Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Online Robust Low Rank Matrix Recovery

Xiaojie Guo
State Key Laboratory of Information Security
Institute of Information Engineering, Chinese Academy of Sciences
xj.max.guo@gmail.com

Abstract

Low rank matrix recovery has shown its importance as a theoretic foundation in many areas of information processing. Its solutions are usually obtained in batch mode that requires to load all the data into memory during processing, and thus are hardly applicable on large scale data. Moreover, a fraction of data may be severely contaminated by outliers, which makes accurate recovery significantly more challenging. This paper proposes a novel online robust low rank matrix recovery method to address these difficulties. In particular, we first introduce an online algorithm to solve the problem of low rank matrix completion. Then we move on to low rank matrix recovery from observations with intensive outliers. The outlier support is robustly estimated from a perspective of mixture model. Experiments on both synthetic and real data are conducted to demonstrate the efficacy of our method and show its superior performance over the state-of-the-arts.

1 Introduction

Low Rank Matrix Recovery (LRMR) aims to seek a low-dimensional subspace from given data such that the residual between the given and the recovery is minimized with respect to some optimality criterion [Recht *et al.*, 2010; Chandrasekaran *et al.*, 2011]. Its benefit, besides being a standard tool for dimensionality reduction, has been witnessed by a wide spectrum of low level signal processing applications, such as image denosing [Gu *et al.*, 2014], colorization [Wang and Zhang, 2012], reflection separation [Guo *et al.*, 2014a] and rectification [Zhang *et al.*, 2012], as well as higher level tasks, like collaborative filtering [Jamali and Ester, 2011; Park *et al.*, 2013; Shi *et al.*, 2013; Chen *et al.*, 2014], visual salience detection [Shen and Wu, 2012], video editing [Guo *et al.*, 2013] and dictionary learning [Mairal *et al.*, 2010].

The quality of LRMR largely depends on the choice of optimality criteria for different tasks. Principle Component Analysis (PCA) [Pearson, 1901] adopts the ℓ^2 loss (also known as the squared loss) by assuming that the residual existing in the observation follows a Gaussian distribution. Although the ℓ^2 is arguably the most commonly used loss, its performance may sharply degenerate when dealing with data polluted by outliers, which is frequently encountered in real-world applications. To be robust against outliers, the ℓ^0 loss is the ideal option. Unfortunately, the non-convexity, non-smoothness and difficulty of being approximated (NP-hard) make the ℓ^0 unsuitable for practical use. Alternatively, its tightest convex surrogate, the ℓ^1, is employed to achieve the robustness, which corresponds to the least absolute deviations technique and is actually optimal for Laplacian noises. In this way, Candès *et al.* proved that their proposed method, namely PCP [Candès *et al.*, 2011], can exactly recover the underling low rank matrix from the observation even with the fraction of outlier up to about $\frac{1}{3}$ under some conditions. In parallel, there are a few works carried out from probabilistic standpoints. Probabilistic Matrix Factorization (PMF) [Salakhutdinov and Mnih, 2008] and Probabilistic Robust Matrix Factorization (PRMF) [Wang *et al.*, 2012] are two representatives, the residual penalties of which are equivalent to using the ℓ^2 (PCA) and ℓ^1 (PCP), respectively. Although the techniques above have made great progresses in LRMR, the tolerance to outliers is expected to be further improved.

Another issue should be concerned is the applicability to large-scale data. Conventional solvers of LRMR, like PCA [Pearson, 1901], PCP [Candès *et al.*, 2011], PMF [Salakhutdinov and Mnih, 2008] and PRMF [Wang *et al.*, 2012] process data in batch mode, which are both computationally expensive and memory exhausting. Moreover, they are inflexible to incrementally add samples. To overcome the shortcomings of batch mode, several online strategies have been developed recently. GROUSE [Balzano *et al.*, 2011], an online extension of PCA, is derived by analyzing incremental gradient descent on the Grassmannian manifold of subspaces. GRASTA [He *et al.*, 2012] follows the same technical line as GROUSE with further consideration of outliers, which can be seen as an online version of PCP. Wang *et al.* modified PRMF as Online PRMF for sequentially processing samples [Wang *et al.*, 2012]. Online Robust PCA (OR-PCA) [Feng *et al.*, 2013] has proven that it converges to the same optimal solution as PCP [Candès *et al.*, 2011] does asymptotically. The methods mentioned above indeed enable the traditional schemes with the scalability, but the improvement space exists, especially in terms of the robustness to more corruptions.

