A CORRECTNESS RESULT FOR ONLINE ROBUST PCA

Brian Lois and Namrata Vaswani

Iowa State University, Ames, IA, USA
Email: {blois,namrata}@iastate.edu

ABSTRACT

We study the problem of sequentially recovering a sparse vector $\boldsymbol{x}_t$ and a vector from a low-dimensional subspace $\boldsymbol{\ell}_t$ from knowledge of their sum $\boldsymbol{m}_t = \boldsymbol{x}_t + \boldsymbol{\ell}_t$. If the primary goal is to recover the low-dimensional subspace where the $\boldsymbol{\ell}_t$'s lie, then the problem is one of online or recursive robust principal components analysis (PCA). To the best of our knowledge, this is the first correctness result for this problem. We prove that if a good estimate of the initial subspace is available; the $\boldsymbol{\ell}_t$'s obey certain denseness and slow subspace change assumptions; and the support of $\boldsymbol{x}_t$ changes either at every frame or at least every so often, then with high probability, the support of $\boldsymbol{x}_t$ will be recovered exactly, and the error made in estimating $\boldsymbol{x}_t$ and $\boldsymbol{\ell}_t$ will be small. An example where this problem occurs is in separating a sparse foreground and a slowly changing dense background from surveillance videos.

1. INTRODUCTION

Principal Components Analysis (PCA) is a widely used tool for dimension reduction. As is well known, the standard PCA approach (computing SVD of the data matrix) is highly sensitive to outliers. A common way to model outliers is as sparse vectors [2]. In seminal papers Candès et. al. and Chandrasekaran et. al. introduced the Principal Components Pursuit (PCP) program and proved its robustness to sparse outliers [3], [4]. Later work by Hsu et. al. [5] improved the result of [4]. Since then, there has been much later work on obtaining guarantees for robust PCA, e.g. [6, 7, 8, 9] and many others, but all of it has been for batch methods.

In this work we consider an online or recursive version of the robust PCA problem where we seek to separate vectors into low dimensional and sparse components as they arrive, using the previous estimates, rather than re-solving the entire problem at each time t. An application where this type of problem is useful is in video analysis [10]. Imagine a video sequence that has a distinct background and foreground. An example might be a surveillance camera where a person walks across the scene. If the background does not change very much, and the foreground is sparse (both practical assumptions), then separating the background and foreground can be viewed as a robust PCA problem. In this and many other applications, e.g. sensor networks based detection of outlier events such as forest fires, network anomaly detection, or other streaming video analytics problems, an online solution is desirable.

Contributions. To the best of our knowledge, this is among the first works that provides a correctness result for an online (recursive) algorithm for sparse plus low-rank matrix recovery. We study the ReProCS algorithm introduced in [11]. As shown in [12], with practical heuristics used to set its parameters, ReProCS has significantly improved recovery performance compared to other recursive and even batch methods for many simulated and real video datasets.

We show that as long as algorithm parameters are set appropriately (which requires knowledge of subspace change model parameters), a good-enough estimate of the initial subspace is available, slow subspace change holds, the subspaces are dense enough, and there is a certain amount of support change at least every so often, then the support can be exactly recovered with high probability; the sparse and low-rank matrix columns can be recovered with bounded and small error; and the subspace recovery error decays to a small value within a short delay of a subspace change.

Online algorithms are needed for real-time applications; and even for offline applications, they are faster and need less storage compared to batch techniques. Moreover, online approaches can provide a natural way to exploit temporal dependencies in the dataset. In our case, we show that ReProCS uses slow subspace change to allow for significantly more correlated support sets of the sparse vectors than do the various results for PCP [3, 4, 5]. Of course this advantage comes at a cost. We need a tighter bound on the rank-sparsity product compared to [3] and some extra but practically valid assumptions (see Sec 6).

Finally, we also develop new proof techniques to prove our results. A brief discussion is provided in Sec 5.

