Selective Erasures for High-Dimensional Robust Subspace Tracking

Daniel L. Pimentel-Alarcón

Georgia State University, Atlanta, GA, USA

ABSTRACT

This paper presents an online method to track a subspace $\mathcal{U}$ from severely corrupted and incomplete data. If we could identify the corrupted entries in a new observation $\mathbf{x}$, then we would be able to update $\mathcal{U}$ according to the uncorrupted entries in $\mathbf{x}$ using an incomplete-data rank-one update. The challenge is to identify the corrupted entries in $\mathbf{x}$, which is in general NP-hard. To work around this we propose an approach that iteratively removes the entries that most affect partial projections of $\mathbf{x}$ onto $\mathcal{U}$. Our experiments show that this simple approach outperforms state-of-the-art methods, including ℓ_1-optimization, specially when most entries in $\mathbf{x}$ are corrupted.

Keywords: Missing data, matrix completion, compressed sensing, sparse.

1. INTRODUCTION

Subspace tracking aims to continuously estimate a subspace $\mathcal{U}$ that changes over time. At each time we observe a new data vector $\mathbf{x}$, and the goal is to update $\mathcal{U}$ accordingly. The challenge is that $\mathbf{x}$ often has missing values and corrupted entries (outliers).

For example, suppose you want to distinguish between background and foreground in a video. The background can be modeled as a low-dimensional subspace $\mathcal{U}$ that slowly evolves over time (illumination changes, small camera variations, etc.). At each time we observe a new video frame (image), and want to update the background based on this new image $\mathbf{x}$. However, some pixels in $\mathbf{x}$ may correspond to foreground objects, which are outliers from the background. In addition, processing the whole image can be time consuming. Fortunately, recent theory on missing data shows that a subspace can be estimated from highly incomplete data. So, for real-time video surveillance it is often better to subsample $\mathbf{x}$ (inducing missing data). To summarize, we want to constantly update the background subspace $\mathcal{U}$ according to a subsampled image $\mathbf{x}$ that is corrupted with outliers.

Background segmentation is just one example of the applicability of robust subspace tracking from incomplete data [1–3]. Similar examples arise in medical imaging [4], recommender systems [5, 6], communications [7], anomaly detection [8, 9], hyperspectral imaging [10] and target localization [11], among many others [12].

Further author information: pimentel@gsu.edu

Motivated by these applications, the last years have seen a wide range of approaches to subspace estimation and tracking, for example GRASTA [1], ROSETA [2], optical flow orientations [3], kernels [9, 10] and ESPRIT [11], to name a few.

The current state-of-the-art approaches [1, 2] use convex relaxation techniques to identify the inlier entries in $\mathbf{x}$, and then uses those inliers to do an incomplete-data rank-one subspace update (move in the direction of the gradient on the Grassmann manifold [13]). Typically, these approaches require that the subspace $\mathcal{U}$ has low coherence (parameter indicating how aligned a subspace is to the canonical axes), and that the outliers are sparse, uniformly located, and unstructured (for example, not lying in an other subspace).

These are reasonable assumptions in many applications, but certainly not all. For example, in many microscopy and astronomy videos (where most pixels are dark) the background subspace is highly coherent. In some cases, like traffic monitoring or video surveillance (where foreground objects like cars or pedestrians only take a small fraction of each frame), outliers are indeed sparse. However, in many other scenarios this is not the case. For example, if the camera is close to a foreground object (e.g., in a selfie), then the foreground may take most of the fame, producing an image predominantly composed of outliers. Furthermore, the location, color and shape of foreground objects are strongly correlated over time. This results in non-uniformly located and highly structured outliers.

These limitations are not specific to computer vision applications. For another example, consider recommender systems. You have a collection of users and items (movies, restaurants, shoes, etc.). The goal is to predict which items a user would like in order to make good recommendations. The preferences of some users can be explained with a subspace $\mathcal{U}$; the preferences of other users can be explained with an other subspace $\mathcal{V}$ (these subspaces may or may not be coherent). Now suppose that you observe a vector $\mathbf{x}$ containing mixed information (ratings) from two users that share the same account (as is often the case, for example, in Netflix), and you want to update $\mathcal{U}$ and $\mathcal{V}$ accordingly. The entries of $\mathbf{x}$ corresponding to $\mathcal{V}$ are outliers with respect to $\mathcal{U}$, and vice-versa. In other words, the ratings of one user become the outliers of an other. This implies that (depending on the number of users, and how many items each rates) most entries in $\mathbf{x}$ may be outliers for each subspace. Furthermore, these outliers are highly structured (they lie in an other subspace). In addition, the items that each user rates tend to be highly correlated (a child is more likely to rate children movies), and hence the outliers (ratings of other users) will not be uniformly distributed. As we will see, existing methods tend to fail in scenarios like these.

Motivated by this, we propose a new approach for robust subspace tracking from incomplete data. In contrast to existing algorithms, our method can handle a large number of arbitrarily located and highly structured outliers, and even highly coherent subspaces.

Notice that the main challenge is to determine which entries of $\mathbf{x}$ are outliers. After this, we can update $\mathcal{U}$ according to the inliers using an incomplete-data rank-one update, as in [13]. Hence, we will focus on identifying outliers. The main insight behind our approach is that outlier entries often increase the ratio between the norm of $\mathbf{x}$ and its projection onto $\mathcal{U}$. Hence, we will search for all such entries and remove them (hence the name *erasure*), until we end with a sub-vector of $\mathbf{x}$ whose norm is equal (or close) to its projection (indicating that all the remaining

entries are inliers). This process may remove some inliers along the way. However, as we will see, finding a few inliers is enough to find them all.