This work proposes an online technique to robustly recover the low rank component from the observation with heavy outliers. Please notice that if the outlier support is given or precisely estimated, the problem reduces to the completion task that is much easier to conquer. Specifically, we first develop an online algorithm to solve the low rank matrix completion. Then, we step forward to a more challenging problem, say online low rank matrix recovery with robust outlier support estimation. As regards residual modeling, we employ a Gaussian component to fit the noise while a uniform distributed component to host the oultier. With this strategy, we are capable to handle more severely contaminated data. Extensive experiments on both synthetic and real data are conducted to verify the effectiveness of the proposed technique and demonstrate its advantages over the state-of-the-arts.

2 Our Method

Suppose we are given a finite set of data vectors $\boldsymbol{Y} = [\boldsymbol{y}_1, ..., \boldsymbol{y}_n] \in \mathbb{R}^{m\times n}$. It will be helpful to consider that low rank matrices can be factorized by $\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{V}^T$, where $\boldsymbol{U} \in \mathbb{R}^{m\times r}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times r}$. In such a way, the rank of recovered matrices is guaranteed to be never over r. This work formally concentrates on the following general problem:

$$\min_{\boldsymbol{U},\boldsymbol{V}} \|\boldsymbol{W} \odot (\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{V}^T)\|_F^2. \tag{1}$$

To prevent $\boldsymbol{U}$ from being with arbitrarily large values, the common manner is to enforce the ℓ^2 norm of each column to be less than or equal to one. For the rest of the paper, the discussion is always under this constraint.

Having $\boldsymbol{W} = [\boldsymbol{w}_1, ..., \boldsymbol{w}_n] \in \{0, 1\}^{m\times n}$ that indicates which elements of $\boldsymbol{Y}$ are observed, the associated problem is designated as Low Rank Matrix Completion (LRMC). The difficulty, however, greatly increases when the data are ruined by outliers, as the outlier support is unknown, we call this problem Robust Low Rank Matrix Recovery (RLRMR). In the next subsections, we will first develop an online algorithm based on stochastic optimization to handle LRMC, and then design a robust outlier support estimation scheme for solving RLRMF in an online fashion.

2.1 Online Low Rank Matrix Completion

With the data matrix $\boldsymbol{Y}$ and its corresponding support $\boldsymbol{W}$, solving the problem (1) is indeed equivalent to minimize the following empirical cost function:

$$\mathcal{F}_n(\boldsymbol{U}) \equiv \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{y}_i, \boldsymbol{w}_i, \boldsymbol{U}), \tag{2}$$

where the loss function for each sample is defined as:

$$f(\boldsymbol{y}_i, \boldsymbol{w}_i, \boldsymbol{U}) \equiv \min_{\boldsymbol{v}_i} \|\boldsymbol{w}_i \odot (\boldsymbol{y}_i - \boldsymbol{U}\boldsymbol{v}_i)\|_2^2. \tag{3}$$

As pointed out by Bottou and Bousquet [Bottou and Bousquet, 2008], one may be interested in the minimization of the expected cost instead of the empirical cost, as the empirical is just an approximation of the expected. The corresponding expected cost function is:

$$\mathcal{F}(\boldsymbol{U}) \equiv \mathbb{E}_{\boldsymbol{y},\boldsymbol{w}}\left[f(\boldsymbol{y}, \boldsymbol{w}, \boldsymbol{U})\right] = \lim_{n\to\infty} \mathcal{F}_n(\boldsymbol{U}) \quad a.s.^1 \tag{4}$$

[1]Here, "a.s." means almost sure convergence.

where the expectation is taken with respect to the distribution of the samples and the known supports. However, classical batch optimization methods become impractical in terms of speed and/or memory requirements to do so. As a consequence, online techniques are desired.