Partial results have been provided for online sparse plus low-rank matrix recovery in [11]; and also in later work by Feng et. al. [13]; however, all require an assumption on intermediate algorithm estimates. We discuss these and [14, 15] in Sec 6. There is some more recent work on online robust PCA algorithms and their experimental evaluation, e.g. [16].

Notation. We use lowercase bold letters for vectors, cap-

Longer version of this paper is under submission to IEEE Trans. Info. Th. [1]. This work was supported by NSF grant CCF-1117125.

ital bold letters for matrices, and calligraphic capital letters for sets. We use $\boldsymbol{x}'$ for the transpose of $\boldsymbol{x}$. The 2-norm of a vector and the induced 2-norm of a matrix are denoted by $\|\cdot\|_2$. We refer to a matrix with orthonormal columns as a *basis matrix*. Notice that for a basis matrix $\boldsymbol{P}$, $\boldsymbol{P}'\boldsymbol{P} = \boldsymbol{I}$. For a set $\mathcal{T}$ of integers, $|\mathcal{T}|$ denotes its cardinality. For a vector $\boldsymbol{x}$, $\boldsymbol{x}_{\mathcal{T}}$ is a vector containing the entries of $\boldsymbol{x}$ indexed by $\mathcal{T}$. Define $\boldsymbol{I}_{\mathcal{T}}$ to be the matrix of those columns of the identity matrix indexed by $\mathcal{T}$. We use the interval notation $[a, b]$ to mean all of the integers between a and b, inclusive.

2. PROBLEM DEFINITION AND ASSUMPTIONS

At time t we observe a vector $\boldsymbol{m}_t \in \mathbb{R}^n$ that is the sum of a vector from a slowly changing low-dimensional subspace $\boldsymbol{\ell}_t$ and a sparse vector $\boldsymbol{x}_t$. So

$$\boldsymbol{m}_t = \boldsymbol{\ell}_t + \boldsymbol{x}_t \qquad \text{for } t = 0, 1, 2, \ldots, t_{\max},$$

We model the low-dimensional $\boldsymbol{\ell}_t$'s as $\boldsymbol{\ell}_t = \boldsymbol{P}_t \boldsymbol{a}_t$ for a basis matrix $\boldsymbol{P}_t$ that is allowed to change slowly over time. Given an estimate of the initial subspace $\hat{\boldsymbol{P}}_{(0)}$, the goal is to obtain estimates $\hat{\boldsymbol{x}}_t$ and $\hat{\boldsymbol{\ell}}_t$ at each time t and to periodically update the estimate of the subspace $\operatorname{span}(\boldsymbol{P}_t)$.

2.1. Model on $\boldsymbol{\ell}_t$

1. Subspace Change Model for $\boldsymbol{\ell}_t$

   Let t_j for $j = 1, \ldots, J$ be the times at which the subspace where the $\boldsymbol{\ell}_t$'s lie changes. We assume $\boldsymbol{\ell}_t = \boldsymbol{P}_t \boldsymbol{a}_t$ where $\boldsymbol{P}_t = \boldsymbol{P}_{(j)}$ for $t_j \leq t < t_{j+1}$. $\boldsymbol{P}_{(j)}$ is a basis matrix that changes as $\boldsymbol{P}_{(j)} = [\boldsymbol{P}_{(j-1)} \ \boldsymbol{P}_{(j),\text{new}}]$. Then $\boldsymbol{a}_t$ can be split as $\boldsymbol{a}_t = [{\boldsymbol{a}_{t,*}}' \ {\boldsymbol{a}_{t,\text{new}}}']'$. Let $r_j = \operatorname{rank}(\boldsymbol{P}_{(j)})$ and define $r := r_J = \max_j \operatorname{rank} \boldsymbol{P}_{(j)}$. Also let $c_{j,\text{new}} = \operatorname{rank}(\boldsymbol{P}_{(j),\text{new}})$ and define $c := \max_j \operatorname{rank}(\boldsymbol{P}_{(j),\text{new}})$.