Our experiments show that our approach outperforms the current state-of-the-art algorithms, including ℓ_1-minimization, specially when $\mathbf{x}$ has more outliers than inliers.

Organization of the paper

In Section 2 we give the formal setup of the problem and our assumptions. In Section 3 we introduce our *erasure* algorithm. We present all our experiments in Section 4.

2. MODEL AND ASSUMPTIONS

Let $\mathcal{U}$ and $\mathcal{U}'$ be r-dimensional subspaces of $\mathbb{R}^{\mathrm{d}}$. Let $\mathbf{u} \in \mathbb{R}^{\mathrm{d}}$ be a vector in $\mathcal{U}'$. The ideal goal is to update $\mathcal{U}$ according to $\mathbf{u}$ in order to obtain a new subspace $\hat{\mathcal{U}}$ that is closer to $\mathcal{U}'$. The difficulty is that instead of observing $\mathbf{u}$, we observe $\mathbf{x}_{\boldsymbol{\omega}}$, defined as follows.

Let $\boldsymbol{\omega}$ be a subset of $\{1, \dots, \mathrm{d}\}$. For any subspace, matrix or vector that is compatible with a set of indices $\cdot$, we will use the subscript $\cdot$ to denote its restriction to the coordinates/rows in $\cdot$. For example, $\mathcal{U}_{\boldsymbol{\omega}} \subset \mathbb{R}^{|\boldsymbol{\omega}|}$ denotes the restriction of $\mathcal{U}$ to the coordinates in $\boldsymbol{\omega}$, and $\mathbf{u}_{\boldsymbol{\omega}} \in \mathbb{R}^{|\boldsymbol{\omega}|}$ denotes the restriction of $\mathbf{u}$ to the rows in $\boldsymbol{\omega}$.

Let $\{\boldsymbol{i}, \boldsymbol{i}^{\mathsf{c}}\}$ be a partition of the elements in $\boldsymbol{\omega}$. Let $\mathbf{x}$ be a vector in $\mathbb{R}^{\mathrm{d}}$ that is equal to $\mathbf{u}$ on the rows in $\boldsymbol{i}$, equal to an arbitrary vector $\mathbf{v} \in \mathbb{R}^{\mathrm{d}}$ on the rows in $\boldsymbol{i}^{\mathsf{c}}$, and unknown on the rows in $\{1, \dots, \mathrm{d}\} \backslash \boldsymbol{\omega}$. In other words, $\boldsymbol{i}$ indicates the *inlier* entries of $\mathbf{x}_{\boldsymbol{\omega}}$. To summarize, we observe $\mathbf{x}_{\boldsymbol{\omega}}$ composed of:

$$\begin{aligned} \mathbf{x}_{\boldsymbol{i}} &= \mathbf{u}_{\boldsymbol{i}}, \\ \mathbf{x}_{\boldsymbol{i}^{\mathsf{c}}} &= \mathbf{v}_{\boldsymbol{i}^{\mathsf{c}}}, \end{aligned} \tag{1}$$

and the goal is to update $\mathcal{U}$ according to the incomplete inlier vector $\mathbf{u}_{\boldsymbol{i}}$.

Remark 1. *The updating process is repeated continuously. More precisely, we have a sequence of subspaces* $\{\mathcal{U}^{\mathrm{t}}\}_{\mathrm{t} \geq 0}$. *At each time* $\mathrm{t} > 0$ *we have an estimate* $\hat{\mathcal{U}}^{\mathrm{t}-1}$, *and we observe* $(\mathbf{x}^{\mathrm{t}})_{\boldsymbol{\omega}^{\mathrm{t}}}$, *composed of* $(\mathbf{x}^{\mathrm{t}})_{\boldsymbol{i}^{\mathrm{t}}} = (\mathbf{u}^{\mathrm{t}})_{\boldsymbol{i}^{\mathrm{t}}}$ *and* $(\mathbf{x}^{\mathrm{t}})_{(\boldsymbol{i}^{\mathsf{c}})^{\mathrm{t}}} = (\mathbf{v}^{\mathrm{t}})_{(\boldsymbol{i}^{\mathsf{c}})^{\mathrm{t}}}$, *where* $\mathbf{u}^{\mathrm{t}} \in \mathcal{U}^{\mathrm{t}}$ *and* $\mathbf{v}^{\mathrm{t}} \in \mathbb{R}^{\mathrm{d}}$. *The goal is to update* $\hat{\mathcal{U}}^{\mathrm{t}-1}$ *according to* $(\mathbf{u}^{\mathrm{t}})_{\boldsymbol{i}^{\mathrm{t}}}$ *to obtain a new estimate* $\hat{\mathcal{U}}^{\mathrm{t}}$, *to be updated at time* $\mathrm{t}+1$ *from* $(\mathbf{x}^{\mathrm{t}+1})_{\boldsymbol{\omega}^{\mathrm{t}+1}}$, *and so on. However, to ease the notation we will drop the subscript* t *with the understanding that the updating process is repeated iteratively.*

Without any further assumptions on $\boldsymbol{i}$ and $\mathcal{U}$, this problem is ill-posed. For instance, for any subspace $\mathcal{U}$ in general position, and any *candidate* set $\boldsymbol{\kappa} \subset \{1, \dots, \mathrm{d}\}$ with fewer than $\mathrm{r}+1$ elements, we have $\mathcal{U}_{\boldsymbol{\kappa}} = \mathbb{R}^{|\boldsymbol{\kappa}|}$. This implies that *any* incomplete vector $\mathbf{x}_{\boldsymbol{\kappa}}$ will *perfectly* fit in $\mathcal{U}_{\boldsymbol{\kappa}}$ (regardless of whether the entries in $\boldsymbol{\kappa}$ are inliers or outliers). In such case, since $\mathbf{x}_{\boldsymbol{\kappa}} \in \mathcal{U}_{\boldsymbol{\kappa}}$, the update in $\mathcal{U}$ would be zero. Consequently, a fundamental requirement of subspace tracking is that $\mathbf{x}_{\boldsymbol{\omega}}$ contains at least $\mathrm{r}+1$ inlier entries (so that we can update $\mathcal{U}$). Hence, we will assume without loss of generality that

(**A1**) The set of inliers $\boldsymbol{i}$ contains more than r elements.