In the following, we first establish an online algorithm to minimize the empirical cost function (2). At the end of this subsection, we show that the proposed algorithm converges to the solution of the expected cost function (4) asymptotically. When a new sample $\boldsymbol{y}_t$ arrives, two things have to be dealt with, *i.e.* computing $\boldsymbol{v}_t$ and updating $\boldsymbol{U}^{(t)}$. With the help of alternating minimizing strategy, we first focus on $\boldsymbol{v}_t$, which associates with the following optimization problem with fixed $\boldsymbol{U}^{(t-1)}$ (obtained from previous iteration):

$$\begin{aligned} \boldsymbol{v}_t &= \operatorname*{argmin}_{\boldsymbol{v}_t} \|\boldsymbol{w}_t \odot (\boldsymbol{y}_t - \boldsymbol{U}^{(t-1)}\boldsymbol{v}_t)\|_2^2 \\ &= \operatorname*{argmin}_{\boldsymbol{v}_t} \|\boldsymbol{D}_t\boldsymbol{y}_t - \boldsymbol{D}_t\boldsymbol{U}^{(t-1)}\boldsymbol{v}_t\|_2^2, \end{aligned} \tag{5}$$

where $\boldsymbol{D}_t \equiv \mathrm{Diag}(\boldsymbol{w}_t)$. As can be seen from (5), it is a classic least squares problem and has a closed-form solution:

$$\boldsymbol{v}_t = (\boldsymbol{D}_t\boldsymbol{U}^{(t-1)})^\dagger \boldsymbol{D}_t\boldsymbol{y}_t, \tag{6}$$

where $(\boldsymbol{D}_t\boldsymbol{U}^{(t-1)})^\dagger$ denotes the Moore-Penrose pseudoinverse of $\boldsymbol{D}_t\boldsymbol{U}^{(t-1)}$.

For updating $\boldsymbol{U}$ in the t^{th} time instance, say $\boldsymbol{U}^{(t)}$, we minimize the following objective function:

$$\mathcal{S}_t(\boldsymbol{U}) \equiv \frac{1}{t}\sum_{i=1}^{t} \|\boldsymbol{D}_i\boldsymbol{y}_i - \boldsymbol{D}_i\boldsymbol{U}\boldsymbol{v}_i\|_2^2. \tag{7}$$

We can observe that $\mathcal{S}_t(\boldsymbol{U})$ is quadratic in $\boldsymbol{U}$ and aggregates the past data with a few sufficient $\boldsymbol{v}_i$. It is obvious that $\mathcal{S}_t(\boldsymbol{U})$ upper-bounds $\mathcal{F}_t(\boldsymbol{U})$ defined in (2). Similar to [Mairal *et al.*, 2010], we can adopt $\boldsymbol{U}^{(t-1)}$ as the warm restart for calculating $\boldsymbol{U}^{(t)}$. Because $\mathcal{S}_{t-1}(\boldsymbol{U})$ and $\mathcal{S}_t(\boldsymbol{U})$ become closer as t gets larger, and so do $\boldsymbol{U}^{(t-1)}$ and $\boldsymbol{U}^{(t)}$. This inspires us to employ the block-coordinate decent with warm restarts [Bertsekas, 1999] to update $\boldsymbol{U}$ efficiently. Moreover, considering some entries in the samples may be missing leads us to handle different rows of $\boldsymbol{U}$ separately based on the visibility of the corresponding elements, for the sake of information balance. In other words, if an element in a sample is invisible, it should contribute nothing to the update of the corresponding row of $\boldsymbol{U}$. Derived from (7), for each row $\boldsymbol{U}_{j\cdot}$, we have:

$$\mathcal{S}_t(\boldsymbol{U}_{j\cdot}) \equiv \frac{1}{t_j}\sum_{i_j=1}^{t_j} ([\boldsymbol{y}_{i_j}]_j - \boldsymbol{U}_{j\cdot}\boldsymbol{v}_{i_j})^2, \tag{8}$$