   Assume that $r < \min\{n, t_{j+1} - t_j\}$ for all j.

2. Assumptions and notation for $\boldsymbol{a}_t$

   We assume that the $\boldsymbol{a}_t$'s are zero mean bounded random variables that are mutually independent over time. Let

   $$\gamma := \sup_t \|\boldsymbol{a}_t\|_\infty \quad \text{and} \quad \gamma_{\text{new}} := \sup_t \|\boldsymbol{a}_{t,\text{new}}\|_\infty.$$

   Define $\boldsymbol{\Lambda}_t := \operatorname{Cov}(\boldsymbol{a}_t)$ and assume it is diagonal. Let $(\boldsymbol{\Lambda}_t)_{\text{new}} := \operatorname{Cov}(\boldsymbol{a}_{t,\text{new}})$. Define $\lambda^- := \inf_t \lambda_{\min}(\boldsymbol{\Lambda}_t)$ and $\lambda^+ := \sup_t \lambda_{\max}(\boldsymbol{\Lambda}_t)$, and assume that $0 < \lambda^- \leq \lambda^+ < \infty$. Also, for an integer d, define

   $$\lambda_{\text{new}}^- := \min_j \min_{t \in [t_j, t_j + d]} \lambda_{\min}((\boldsymbol{\Lambda}_t)_{\text{new}})$$

   $$\lambda_{\text{new}}^+ := \max_j \max_{t \in [t_j, t_j + d]} \lambda_{\max}((\boldsymbol{\Lambda}_t)_{\text{new}}).$$

   Then define

   $$f := \frac{\lambda^+}{\lambda^-} \quad \text{and} \quad g := \frac{\lambda_{\text{new}}^+}{\lambda_{\text{new}}^-}.$$

2.2. Model on $\boldsymbol{x}_t$

Let $\mathcal{T}_t := \{i : (\boldsymbol{x}_t)_i \neq 0\}$ be the support set of $\boldsymbol{x}_t$ and let $s := \max_t |\mathcal{T}_t|$ be the size of the largest support. Let $x_{\min} := \inf_t \min_{i \in \mathcal{T}_t} |(\boldsymbol{x}_t)_i|$ denote the size of the smallest non-zero entry of any $\boldsymbol{x}_t$.

Assume one of the following two models on support change of $\boldsymbol{x}_t$. The first model is for an object of length s or less that moves with probability q and remains stationary with probability $1 - q$ at each time instant independent of all other times. Also, when it moves, it moves by $\frac{s}{\varrho}$ plus small random acceleration, ν_t for a constant $\varrho \geq 1$. The second model is for an object of length s that moves a little at each time. These are two special cases that are our result can handle. For the most general case, see [1, Section III].

Model 2.1. *Consider one-dimensional motion of the support of $\boldsymbol{x}_t$, and let o_t be its center at time t. Suppose that the support moves according to the model*

$$o_t = o_{t-1} + \theta_t \left(1.1 \frac{s}{\varrho} + \nu_t\right) \tag{1}$$

where ν_t is Gaussian $\mathcal{N}(0, \sigma^2)$ and θ_t is a Bernoulli random variable that takes the value 1 with probability q and 0 with probability $1 - q$, and $\varrho \geq 1$ is a constant. Assume that $\{\nu_t\}$, $\{\theta_t\}$ are mutually independent and independent of $\{\boldsymbol{a}_t\}$ for $t = 1, \ldots, t_{\max}$.

The above model allows the object to change in size over time as long as its *center* moves by the required amount and its size is bounded by s.