On the other hand, consider subspaces $\mathcal{U}, \mathcal{U}'$ that are only supported on the first $\mathrm{r}+1$ coordinates. Then we would be unable to track $\mathcal{U}$ unless the set of inliers $\boldsymbol{i}$ includes those coordinates. To rule out scenarios like this, typical completion results require low coherence and uniform sampling [1, 2, 14]. However, as discussed above, these assumptions do not hold in many settings. Hence, to avoid these issues, we will assume that $\mathcal{U}$ and $\mathcal{U}'$ are in general position. More precisely, we will assume:

(**A2**) $\mathcal{U}$ and $\mathcal{U}'$ are drawn independently according to an absolutely continuous distribution with respect to the uniform measure on the Grassmann manifold of r-dimensional subspaces of $\mathbb{R}^{\mathrm{d}}$.

A2 essentially requires that $\mathcal{U}$ and $\mathcal{U}'$ are *generic* subspaces. Similar sorts of genericity assumptions have been used to study standard LRMC and related problems [15–22].

3. ERASURE ALGORITHM

Notice that if we knew the set of inliers $\boldsymbol{i}$, then we would know which entries of $\mathbf{x}_{\boldsymbol{\omega}}$ correspond to $\mathbf{u}$, and we would be able to update $\mathcal{U}$ using the results in [13], which precisely address how to update a subspace according to an incomplete vector. In fact, [13] shows that moving $\mathcal{U}$ a step of length η in the direction of $\mathbf{u}_{\boldsymbol{i}}$ results in the subspace $\hat{\mathcal{U}}$ spanned by

$$\mathbf{U} \;+\; \left((\cos(\sigma\eta) - 1) \frac{\hat{\mathbf{u}}}{\|\hat{\mathbf{u}}\|} + \sin(\sigma\eta) \frac{\mathbf{r}}{\|\mathbf{r}\|} \right) \frac{\boldsymbol{\theta}^{\mathsf{T}}}{\|\boldsymbol{\theta}\|}, \tag{2}$$

where

- $\mathbf{U} \in \mathbb{R}^{\mathrm{d}\times\mathrm{r}}$ is an orthogonal basis of $\mathcal{U}$,
- $\boldsymbol{\theta} = (\mathbf{U}_{\boldsymbol{i}}^{\mathsf{T}}\mathbf{U}_{\boldsymbol{i}})^{-1}\mathbf{U}_{\boldsymbol{i}}^{\mathsf{T}}\mathbf{u}_{\boldsymbol{i}}$ is the coefficient of $\mathbf{u}_{\boldsymbol{i}}$ with respect to $\mathbf{U}_{\boldsymbol{i}}$ (notice that **A1** and **A2** guarantee that $\mathbf{U}_{\boldsymbol{i}}^{\mathsf{T}}\mathbf{U}_{\boldsymbol{i}}$ is invertible).
- $\hat{\mathbf{u}} = \mathbf{U}\boldsymbol{\theta}$ is the *completion* of $\mathbf{u}_{\boldsymbol{i}}$,
- $\mathbf{r}_{\mathrm{i}} = \mathbf{u}_{\mathrm{i}} - \hat{\mathbf{u}}_{\mathrm{i}}$ is the *residual* of $\mathbf{u}_{\mathrm{i}}$,
- $\mathbf{r} \in \mathbb{R}^{\mathrm{d}}$ is the *padded* residual, which has the entries of $\mathbf{r}_{\mathrm{i}}$ in the locations of i, and zeros elsewhere,

- $\sigma = \|\hat{\mathbf{u}}\|\|\mathbf{r}\|$, and
- $\|\cdot\|$ denotes the euclidian norm.

The challenge is that we don't know the set of inliers $\boldsymbol{i}$. Hence, we will focus on identifying $\boldsymbol{i}$, with the understanding that after this, we can update $\mathcal{U}$ according to (2).

Using now standard convex relaxation techniques, [1] suggests to find $\boldsymbol{i}$ by solving the following convex optimization:

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^{\mathrm{r}}} \|\mathbf{U}_{\boldsymbol{\omega}}\boldsymbol{\theta} - \mathbf{x}_{\boldsymbol{\omega}}\|_1, \tag{3}$$

where $\|\cdot\|_1$ denotes the ℓ_1-norm, given by the sum of absolute values. The insight behind (3) is that the ℓ_1-norm will favor solutions with many entries close to zero. We know that the entries in $(\mathbf{U}_{\boldsymbol{i}}\boldsymbol{\theta} - \mathbf{x}_{\boldsymbol{i}})$ are small. Hence, if $|\boldsymbol{i}| > |\boldsymbol{i}^{\mathsf{c}}|$ (most entries of $\boldsymbol{\omega}$ are inliers), we can expect (3) to produce a solution $\boldsymbol{\theta}$ such that most entries in $(\mathbf{U}_{\boldsymbol{\omega}}\boldsymbol{\theta} - \mathbf{x}_{\boldsymbol{\omega}})$ are close to zero, at which point we can find $\boldsymbol{i}$ by inspection: the small entries in $(\mathbf{U}_{\boldsymbol{\omega}}\boldsymbol{\theta} - \mathbf{x}_{\boldsymbol{\omega}})$.