where $[\boldsymbol{y}_{i_j}]_j$ denotes the j^{th} entry of vector $\boldsymbol{y}_{i_j}$, t_j represents the number of times that the j^{th} row of samples is observed, and i_j is the refreshed observation index. The details of solving the problem (7) are provided in Algorithm 1. Please note that, $\boldsymbol{A_j}^{(t)} \equiv \sum_{i_j=1}^{t_j} \boldsymbol{v}_{i_j}\boldsymbol{v}_{i_j}^T$, and $\boldsymbol{b_j}^{(t)} \equiv \sum_{i_j=1}^{t_j} [\boldsymbol{y}_{i_j}]_j\boldsymbol{v}_{i_j}$. For clarity, we summarize the whole procedure to solve the problem (2) in Algorithm 2.

Algorithm 1: The Basis Update

```
Input: $\boldsymbol{U} = [\boldsymbol{u}_1, ..., \boldsymbol{u}_r] \in \mathbb{R}^{m\times r}, \forall k \in \{1, ..., m\}$
       $\boldsymbol{A_k} = [\boldsymbol{a_k}_1, ..., \boldsymbol{a_k}_r] \in \mathbb{R}^{r\times r}$ and $\boldsymbol{b_k} \in \mathbb{R}^r$.
for $j = 1$ to $r$ do
    for $k = 1$ to $m$ do
        $[\boldsymbol{u}_j]_k \leftarrow \frac{1}{[\boldsymbol{a_k}_j]_j}([\boldsymbol{b_k}]_j - \boldsymbol{U}_{k\cdot}\boldsymbol{a_k}_j) + [\boldsymbol{u}_j]_k$;
    end
    $\boldsymbol{u}_j \leftarrow \frac{\boldsymbol{u}_j}{\max(\|\boldsymbol{u}_j\|_2, 1)}$;
end
Output: Newly updated $\boldsymbol{U}$
```

Algorithm 2: Online Low Rank Matrix Completion

```
Input: The observed data that will be revealed sequentially
       $\{\boldsymbol{y}_1, ..., \boldsymbol{y}_T\}$, and the corresponding support
       $\{\boldsymbol{w}_1, ..., \boldsymbol{w}_T\}, \boldsymbol{U}_0 \in \mathbb{R}^{m\times r}$.
$\forall k \in \{1, ..., m\}$ $\boldsymbol{A_k} \leftarrow \boldsymbol{0} \in \mathbb{R}^{r\times r}$; $\boldsymbol{b_k} \leftarrow \boldsymbol{0} \in \mathbb{R}^r$;
for $t = 1$ to $T$ do
    $\boldsymbol{v}_t \leftarrow \mathrm{argmin}_{\boldsymbol{v}_t} \|\boldsymbol{w}_t \odot (\boldsymbol{y}_t - \boldsymbol{U}_{t-1}\boldsymbol{v}_t)\|_2^2$ ;
    for $k = 1$ to $m$ do
        $\boldsymbol{A_k}^{(t)} \leftarrow \boldsymbol{A_k}^{(t-1)} + [\boldsymbol{w}_t]_k \times \boldsymbol{v}_t\boldsymbol{v}_t^T$;
        $\boldsymbol{b_k}^{(t)} \leftarrow \boldsymbol{b_k}^{(t-1)} + ([\boldsymbol{w}_t]_k \times [\boldsymbol{y}_t]_k)\boldsymbol{v}_t^T$;
    end
    Update $\boldsymbol{U}^{(t)}$ using $\boldsymbol{U}^{(t-1)}$ as warm restart by Algorithm 1;
end
Output: $\boldsymbol{U}^{(T)}$
```