Model 2.2. *Suppose that the support of $\boldsymbol{x}_t$ is of a constant size s, consists of consecutive indices, and moves in a given direction by between 1 and a indices at every time t.*

Remark 2.3. *In both models, when the object reaches index n, it can change direction and move up until it reaches index 1, where it is reflected back downward again. Or, a new object can appear at index 1 or n after the first has left the scene.*

2.3. Subspace Denseness

Definition 2.4. *For a basis matrix $\boldsymbol{P}$, define $\kappa_s(\boldsymbol{P}) := \max_{|\mathcal{T}| \leq s} \|{\boldsymbol{I}_{\mathcal{T}}}'\boldsymbol{P}\|_2$. As described in [11],* small *κ_s means that the columns of $\boldsymbol{P}$ are dense vectors.*

Our result needs an upper bound on $\kappa_{2s}(\boldsymbol{P}_{(J)})$ and $\kappa_{2s}(\boldsymbol{P}_{(j),\text{new}})$.

3. MAIN RESULT

In this section we state and discuss our main result for the ReProCS algorithm introduced in [11, Algorithm 1]. We do not repeat the algorithm here due to lack of space. Its main idea is briefly explained next.

Main idea of the ReProCS algorithm [11, Algorithm 1]. Given an accurate estimate of the subspace where the $\boldsymbol{\ell}_t$'s lie, projecting the measurement $\boldsymbol{m}_t = \boldsymbol{x}_t + \boldsymbol{\ell}_t$ onto the orthogonal complement of the estimated subspace will nullify most of $\boldsymbol{\ell}_t$. The denseness of $\boldsymbol{\ell}_t$ implies that this projection will have a small restricted isometry constant [11]. Thus basis pursuit denoising (BPDN) applied to the projected measurements will produce an accurate estimate $\hat{\boldsymbol{x}}_t$ [17]. Then, subtraction also gives a good estimate $\hat{\boldsymbol{\ell}}_t = \boldsymbol{m}_t - \hat{\boldsymbol{x}}_t$. Using these $\hat{\boldsymbol{\ell}}_t$'s, the algorithm successively updates the subspace estimate by a modification of the standard PCA procedure, which we call projection PCA. The algorithm uses knowledge of t_j, $c_{j,\text{new}}$, r_0, and γ_{new}.

Theorem 3.1. *Consider Algorithm 1 of [11]. Assume the model given in Sec. 2. Pick a ζ that satisfies*

$$\zeta \leq \min\left(\frac{10^{-4}}{r^2}, \frac{1.5\times 10^{-4}}{r^2 f}, \frac{1}{r^3\gamma^2}\right)$$

and suppose that $t_{\max} \leq n^{10}$. If

1. *The algorithm parameters are set as: $K = \left\lceil \frac{\log(0.17c\zeta)}{\log(0.72)} \right\rceil$; $\xi = \sqrt{c}\gamma_{\text{new}} + \sqrt{\zeta}(\sqrt{r}+\sqrt{c})$; $7\xi \leq \omega \leq x_{\min} - 7\xi$; $\alpha = C(\log(6KJ) + 11\log(n))$ for a constant $C \geq C_{\text{add}} := \frac{4800}{(\zeta\lambda^-)^2}\max\{16, (1.2\xi)^4\}$*
2. *$\|(\boldsymbol{I} - \hat{\boldsymbol{P}}_{(0)}\hat{\boldsymbol{P}}_{(0)}{}')\boldsymbol{P}_{(0)}\|_2 \leq r_0\zeta$;*
3. *The subspace changes slowly enough such that*
   - *$t_{j+1} - t_j > d \geq K\alpha$ for all j;*
   - *$\sqrt{c}\gamma_{\text{new}} + \sqrt{\zeta}(\sqrt{r}+\sqrt{c}) \leq \frac{x_{\min}}{14}$;*
   - *$g \leq \sqrt{2}$;*
4. *The low dimensional subspace is dense such that $\kappa_{2s}(\boldsymbol{P}_{(J)}) \leq 0.3$; and $\max_j \kappa_{2s}(\boldsymbol{P}_{(j),\text{new}}) \leq 0.02$.*
5. *The support set of $\boldsymbol{x}_t$ changes enough so that either*
   - *Model 2.1 holds with $\sigma^2 \leq \dfrac{s^2}{4000\varrho^2\log(n)}$; $q \geq 1 - \left(\frac{n^{-10}}{2(t_{\max}+\alpha)}\right)^{\frac{50\varrho^2}{\alpha}}$; and $s \leq \frac{n}{2\alpha}$; or*
   - *Model 2.2 holds with $s \leq \frac{\alpha}{400}$; and $\alpha \leq \frac{n}{a}$*