However, as we will see, if $|\boldsymbol{i}| \leq |\boldsymbol{i}^{\mathsf{c}}|$ (most entries of $\boldsymbol{\omega}$ are outliers), then (3) will produce a solution $\boldsymbol{\theta}$ that minimizes most entries in $(\mathbf{U}_{\boldsymbol{\omega}}\boldsymbol{\theta} - \mathbf{x}_{\boldsymbol{\omega}})$. In general, such solution will not produce a vector $(\mathbf{U}_{\boldsymbol{\omega}}\boldsymbol{\theta} - \mathbf{x}_{\boldsymbol{\omega}})$ with near-zero entries, whence $\boldsymbol{i}$ can no longer be found by inspection, as before.

Hence, instead of searching for a coefficient $\boldsymbol{\theta}$ that explains *most* entries of $\mathbf{x}_{\boldsymbol{\omega}}$, we propose to search for $\boldsymbol{i}$ directly, by finding a subset of entries $\boldsymbol{\kappa} \subset \boldsymbol{\omega}$ where $\mathbf{x}_{\boldsymbol{\kappa}}$ is close to $\mathcal{U}_{\boldsymbol{\kappa}}$ (indicating that the entries in $\boldsymbol{\kappa}$ are inliers). To this end, let $\mathbf{P}_{\boldsymbol{\kappa}} = \mathbf{U}_{\boldsymbol{\kappa}}(\mathbf{U}_{\boldsymbol{\kappa}}^{\mathsf{T}}\mathbf{U}_{\boldsymbol{\kappa}})^{-1}\mathbf{U}_{\boldsymbol{\kappa}}^{\mathsf{T}}$ denote the projection operator onto $\mathcal{U}_{\boldsymbol{\kappa}}$. Recall that $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\| \leq 0$ with equality iff $\mathbf{x}_{\boldsymbol{\kappa}} \in \mathcal{U}_{\boldsymbol{\kappa}}$. In fact, $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\|$ is close to zero iff $\mathbf{x}_{\boldsymbol{\kappa}}$ is close to $\mathcal{U}_{\boldsymbol{\kappa}}$, and $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\|$ is far from zero iff $\mathbf{x}_{\boldsymbol{\kappa}}$ is far from $\mathcal{U}_{\boldsymbol{\kappa}}$. Hence we want to find a set $\boldsymbol{\kappa} \subset \boldsymbol{\omega}$ that maximizes $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\|$.

On the other hand, recall that if $|\boldsymbol{\kappa}| \leq \mathrm{r}$, then $\mathbf{x}_{\boldsymbol{\kappa}}$ will trivially lie in $\mathcal{U}_{\boldsymbol{\kappa}}$ (see the discussion above **A1**), whence $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\| = 0$. Consequently, any set $\boldsymbol{\kappa}$ with fewer than r elements will trivially maximize $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\|$, regardless of whether the entries in $\boldsymbol{\kappa}$ are inliers or outliers. It follows that only sets $\boldsymbol{\kappa}$ with more than r elements are meaningful, in the sense that they may reveal $\boldsymbol{i}$.

To summarize, we want to find a set $\boldsymbol{\kappa}$ with more than r elements such that $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\| \approx 0$. Moreover, we want to find the largest such set $\boldsymbol{\kappa}$, so that it contains most of the elements in $\boldsymbol{i}$. More precisely, we propose to solve:

$$\hat{\boldsymbol{i}} \;:=\; \arg\max_{\substack{\boldsymbol{\kappa}\subset\boldsymbol{\omega}:\\ |\boldsymbol{\kappa}|>\mathrm{r}}} |\boldsymbol{\kappa}| \quad \text{such that} \quad \|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\| < -\eta\frac{|\boldsymbol{\kappa}|}{\mathrm{d}}, \tag{4}$$

where η denotes is a parameter quantifying the distance between $\mathcal{U}$ and $\mathcal{U}'$ (i.e., how fast $\mathcal{U}$ changes with time). The term $\frac{|\boldsymbol{\kappa}|}{\mathrm{d}}$ is just weighting that by the fraction of entries in $\boldsymbol{\kappa}$. If $\boldsymbol{\kappa} \cap \boldsymbol{i}^{\mathsf{c}} = \emptyset$ (all entries in $\boldsymbol{\kappa}$ are inliers), we can expect the difference between the norms of $\mathbf{x}_{\boldsymbol{\kappa}}$ and its projection onto $\mathcal{U}_{\boldsymbol{\kappa}}$ to be smaller than $\eta\frac{|\boldsymbol{\kappa}|}{\mathrm{d}}$ (close to zero). On the other hand, if

$\boldsymbol{\kappa} \cap \boldsymbol{i}^{\mathsf{c}} \neq \emptyset$ (some entries in $\boldsymbol{\kappa}$ are outliers), we can expect the difference between the norms of $\mathbf{x}_{\boldsymbol{\kappa}}$ and its projection onto $\mathcal{U}_{\boldsymbol{\kappa}}$ to be larger than $\eta\frac{|\boldsymbol{\kappa}|}{\mathrm{d}}$ (far from zero).

Hence, the solution to (4) should recover the desired set of inliers $\boldsymbol{i}$. Unfortunately, (4) is non-convex, and in general NP-hard. Hence, we propose a top-down *erasure* approach to try to solve (4). The main idea is to start our search with $\boldsymbol{\kappa} = \boldsymbol{\omega}$, and then iteratively remove the entries (coordinates) of $\boldsymbol{\kappa}$ that most increase the gap between $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\|$ and $\|\mathbf{x}_{\boldsymbol{\kappa}}\|$ (hence the term *erasure*). We stop this procedure when $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\|$ is close to $\|\mathbf{x}_{\boldsymbol{\kappa}}\|$.