Prior to analyzing the convergence of the designed online solver for LRMC, we state some assumptions under which our analysis holds. The first assumption is that the data are bounded, which is natural for the realistic data, such as audio signals, images and videos. The second is the uniqueness of the solution to $\boldsymbol{v}_t$ (5) is satisfied. The solution is unique if and only if $\boldsymbol{D}_t\boldsymbol{U}^{(t-1)}$ is full column rank. In practice, this condition is easy to meet as $r \ll m$ and the fraction of missing elements in $\boldsymbol{y}_i$ is not extremely large typically. The third is that the quadratic surrogate functions $\mathcal{S}_t(\boldsymbol{U}_{j\cdot}^{(t)})$ are strictly convex with lower-bounded Hessians. This hypothesis is experimentally verified after processing several data points with reasonable initialization (we will see the random initialization also works sufficiently well in Sec. 3), which has also been experimentally verified on the task of online dictionary learning [Mairal *et al.*, 2010]. The convergence property of the proposed Algorithm 2 is given by Theorem 1.

Theorem 1. *Assume the observations are always bounded. Given the rank of the optimal solution to* (4) *provided as* r, *and the solution* $\boldsymbol{U}^{(t)} \in \mathbb{R}^{m\times r}$ *obtained by Algorithm 2 is full column rank, then* $\boldsymbol{U}^{(t)}$ *converges to at least a stationary point of the problem* (4) *asymptotically.*

Proof Sketch. *From formula* (7)*, we can obtain a set of row-wise sub-problems as given in* (8)*, which are independent to each other. It indicates that the theorem can be recognized if each sub-problem holds. Please note that the separated sub-problems have the same form as the problem (6) shown in [Mairal* et al.*, 2010], and thus can be proved in the way offered by [Mairal* et al.*, 2010]. Due to space limit, instead of the complete proof, we here only provide the proof sketch including three steps: 1) the surrogate function* $\mathcal{S}_t(\boldsymbol{U}_{j\cdot}^{(t)})$ *converges almost surely; 2) the variations of* $\boldsymbol{U}_{j\cdot}^{(t)}$ *are asymptotic, i.e.* $\|\boldsymbol{U}_{j\cdot}^{(t+1)} - \boldsymbol{U}_{j\cdot}^{(t)}\|_F = O(\frac{1}{t})$*; and 3)* $\mathcal{F}(\boldsymbol{U}_{j\cdot}^{(t)}) - \mathcal{S}_t(\boldsymbol{U}_{j\cdot}^{(t)}) \to 0$ *almost surely.*

2.2 Online Robust Low Rank Matrix Recovery

In many real world applications, the observed data are very likely contaminated by outliers, the support of which is usually unknown in advance. For the purpose of accurate recovery of underling matrices, it is essential to eliminate or reduce the disturbance from these outliers. Now we move on from OLRMC to its upgraded problem, say Online Robust Low Rank Matrix Recovery (ORLRMR). It is sure that, if the outlier support is at hand, ORLRMR degenerates to OLRMC. Thus, compared with OLRMC, ORLRMR requires to detect outliers from observations automatically and robustly.

Recall that each entry $[\boldsymbol{y}_i]_j (i = 1, ..., n; j = 1, 2, ..., m)$ of the sample $\boldsymbol{y}_i$ can be modeled as $[\boldsymbol{y}_i]_j = \boldsymbol{U}_{j\cdot}\boldsymbol{v}_i + [\boldsymbol{e}_i]_j$. Suppose observations are always bounded, an outlier can be with any value in the range with equal probability instead of either Gaussian (ℓ^2–PCA and PMF) or Laplacian (ℓ^1–PCP and PRMF). Therefore, we assume the outliers follow the uniform distribution $\frac{1}{\epsilon}$. As for the residuals caused by other factors, we simply assume they (approximately) follow a Gaussian distribution $\mathcal{N}(0, \sigma_i^2)$. Let π_i^u and π_i^g, such that $\pi_i^u + \pi_i^g = 1$, respectively denote the percentages of outlier and the other in the i^{th} sample, which are also unknown in advance. As a result, each $[\boldsymbol{e}_i]_j$ can be viewed as a sample from a mixture model of distributions with probability $p([\boldsymbol{e}_i]_j) = \pi_i^u\mathcal{N}(0, \sigma_i^2) + \pi_i^g\frac{1}{\epsilon}$. The likelihood of $\boldsymbol{Y}$ can be written in the following form:

