *arXiv preprint arXiv:1606.07315*, 2016.

[7] Florian Seidel, Clemens Hage, and Martin Kleinsteuber, "pROST: a smoothed\ell _p-norm robust online subspace tracking method for background subtraction in video," *Machine vision and applications*, vol. 25, no. 5, pp. 1227–1240, 2014.

[8] Ke Ye and Lek-Heng Lim, "Schubert varieties and distances between subspaces of different dimensions," *SIAM Journal on Matrix Analysis and Applications*, vol. 37, no. 3, pp. 1176–1197, 2016.

[9] C. Qiu, N. Vaswani, B. Lois, and L. Hogben, "Recursive robust pca or recursive sparse recovery in large but structured noise," *IEEE Trans. Info. Th.*, pp. 5007–5039, August 2014.

[10] B. Lois and N. Vaswani, "Online matrix completion and online robust pca," in *IEEE Intl. Symp. Info. Th. (ISIT)*, 2015.

[11] J. Zhan, B. Lois, H. Guo, and N. Vaswani, "Online (and Offline) Robust PCA: Novel Algorithms and Performance Guarantees," in *Intnl. Conf. Artif. Intell. and Stat. (AISTATS)*, 2016.

[12] P. Narayanamurthy and N. Vaswani, "New Results for Provable Dynamic Robust PCA," *arXiv:1705.08948*, 2017.

[13] E. Candes, "The restricted isometry property and its implications for compressed sensing," *Compte Rendus de l'Academie des Sciences, Paris, Serie I*, pp. 589–592, 2008.

[14] N. Vaswani and P. Narayanamurthy, "PCA in Data-Dependent Noise: Nearly Optimal Finite Sample Guarantees," *arXiv:1702.03070*, 2017.

[15] Laura Balzano, Robert Nowak, and Benjamin Recht, "Online identification and tracking of subspaces from highly incomplete information," *arXiv:1012.1086v3*, 2010.

[16] J. He, L. Balzano, and A. Szlam, "Incremental gradient on the grassmannian for online foreground and background separation in subsampled video," in *IEEE Conf. on Comp. Vis. Pat. Rec. (CVPR)*, 2012.

[17] J. Feng, H. Xu, and S. Yan, "Online robust pca via stochastic optimization," in *Adv. Neural Info. Proc. Sys. (NIPS)*, 2013.

[18] UN Niranjan and Yang Shi, "Streaming robust pca," 2016.

[19] Laura Balzano, Robert Nowak, and Benjamin Recht, "Online identification and tracking of subspaces from highly incomplete information," in *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*. IEEE, 2010, pp. 704–711.

A. PROOF OF MAIN RESULT

In this section we provide the proof of Theorem 2.1. To prove this, we first state two main Lemmas which are used in the proof. We provide the proof of the Lemmas in the long version. We use the following definitions in our proof

1. $g_k = (0.3)^k \Delta_{\text{init}}$
2. $\Gamma_0 = \sin\theta_{\max}(\hat{\boldsymbol{P}}_{\text{init}}, \boldsymbol{P}) \leq \Delta_{\text{init}}/\sqrt{r}$
3. $\Gamma_k = \{\Gamma_{k-1} \cap \sin\theta_{\max}(\hat{\boldsymbol{P}}_k, \boldsymbol{P}) \leq g_k)\}$

Lemma A.3 (MERoP First Subspace Update)**.** *Under the conditions of Theorem 2.1, conditioned on* Γ_0

1. *For all* $t \in [t_{\text{train}}, t_{\text{train}} + \alpha)$*, the error* $\boldsymbol{e}_t = \hat{\boldsymbol{x}}_t - \boldsymbol{x}_t = \boldsymbol{\ell}_t - \hat{\boldsymbol{\ell}}_t$ *satisfies*
$$\boldsymbol{e}_t = \boldsymbol{I}_{\mathcal{T}_t}\left(\boldsymbol{\Psi}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\right)^{-1}\boldsymbol{I}_{\mathcal{T}_t}{}'\boldsymbol{\Psi}_{\mathcal{T}_t}\boldsymbol{\ell}_t, \tag{5}$$
and $\|\boldsymbol{e}_t\| \leq 1.2\varepsilon_{\text{dist}}\sqrt{\eta\lambda^+}$.
2. *With probability at least* $1 - 10n^{-10}$ *the subspace estimate* $\hat{\boldsymbol{P}}_1$ *satisfies* $\sin\theta_{\max}(\hat{\boldsymbol{P}}_1, \boldsymbol{P}) \leq g_1$*, i.e.,* Γ_1 *holds*