More precisely, we initialize $\boldsymbol{\kappa} = \boldsymbol{\omega}$, and we iteratively redefine $\boldsymbol{\kappa}$ as the set $\boldsymbol{\kappa}\backslash\mathrm{k}$, where

$$\mathrm{k} \;=\; \arg\max_{k\in\boldsymbol{\kappa}} \;\; \|\mathbf{P}_{\boldsymbol{\kappa}\backslash k}\mathbf{x}_{\boldsymbol{\kappa}\backslash k}\| - \|\mathbf{x}_{\boldsymbol{\kappa}\backslash k}\|.$$

In words, k is the coordinate in $\boldsymbol{\kappa}$ such that if ignored, the gap between the remaining vector $\mathbf{x}_{\boldsymbol{\kappa}\backslash\mathrm{k}}$ and its projection $\mathbf{P}_{\boldsymbol{\kappa}\backslash\mathrm{k}}\mathbf{x}_{\boldsymbol{\kappa}\backslash\mathrm{k}}$ is reduced the most. Each iteration we remove (erase) such coordinate k from $\boldsymbol{\kappa}$. The intuition behind this approach is that the outlier entries in $\boldsymbol{\kappa}$ are more likely to increase the gap between $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\|$ and $\|\mathbf{x}_{\boldsymbol{\kappa}}\|$.

Recall that if $|\boldsymbol{\kappa}| \leq \mathrm{r}$, then $\mathbf{x}_{\boldsymbol{\kappa}}$ trivially lies in $\mathcal{U}_{\boldsymbol{\kappa}}$ (see discussion above **A1**), whence $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\| = 0$. Hence the procedure above is guaranteed to terminate after at most $\|\boldsymbol{\omega}\| - \mathrm{r}$ iterations. If $|\boldsymbol{\kappa}| = \mathrm{r}$, then we know that we were unable to find $\boldsymbol{i}$ (or a subset of it). One alternative is to start with a different $\boldsymbol{\kappa} \subsetneq \boldsymbol{\omega}$, and search again.

The erasure procedure may remove some inlier entries along the way, so in general, the output of this procedure will be a set $\boldsymbol{\kappa} \subset \boldsymbol{i}$. However, finding a subset of $\boldsymbol{i}$ is enough to find $\boldsymbol{i}$. To see this, recall that since $\mathbf{x}_{\boldsymbol{i}} = \mathbf{u}_{\boldsymbol{i}}$ lies in $\mathcal{U}'_{\boldsymbol{i}}$, there is a coefficient vector $\boldsymbol{\theta}' \in \mathbb{R}^{\mathrm{r}}$ such

---

Algorithm 1: Selective Erasure Robust Subspace Tracking.

Input: Subspace basis $\mathbf{U} \in \mathbb{R}^{\mathrm{d}\times\mathrm{r}}$, partially observed data vector $\mathbf{x}_{\boldsymbol{\omega}}$, update step η.

Initialize: $\boldsymbol{\kappa} = \boldsymbol{\omega}$. Recall that $\mathbf{P}_{\boldsymbol{\kappa}} = \mathbf{U}_{\boldsymbol{\kappa}}(\mathbf{U}_{\boldsymbol{\kappa}}^{\mathsf{T}}\mathbf{U}_{\boldsymbol{\kappa}})^{-1}\mathbf{U}_{\boldsymbol{\kappa}}^{\mathsf{T}}$ is the projection operator onto $\mathcal{U}_{\boldsymbol{\kappa}}$.

Erasure: repeat until $\|\mathbf{P}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}\| - \|\mathbf{x}_{\boldsymbol{\kappa}}\| < -\eta\frac{|\boldsymbol{\kappa}|}{\mathrm{d}}$:

$$\begin{aligned}\mathrm{k} \;&=\; \arg\max_{k\in\boldsymbol{\kappa}} \;\; \|\mathbf{P}_{\boldsymbol{\kappa}\backslash k}\mathbf{x}_{\boldsymbol{\kappa}\backslash k}\| - \|\mathbf{x}_{\boldsymbol{\kappa}\backslash k}\|,\\ \boldsymbol{\kappa} \;&=\; \boldsymbol{\kappa}\backslash\mathrm{k},\end{aligned}$$

Refinement of $\boldsymbol{\kappa}$:

Complete: $\hat{\mathbf{u}} = \mathbf{U}(\mathbf{U}_{\boldsymbol{\kappa}}^{\mathsf{T}}\mathbf{U}_{\boldsymbol{\kappa}})^{-1}\mathbf{U}_{\boldsymbol{\kappa}}^{\mathsf{T}}\mathbf{x}_{\boldsymbol{\kappa}}$.

Estimate inliers as entries close to $\hat{\mathbf{u}}$:

$$\hat{\boldsymbol{i}} \;=\; \left\{\mathrm{i} \in \boldsymbol{\omega} \;:\; |\mathrm{x_i} - \hat{\mathrm{u}}_{\mathrm{i}}| < \frac{\eta}{\mathrm{d}}\right\}.$$

Update: New subspace $\hat{\mathcal{U}}$ is spanned by (2), with $\hat{\boldsymbol{i}}$ instead of $\boldsymbol{i}$.

Output: Updated subspace $\hat{\mathcal{U}}$.