$$p(\boldsymbol{Y}|\boldsymbol{U}, \boldsymbol{V}, \Theta) = \prod_{i,j}(\pi_i^u\mathcal{N}([\boldsymbol{y}_i]_j|\boldsymbol{U}_{j\cdot}\boldsymbol{v}_i, \Theta_i) + \frac{\pi_i^g}{\epsilon}), \quad (9)$$

where $\Theta_i = \{\sigma_i^2, \pi_i^u, \pi_i^g\}$ is a parameter vector. The nonnegative log-likelihood function of (9) is:

$$\mathcal{L}(\boldsymbol{U}, \boldsymbol{V}, \Theta) \equiv -\sum_{i,j}\log\left(\pi_i^u\mathcal{N}([\boldsymbol{y}_i]_j|\boldsymbol{U}_{j\cdot}\boldsymbol{v}_i, \Theta_i) + \frac{\pi_i^g}{\epsilon}\right). \quad (10)$$

So far, the outlier support has not been explicitly touched upon. But, as can be seen from (10), it is a typical mixture model that can be immediately minimized by Expectation-Maximization algorithms. This reminds us that there involve latent variables $[\boldsymbol{z}_i^k]_j \in \{0, 1\}$ with $\sum_{k=1}^K[\boldsymbol{z}_i^k]_j = 1$, which indicate the assignment of $[\boldsymbol{e}_i]_j$ to a specific component of the mixture containing K models in total. In this task, $\boldsymbol{z}_i^g$ that corresponds the Gaussian component, can be employed as the desired support. To be explicit with respect to the latent variables, and make the model more compatible to additional constraints, by applying Proposition 1, we obtain the

Algorithm 3: Online Robust LRMR

```
Input: The observed data that will be revealed sequentially
       $\{\boldsymbol{y}_1, ..., \boldsymbol{y}_T\}$, and non-negative $\lambda$.
$\forall k \in \{1, ..., m\}$ $\boldsymbol{A_k} \leftarrow \boldsymbol{0} \in \mathbb{R}^{r\times r}$; $\boldsymbol{b_k} \leftarrow \boldsymbol{0} \in \mathbb{R}^r$;
for $t = 1$ to $T$ do
    while not converged do
        Update $\Phi_t$ via (13), (14);
        Update $\Theta_t$ via (15);
        Update $\hat{\boldsymbol{v}}_t$ via (16);
    end
    $\boldsymbol{w}_t \leftarrow \text{binarize}(\Phi_t^g)$;
    $\boldsymbol{v}_t \leftarrow \text{argmin}_{\boldsymbol{v}_t} \|\boldsymbol{w}_t \odot (\boldsymbol{y}_t - \boldsymbol{U}_{t-1}\boldsymbol{v}_t)\|_2^2$ ;
    for $k = 1$ to $m$ do
        $\boldsymbol{A_k}^{(t)} \leftarrow \boldsymbol{A_k}^{(t-1)} + [\boldsymbol{w}_t]_k \times \boldsymbol{v}_t\boldsymbol{v}_t^T$;
        $\boldsymbol{b_k}^{(t)} \leftarrow \boldsymbol{b_k}^{(t-1)} + ([\boldsymbol{w}_t]_k \times [\boldsymbol{y}_t]_k)\boldsymbol{v}_t^T$;
    end
    Update $\boldsymbol{U}_t$ via Algorithm 1;
end
Output: Newly updated $\boldsymbol{U}$
```

cost function like:

$$\mathcal{C}(\boldsymbol{U}, \boldsymbol{V}, \Theta, \Phi) \equiv \sum_{i,j} \begin{pmatrix} [\Phi_i^g]_j \log \dfrac{[\Phi_i^g]_j}{\pi_i^g \mathcal{N}([\boldsymbol{y}_i]_j | \boldsymbol{U}_{j\cdot}\boldsymbol{v}_i, \sigma_i^2)} \\ + [\Phi_i^u]_j \log \dfrac{\epsilon[\Phi_i^u]_j}{\pi_i^g} \end{pmatrix}. \tag{11}$$