Then, with probability at least $1 - n^{-10}$, at all times t, the support of $\boldsymbol{x}_t$ is recovered exactly, i.e. $\hat{\mathcal{T}}_t = \mathcal{T}_t$.

Corollary 3.2. *Under the above assumptions, the recovery error satisfies: $e_t := \hat{\boldsymbol{x}}_t - \boldsymbol{x}_t = \boldsymbol{\ell}_t - \hat{\boldsymbol{\ell}}_t$ satisfies $\|e_t\|_2 \leq 1.2\left(1.83\sqrt{\zeta} + (0.72)^{k-1}\sqrt{c}\gamma_{\text{new}}\right)$ when $t \in [t_j + (k-1)\alpha, t_j + k\alpha - 1]$, $k = 1, 2, \dots, K$ and $\|e_t\|_2 \leq 2.4\sqrt{\zeta}$ when $t \in [t_j + K\alpha, t_{j+1} - 1]$.*

The subspace error $\text{SE}_t := \|(\boldsymbol{I} - \hat{\boldsymbol{P}}_t\hat{\boldsymbol{P}}_t{}')\boldsymbol{P}_t\|_2$ *satisfies:* $\text{SE}_t \leq 10^{-2}\sqrt{\zeta} + 0.72^{k-1}$ *when* $t \in [t_j + (k-1)\alpha, t_j + k\alpha - 1]$, $k = 1, 2, \dots, K$ *and* $\text{SE}_t \leq 10^{-2}\sqrt{\zeta}$ *when* $t \in [t_j + K\alpha, t_{j+1} - 1]$.

Proof: For the proof, see [1].

4. SIMULATION EXPERIMENT

Figure 1 is shows the results of a simulation experiment that demonstrates Theorem 3.1 and Corollary 3.2. Data was generated to satisfy the assumptions of the theorem. For details, see [1, Section VIII]. The batch method PCP was performed every α time instants using all of measurements up to that point. Since the support of $\boldsymbol{x}_t$ changed in a highly correlated fashion, it resulted in the matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_{t_{\max}}]$ being also very low rank. Because of this, the PCP recovery error is large. The ReProCS error is much smaller and decays exponentially with each projection PCA step (as shown by Corollary 3.2).

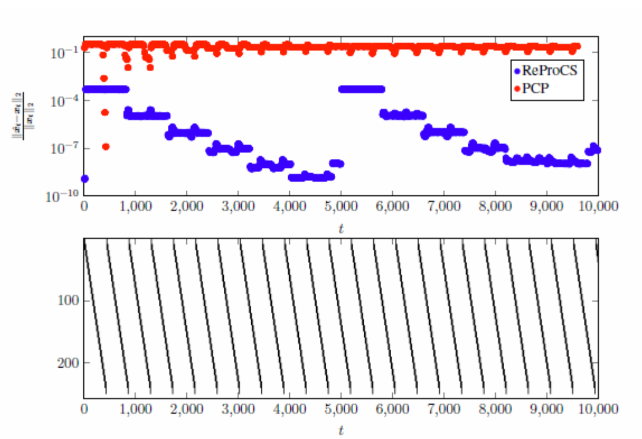

Fig. 1. Recovery error (top) Support pattern of $\boldsymbol{X}$ (bottom)