---

that $\mathbf{x}_{\boldsymbol{i}} = \mathbf{U}'_{\boldsymbol{i}}\boldsymbol{\theta}'$. Since $\boldsymbol{\kappa} \subset \boldsymbol{i}$, it follows that $\mathbf{x}_{\boldsymbol{\kappa}} = \mathbf{U}'_{\boldsymbol{\kappa}}\boldsymbol{\theta}'$. Furthermore, since $|\boldsymbol{\kappa}| \geq \mathrm{r}$, we can find $\boldsymbol{\theta}'$ as $\boldsymbol{\theta}' = (\mathbf{U}'^{\mathsf{T}}_{\boldsymbol{\kappa}}\mathbf{U}'_{\boldsymbol{\kappa}})^{-1}\mathbf{U}'^{\mathsf{T}}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}$. Since $\mathcal{U}$ is close to $\mathcal{U}'$, we know that $\mathbf{x}_{\boldsymbol{i}} \approx \mathbf{U}_{\boldsymbol{i}}\boldsymbol{\theta}$, where $\boldsymbol{\theta} = (\mathbf{U}^{\mathsf{T}}_{\boldsymbol{\kappa}}\mathbf{U}_{\boldsymbol{\kappa}})^{-1}\mathbf{U}^{\mathsf{T}}_{\boldsymbol{\kappa}}\mathbf{x}_{\boldsymbol{\kappa}}$. At this point we can identify $\boldsymbol{i}$ by simple inspection (the entries in $\mathbf{x}_{\boldsymbol{\omega}}$ that are close to $\hat{\mathbf{u}}_{\boldsymbol{\omega}} := \mathbf{U}_{\boldsymbol{\omega}}\boldsymbol{\theta}$).

Once $\boldsymbol{i}$ is identified, we update $\mathcal{U}$ according to (2). The whole procedure is summarized in Algorithm 1.

4. EXPERIMENTS

We now present a series of experiments to analyze the performance of Algorithm 1, and compare it with the state-of-the-art ℓ_1-minimization approach.

In our experiments, we first generated a matrix $\mathbf{U} \in \mathbb{R}^{\mathrm{d}\times\mathrm{r}}$ with i.i.d. $\mathcal{N}(0,1)$ entries, to use as a basis of $\mathcal{U}$. Next we added i.i.d. $\mathcal{N}(0,\eta^2)$ entries to $\mathbf{U}$ to obtain a new matrix $\mathbf{U}' \in \mathbb{R}^{\mathrm{d}\times\mathrm{r}}$, which we will use as basis of $\mathcal{U}'$; here $\mathrm{d} = 100$ and $\mathrm{r} = 5$. The parameter η models the distance between $\mathcal{U}'$ and $\mathcal{U}$. Then we generated a coefficient vector $\boldsymbol{\theta}' \in \mathbb{R}^{\mathrm{r}}$ with i.i.d. $\mathcal{N}(0,1)$ entries to construct the inlier vector $\mathbf{u} = \mathbf{U}'\boldsymbol{\theta}'$.

Notice that the problem only depends on $\boldsymbol{\omega}$ (the set of observed entries in $\mathbf{x}$) through $\boldsymbol{i}$ (the set of inlier entries). Given $\boldsymbol{i}$, changing $\boldsymbol{\omega}$ only changes $\boldsymbol{i}^{\mathsf{c}}$ (the set of outliers). In fact, given $\boldsymbol{i}$, enlarging $\boldsymbol{\omega}$ (observing more entries in $\mathbf{x}$) equates to having more outliers, which can only complicate the identification of inliers. In our experiments we will consider the most complicated scenario. That is, we will draw $\boldsymbol{i} \subset \{1,\ldots,\mathrm{d}\}$ randomly, and set $\boldsymbol{\omega} = \{1,\ldots,\mathrm{d}\}$ such that $\boldsymbol{i}^{\mathsf{c}} := \boldsymbol{\omega}\backslash\boldsymbol{i}$ is as large as possible.

Finally, we generated the outlier vector $\mathbf{v} \in \mathbb{R}^{\mathrm{d}}$ with i.i.d. $\mathcal{N}(0,1)$ entries, and constructed the observed vector $\mathbf{x}$ according to (1).

Recall that once the set of inlier entries $\boldsymbol{i}$ is known, we can update $\mathcal{U}$ according to (2). Hence, the performance of Algorithm 1 depends entirely on its ability to identify $\boldsymbol{i}$. With this in mind, our first experiment will tests the accuracy of our approach to identifying the set of inliers $\boldsymbol{i}$ as a function of the fraction of outliers $\mathrm{q} := \frac{|\boldsymbol{i}^{\mathsf{c}}|}{|\boldsymbol{\omega}|}$ and the distance η between $\mathcal{U}$ and $\mathcal{U}'$. We considered a success if the set of inliers $\boldsymbol{i}$ was identified exactly. The results are summarized in Figure 1, where we repeated the simulation above 100 trials for each pair (q,η). We can see that our approach outperforms the state-of-the-art method (ℓ_1-minimization), specially when there are more outliers than inliers ($\mathrm{q} \geq \frac{1}{2}$).

Notice that as η increases, $\mathbf{U}$ and $\mathbf{U}'$ become more separated, whence $\mathbf{u}_{\boldsymbol{i}} \in \mathcal{U}'_{\boldsymbol{i}}$ gets farther from $\mathcal{U}_{\boldsymbol{i}}$, up to the point where even $\mathbf{v}_{\boldsymbol{i}}$ is closer to $\mathcal{U}$. In other words, as η increases, $\mathbf{u}$ becomes more an outlier than an inlier, and vice-versa for $\mathbf{v}$. This way, η encodes the amount of the *noise*. Figure 1 shows, the performance of our *erasure* approach decays nicely as η (noise) increases.