Lemma A.4 (MERoP k-th Subspace Update)**.** *Under the conditions of Theorem 2.1, conditioned on* $\Gamma_{j,k-1}$

1. *For all* $t \in [t_{\text{train}} + (k-1)\alpha, t_{\text{train}} + k\alpha)$*, the error* $\boldsymbol{e}_t = \hat{\boldsymbol{x}}_t - \boldsymbol{x}_t = \boldsymbol{\ell}_t - \hat{\boldsymbol{\ell}}_t$ *satisfies* (5) *and for this interval,* $\|\boldsymbol{e}_t\| \leq (0.3)^{k-1} \cdot 1.2\varepsilon_{\text{dist}}\sqrt{\eta\lambda^+}$.
2. *With probability at least* $1 - 10n^{-10}$ *the subspace estimate* $\hat{\boldsymbol{P}}_k$ *satisfies* $\sin\theta_{\max}(\hat{\boldsymbol{P}}_k, \boldsymbol{P}) \leq (0.3)^k g_1$*, i.e.,* Γ_k *holds.*

The two crucial ideas that are used to prove Lemmas A.3 and A.4 are (i) Using the idea of [9] to relate the order s-Restricted Isometry Constant of the projection matrix, $\boldsymbol{\Psi}$ to the left-incoherence property as

$$\delta_s(\boldsymbol{I} - \boldsymbol{P}\boldsymbol{P}') = \max_{|\mathcal{T}| \leq s} \|\boldsymbol{I}_{\mathcal{T}}{}'\boldsymbol{P}\|^2 \leq s\|\boldsymbol{I}_i{}'\boldsymbol{P}\|^2 \tag{6}$$

To illustrate briefly, we show that the matrix $\boldsymbol{I} - \hat{\boldsymbol{P}}_{\text{init}}\hat{\boldsymbol{P}}_{\text{init}}{}'$ satisfies the $2s$-RIP using (6) and $\sin\theta_{\max}(\hat{\boldsymbol{P}}_{\text{init}}, \boldsymbol{P}) \leq \Delta_{\text{init}}$, followed by the left-incoherence assumption on $\boldsymbol{P}$ and obtain $\delta_{2s}(\boldsymbol{I} - \hat{\boldsymbol{P}}_{\text{init}}\hat{\boldsymbol{P}}_{\text{init}}{}') \leq 0.15$. A similar idea can be used to show that the RIC of all subsequent matrices are small constants. This allows us to get precise bounds on the reconstruction error of $\hat{\boldsymbol{x}}_t$, and subsequently guarantee exact support recovery. At this point, estimating (or updating) the $\hat{\boldsymbol{P}}_k$ is a problem of PCA in sparse data dependent noise and we use the second important idea of [14] which provides finite sample guarantees for the same.

Proof of Theorem 2.1. Notice from the definitions that if we show $\Pr(\Gamma_K|\Gamma_0) \geq 1 - dn^{-10}$ we are done. Also note from the definitions that $\Gamma_K \subseteq \Gamma_{K-1} \subseteq \cdots \subseteq \Gamma_0$ and thus,

$$\begin{aligned}\Pr(\Gamma_K|\Gamma_0) &= \Pr(\Gamma_K, \Gamma_{K-1}, \cdots, \Gamma_1|\Gamma_0) \\ &= \prod_{k=1}^{K} \Pr(\Gamma_k|\Gamma_{k-1}) \overset{(a)}{\geq} (1 - 10n^{-10})^K \\ &\geq 1 - 10dn^{-10}\end{aligned}$$

where (a) used Lemmas A.3 and A.4. □