5. NOVELTY OF PROOF TECHNIQUES

Our proof uses the overall framework of [11], but we need a new approach to analyze the subspace estimate update step in order to remove the assumption on intermediate algorithm estimates used by the result of [11]. The key new idea is to leverage the fact that, because of exact support recovery, the error $\boldsymbol{e}_t := \hat{\boldsymbol{x}}_t - \boldsymbol{x}_t = \boldsymbol{\ell}_t - \hat{\boldsymbol{\ell}}_t$ is supported on $\mathcal{T}_t$. Also, our support change model ensures that $\mathcal{T}_t$ changes at least every so often. Together, this ensures that the matrix $[\boldsymbol{e}_{t_j+(k-1)\alpha}, \boldsymbol{e}_{t_j+(k-1)\alpha+1}, \dots, \boldsymbol{e}_{t_j+k\alpha-1}]$ is a block banded matrix with only $2\varrho + 1$ bands.

This work and the earlier work on which this is based [11] need new proof techniques because, as explained in the introduction, all existing correctness results for this problem are only for batch methods. Moreover, our proof cannot just be a combination of a sparse recovery result and a result for PCA, because in the PCA step for ReProCS, the error between $\hat{\boldsymbol{\ell}}_t$ and $\boldsymbol{\ell}_t$ is correlated with $\boldsymbol{\ell}_t$ (this is because $\hat{\boldsymbol{\ell}}_t = \boldsymbol{m}_t - \hat{\boldsymbol{x}}_t$ and the error in $\hat{\boldsymbol{x}}_t$ depends on the projection of $\boldsymbol{\ell}_t$ into the space perpendicular to $\hat{\boldsymbol{P}}_t$). But almost all existing work on finite sample PCA assumes that the error between the measured and true data vectors is uncorrelated with the true data, see e.g. [18] and references therein.

6. DISCUSSION

The result needs accurate initial subspace knowledge (easy to a obtain using a short training sequence of background-only video data), a slow subspace change assumption, a support change assumption and a denseness assumption.

Consider the subspace change model. This model (along with the bound on γ_{new} from the theorem) assumes that after a subspace change, $\|\boldsymbol{a}_{t,\text{new}}\|_\infty$ and therefore also $\|(\boldsymbol{\Lambda}_t)_{\text{new}}\|_2$ are initially small. After $t_j + d$, the eigenvalues of $(\boldsymbol{\Lambda}_t)_{\text{new}}$ are allowed to increase up to λ^+. Thus a new direction added at time t_j can have variance as large as λ^+ by $t_j + d$ and definitely by the next subspace change time since $t_{j+1} \geq t_j + d$. As demonstrated in [11], this slow subspace change assumption is valid for backgrounds in real video sequences.

Consider the support change models. Both Models 2.1 and 2.2 are valid and commonly used models for foreground object motion in videos. If we assume Model 2.1, our result requires $s \leq \frac{n}{2\alpha}$. If $J \leq C_1 \log n$ for some constant C_1, then using the definition of α, this bound holds if $s \leq C_2 \frac{n}{\log n}$. If $r_0 \leq C_3 \log n$ for a constant C_3, we get that $r \leq C_4 \log n$. Thus, this model allows $s \in O(\frac{n}{\log n})$ and $r \in O(\log n)$. As we explain next, these bounds on s and r also satisfy our denseness assumption.

Consider denseness. The way κ_s is defined, our denseness assumption simultaneously places restrictions on denseness of $\boldsymbol{\ell}_t$, and on r and s. As done in [3], we could assume $\kappa_1(\boldsymbol{P}_{(J)}) \leq \sqrt{\frac{\mu r}{n}}$, where μ is any value between 1 and $\frac{n}{r}$. It is easy to show that $\kappa_s(\boldsymbol{P}) \leq \sqrt{s}\kappa_1(\boldsymbol{P})$ [11]. Thus if $\frac{2sr}{n} \leq \mu^{-1}(0.3)^2$, then our assumption of $\kappa_{2s}(\boldsymbol{P}_{(J)}) \leq 0.3$ will be satisfied. Clearly the bounds on s and r from above ensure this up to appropriate choice of constants.