The experiment in Figure 1 only counts success or failures. In our experiments we also kept track of the *error*, measured as the ratio of misclassified entries (misclassified inliers + misclassified outliers) vs. number of inliers. The top of Figure 2 shows two *slices* of Figure 1 corresponding to low noise ($\eta = 10^{-9}$) and larger noise ($\eta = 10^{-3}$). The bottom of Figure 2 shows the corresponding error in each of these cases. Pay attention to the right side of the

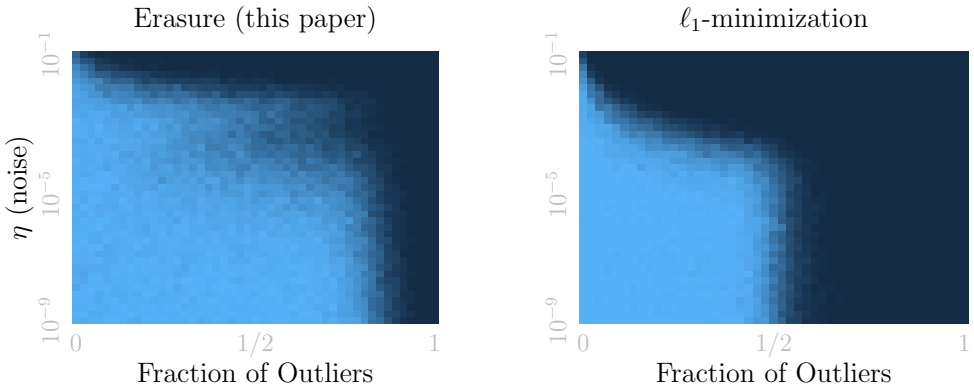

Figure 1: Percentage of times (average over 100 trials) that the set of inliers $\boldsymbol{i}$ was identified exactly as a function of the fraction of outliers and η, which encodes the information of the noise and the distance between subspaces $\mathcal{U}$ and $\mathcal{U}'$. The lightest color represents 100% success rate, and the darkest represents 0%.

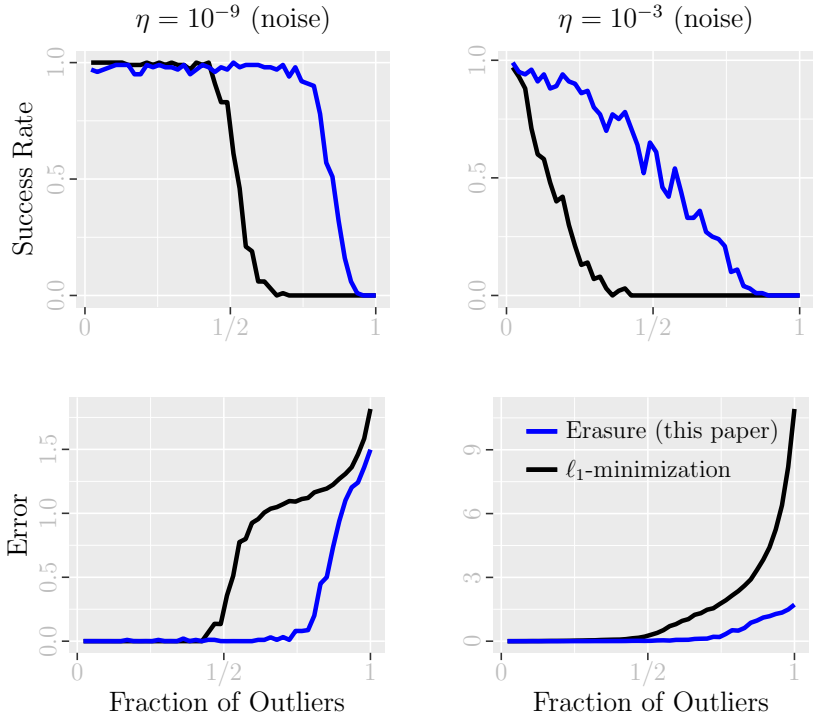

Figure 2: **Top:** Percentage of times (average over 100 trials) that the set of inliers $\boldsymbol{i}$ was identified exactly as a function of the fraction of outliers, for low noise ($\eta = 10^{-9}$) and larger noise ($\eta = 10^{-3}$). The higher the better. **Bottom:** Error (misclassified entries; average over 100 trials) as a function of the fraction of outliers. The lower the better.

plots (fraction of outliers $> \frac{1}{2}$). Notice that if the noise is small ($\eta = 10^{-9}$), both approaches make about the same amount of mistakes, but if the noise is larger ($\eta = 10^{-3}$), then the error of ℓ_1-minimization grows exponentially, while the error of our approach stays constant.

In our final experiment we study the accuracy of Algorithm 1 at tracking a subspace. First we

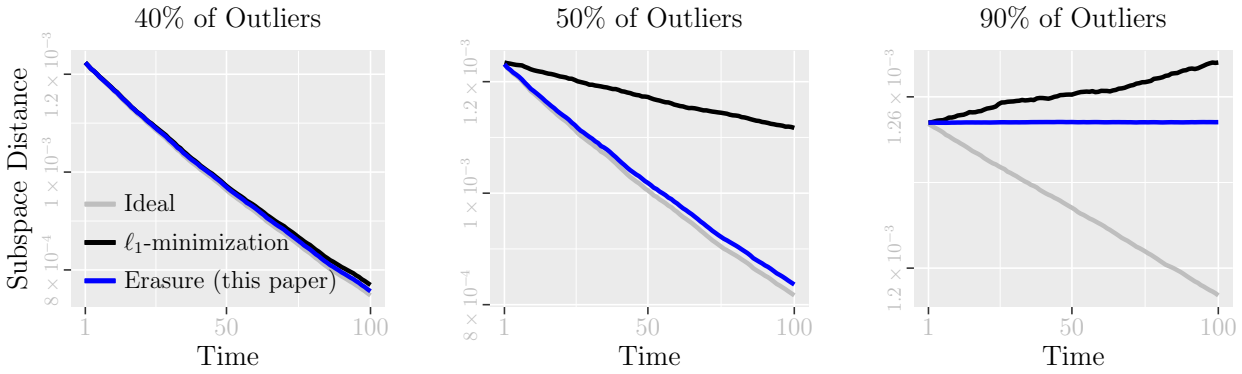

Figure 3: Distance over time between the subspace $\mathcal{U}'$ that we aim to track, and the subspace estimate $\hat{\mathcal{U}}$ produced by our erasure algorithm, by ℓ_1-minimization, and by an ideal algorithm that knows exactly which entries are inliers.