Comparison with other work. The above requirement on s and r is stronger than that used by [3] (which studies the batch approach PCP). There s is allowed to grow linearly with n, and r is simultaneously allowed to grow as $\frac{n}{\log(n)^2}$. But, up to differences in the constants, the above is same as the requirement found in [19] (which also studies the PCP program and is an improvement over [4]), except that [19] does not need specific bounds on s and r. The comparison is not direct though because our result does not need denseness of the right singular vectors of $\boldsymbol{L}$ or a bound on the vector infinity norm of $\boldsymbol{U}\boldsymbol{V}'$, while [3, 4, 19] do. Here $\boldsymbol{L} = [\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_{t_{\max}}] \overset{\text{SVD}}{=} \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}'$. The reason for our stronger requirement on sr is because we study an online algorithm, ReProCS, that recovers the sparse vector $\boldsymbol{x}_t$ at each time t rather than in a batch or a piecewise batch fashion. Because of this, the sparse recovery step does not use the low dimensionality of the new (and still unestimated) subspace.

Because we only require that the support changes after a given maximum allowed duration, it can be constant for a certain period of time (Model 2.1), or it can change only a little at each time (Model 2.2). This is a substantially weaker assumption than the independent or uniformly random supports required by [3] and [15]. As we explain in [1], if we consider the whole matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t_{\max}}]$, then at most $\frac{t_{\max}}{5000}$ non-zero entries per row are allowed. Thus, for $r > 5000$, this also is a significant improvement over [19] which requires at most $\frac{t_{\max}}{r}$ non-zero entries per row. Therefore, an important advantage of our result is that it allows for highly correlated support sets of $\boldsymbol{x}_t$, which is important for applications such as video surveillance that involve one or more moving foreground objects or persons forming the sparse vector $\boldsymbol{x}_t$.

Now consider works that also use initial subspace knowledge. Our result improves upon [11]'s results by removing the denseness requirements on $(\boldsymbol{I} - \boldsymbol{P}_{(j),\text{new}}\boldsymbol{P}_{(j),\text{new}}')\hat{\boldsymbol{P}}_{(j),\text{new},k}$ and $(\boldsymbol{I} - \hat{\boldsymbol{P}}_{(j-1)}\hat{\boldsymbol{P}}_{(j-1)}' - \hat{\boldsymbol{P}}_{(j),\text{new},k}\hat{\boldsymbol{P}}_{(j),\text{new},k}')\boldsymbol{P}_{(j),\text{new}}$ and thus providing a complete correctness result. In [13], Feng et. al. propose a method for online robust PCA and prove a partial result for their algorithm. The approach is to reformulate the PCP program and use this reformulation to develop a recursive algorithm that converges asymptotically to the solution of PCP as long as the basis estimate $\hat{\boldsymbol{P}}_t$ is full rank at each time t. Since this result assumes something about the algorithm estimates, it is only a *partial* result. Another work of Feng et. al. [14] on online robust PCA does not model the outlier as a sparse vector but defines anything that is far from the data subspace as an outlier. Another recent work that uses knowledge of the initial subspace estimate is modified-PCP [15]. However, like PCP, this also needs uniformly random supports. Moreover it is a piecewise batch approach.

Limitations and Ongoing Work. An important limitation of our result is that we analyze an algorithm that needs knowledge of subspace change model parameters (t_j, $c_{j,\text{new}}$, r_0, γ_{new}) which is not true of other algorithms such as PCP. *We should point out though that it does not assume any knowledge of the support change model.* The most limiting assumptions are knowing t_j and $c_{j,\text{new}}$. In ongoing work, we are able to remove this requirement. Another assumption we need is the zero mean and independence of the $\boldsymbol{\ell}_t$'s over time. If a mean background image (obtained by averaging an initial sequence of background only training data) is subtracted from all measurements, then zero mean is valid. Moreover, if background variation is due to small and random illumination changes, then independence is also valid (or close to valid). In ongoing work, we are able to remove this and instead allow for a more realistic autoregressive model on the $\boldsymbol{\ell}_t$'s.