generate $\mathcal{U}$ and $\mathcal{U}'$ as before, and initialize our subspace estimate $\hat{\mathcal{U}}$ as $\mathcal{U}$. Then at each time we generate $\mathbf{u} \in \mathcal{U}'$, $\mathbf{v} \in \mathbb{R}^{\mathrm{d}}$, $\boldsymbol{i}$, and $\mathbf{x}$, as described before. Next we obtain a new estimate $\hat{\mathcal{U}}$ using Algorithm 1 and ℓ_1-minimization (we use ℓ_1 minimization to find inliers, followed (2) to update the subspace). Finally, we record the distance between $\hat{\mathcal{U}}$ and $\mathcal{U}'$ over time, measured as the Frobenius norm of the difference between their projector operators. The results are summarized in Figure 3, where we also compare to an *ideal* update, which *knows* exactly which entries are inliers.

Consistent with our previous experiments, if fewer than 50% of the entries are outliers, then both algorithms have perfect performance (their performance is equal to the *ideal update*, which *knows* which entries are inliers). However, if 50% or more of the entries are outliers, the state-of-the-art solution (ℓ_1-minimization) starts failing severely, up to the point where $\hat{\mathcal{U}}$ starts getting farther and farther from $\mathcal{U}'$. In contrast, the performance of Algorithm 1 decays nicely with the number of outliers, and its estimate $\hat{\mathcal{U}}$ never gets farther from $\mathcal{U}'$.

References

[1] J. He, L. Balzano and A. Szlam, *Incremental gradient on the Grassmannian for online foreground and background separation in subsampled video*, Conference on Computer Vision and Pattern Recognition, 2012.

[2] H. Mansour, X. Jiang, *A robust online subspace estimation and tracking algorithm*, IEEE International Conference on Acoustics, Speech, and Signal Processing, 2015.

[3] E. Learned-Miller M. Narayana and A. Hanson, *Coherent motion segmentation in moving camera videos using optical flow orientations*, International Conference on Computer Vision, 2013.

[4] B. Ardekani, J. Kershaw, K. Kashikura and I. Kanno, *Activation detection in functional MRI using subspace modeling and maximum likelihood estimation*, IEEE Transactions on Medical Imaging, 1999.

[5] D. Gross, *Recovering Low-Rank Matrices From Few Coefficients in Any Basis*, IEEE Transactions on Information Theory, 2011.

[6] B. Recht and C. Ré, *Parallel stochastic gradient algorithms for large-scale matrix completion*, Mathematical Programming Computation, 2013.

[7] M. McCloud and L. Scharf, *Interference estimation with applications to blind multiple-access communication over fading channels*, IEEE Transactions on Information Theory, 2000.

[8] D. Stein, S. Beaven, L. Hoff, E. Winter, A. Schaum, and A. Stocker, *Anomaly detection from hyperspectral imagery*, IEEE Signal Processing Magazine, 2002.

[9] T. Ahmed, M. Coates and A. Lakhina, *Multivariate online anomaly detection using kernel recursive least squares*, INFOCOM. 2007.

[10] H. Kwon and N. Nasrabadi, *Kernel matched subspace detectors for hyperspectral target detection*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006.

[11] S. Shahbazpanahi, S. Valaee and M. Bastani, *Distributed source localization using ESPRIT algorithm*, IEEE Transactions on Signal Processing, 2001.

[12] H. Krim and M. Viberg, *Two decades of array signal processing research: the parametric approach*, Signal Processing Magazine, 1996.

[13] L. Balzano, R. Nowak and B. Recht, *Online identification and tracking of subspaces from highly incomplete information*, Allerton Conference on Communication, Control and Computing, 2010.

[14] L. Balzano, B. Recht and R. Nowak, *High-dimensional matched subspace detection when data are missing*, IEEE International Symposium on Information Theory, 2010.

[15] D. Pimentel-Alarcón and R. Nowak, *The information-theoretic requirements of subspace clustering with missing data*, International Conference on Machine Learning, 2016.

[16] D. Pimentel-Alarcón, N. Boston and R. Nowak, *Deterministic conditions for subspace identifiability from incomplete sampling*, IEEE International Symposium on Information Theory, 2015.

[17] D. Pimentel-Alarcón, N. Boston and R. Nowak, *A characterization of deterministic sampling patterns for low-rank matrix completion*, IEEE Journal of Selected Topics in Signal Processing, 2016.

[18] D. Pimentel-Alarcón and R. Nowak, *A converse to low-rank matrix completion*, IEEE International Symposium on Information Theory, 2016.

[19] D. Pimentel-Alarcón and R. Nowak, *Random consensus robust PCA*, Artificial Intelligence and Statistics, 2017.

[20] F. Király, L. Theran and R. Tomioka, *The algebraic combinatorial approach for low-rank matrix completion*, Journal of Machine Learning Research, 2015.

[21] M. Ashraphijuo, X. Wang and V. Aggarwal, *A characterization of sampling patterns for low-rank multiview data completion problem*, IEEE International Symposium on Information Theory, 2017.

[22] M. Ashraphijuo, V. Aggarwal and X. Wang, *A characterization of sampling patterns for low-tucker-rank tensor completion problem*, IEEE International Symposium on Information Theory, 2017.