Our subspace change model only allows for adding new directions to the subspace. This is a valid model if, at time t, the goal is to estimate the column span of the matrix $\boldsymbol{L}_t := [\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots, \boldsymbol{\ell}_t]$, which is the goal in robust PCA. However, when $\boldsymbol{x}_t$ is the quantity of interest and $\boldsymbol{\ell}_t$ is the large but structured noise, this model can be restrictive. A better model would be one that also allows removal of directions from $\boldsymbol{P}_{(j)}$, e.g., Model 7.1 of [11]. This significantly relaxes the required denseness assumption, and is being done in ongoing work.

A fundamental limitation of our analysis approach is the assumption that subspace changes every so often.

7. REFERENCES

[1] B. Lois and N. Vaswani, “A correctness result for online robust pca,” *submitted to IEEE Trans. Info. Th., arXiv:1409.3959 [cs.IT]*, 2014.

[2] J. Wright and Y. Ma, “Dense error correction via l1-minimization,” *IEEE Trans. on Info. Th.*, vol. 56, no. 7, pp. 3540–3560, 2010.

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

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

[5] D. Hsu, S. M. Kakade, and T. Zhang, “Robust matrix decomposition with outliers,” *arXiv:1011.1518*.

[6] Y. Chen, A. Jalali, S. Sanghavi, and C. Caramanis, “Low-rank matrix recovery from errors and erasures,” in *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 2313–2317.

[7] M. McCoy and J. Tropp, “Two proposals for robust pca using semidefinite programming,” *arXiv:1012.1086v3*, 2010.

[8] T. Zhang and G. Lerman, “A novel m-estimator for robust pca,” *arXiv:1112.4863v1*, 2011.

[9] H. Xu, C. Caramanis, and S. Sanghavi, “Robust pca via outlier pursuit,” *IEEE Tran. on Information Theorey*, vol. 58, no. 5, 2012.

[10] F. D. L. Torre and M. J. Black, “A framework for robust subspace learning,” *International Journal of Computer Vision*, vol. 54, pp. 117–142, 2003.

[11] 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.*, Aug. 2014, shorter versions in ICASSP 2013 and ISIT 2013.

[12] H. Guo, C. Qiu, and N. Vaswani, “An online algorithm for separating sparse and low-dimensional signal sequences from their sum,” *IEEE Trans. Sig. Proc.*, Aug. 2014.

[13] J. Feng, H. Xu, and S. Yan, “Online robust pca via stochastic optimization,” in *Advances in Neural Information Processing Systems 26*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 404–412. [Online]. Available: http://papers.nips.cc/paper/5131-online-robust-pca-via-stochastic-optimization.pdf

[14] J. Feng, H. Xu, S. Mannor, and S. Yan, “Online pca for contaminated data,” in *Advances in Neural Information Processing Systems 26*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 764–772. [Online]. Available: http://papers.nips.cc/paper/5135-online-pca-for-contaminated-data.pdf

[15] J. Zhan and N. Vaswani, “Robust pca with partial subspace knowledge,” *arXiv:1403.1591 [cs.IT]*.

[16] J. He, L. Balzano, and J. Lui, “Online robust subspace tracking from partial information,” *arXiv:1109.3827 [cs.IT]*.

[17] E. Candès, “The restricted isometry property and its implications for compressed sensing,” *Compte Rendus de l’Academie des Sciences, Paris, Serie I*, pp. 589–592, 2008.

[18] B. Nadler, “Finite sample approximation results for principal component analysis: A matrix perturbation approach,” *The Annals of Statistics*, vol. 36, no. 6, 2008.

[19] D. Hsu, S. M. Kakade, and T. Zhang, “Robust matrix decomposition with sparse corruptions,” *Information Theory, IEEE Transactions on*, vol. 57, no. 11, pp. 7221–7234, 2011.