Proposition 1. *Given two functions $\pi_k(x) > 0$ and $p_k(x) > 0$, we have the following [Liu* et al.*, 2013; Guo* et al.*, 2014b]:*

$$-\log \sum_{k=1}^{K} \pi_k(x)p_k(x) = \min_{\Phi(x)\in\Delta_+} \sum_{k=1}^{K} \Phi_k(x) \log \frac{\Phi_k(x)}{\pi_k(x)p_k(x)},$$

where $\Phi(x) = \{\Phi_1(x), ..., \Phi_K(x)\}$ are hidden variables, and $\Delta_+ = \{\Phi(x) : 0 < \Phi_k(x) < 1, and \sum_{k=1}^{K} \Phi_k(x) = 1\}$ is a convex relaxation of a characteristic function decomposition.

Proof. The proof can be easily done via Lagrangian Multiplier method. Due to the limited space, we omit it here. □

With the modification on (10), the cost function (11) is explicit to the latent variables Φ, and more importantly, is ready to take into account extra constraints on Φ, such as sparsity of outlier and smoothness of support. In the following, for instance, we introduce the sparsity of outlier into (12), *i.e.* $\|\Phi^u\|_0$, which is a commonly used prior to a variety of real cases. It is well known that the ℓ^0 norm is non-convex and usually replaced with the ℓ^1. As the elements in Φ distribute between 0 and 1, by slight algebraic transformation, we have the following formula:

$$\hat{\mathcal{C}} \equiv \sum_i \sum_j \begin{pmatrix} [\Phi_i^g]_j \log \dfrac{[\Phi_i^g]_j}{\pi_i^g \mathcal{N}([\boldsymbol{y}_i]_j | \boldsymbol{U}_{j\cdot}\boldsymbol{v}_i, \sigma_i^2)} \\ + [\Phi_i^u]_j \log \dfrac{\epsilon[\Phi_i^u]_j}{\pi_i^g} + \lambda[\Phi_i^u]_j \end{pmatrix}, \tag{12}$$

where λ is a coefficient controlling the weight of outlier sparsity. Please note that we use $\hat{\mathcal{C}}$ to present $\hat{\mathcal{C}}(\boldsymbol{U}, \boldsymbol{V}, \Theta, \Phi)$ for brevity. Following the same spirit of solving the problem (2), we can seek the solution of (12) in an online manner.

When a new data vector $\boldsymbol{y_t}$ comes, we first calculate Φ_t, Θ_t and $\boldsymbol{v}_t$ over the previously updated $\boldsymbol{U}^{(t-1)}$ by the alternating minimizing strategy as follows:

$$[\hat{\Phi_t^u}]_j = \underset{[\Phi_t^u]_j}{\text{argmin}} [\Phi_t^u]_j \log \frac{[\Phi_t^u]_j\epsilon}{\pi_t^u} + \lambda[\Phi_t^u]_j = \frac{\pi_t^u}{\epsilon\exp(1+\lambda)}.$$

$$\begin{aligned} [\hat{\Phi_t^g}]_j &= \underset{[\Phi_t^g]_j}{\text{argmin}} [\Phi_t^g]_j \log \frac{[\Phi_t^g]_j}{\pi_t^g \mathcal{N}([\boldsymbol{y}_t]_j | \boldsymbol{U}_{j\cdot}^{(t-1)}\boldsymbol{v}_t, \sigma_t^2)} \\ &= \frac{\pi_t^g}{\sqrt{2\pi\sigma_t^2}} \exp\left(-1 - \frac{([\boldsymbol{y}_t]_j - \boldsymbol{U}_{j\cdot}^{(t-1)}\boldsymbol{v}_t)^2}{2\sigma_t^2} \right). \end{aligned} \tag{13}$$

To further enforce the summation of $[\Phi_t^u]_j$ and $[\Phi_t^g]_j$ to be 1, simply normalizing them gives:

$$[\Phi_t^u]_j = \frac{[\hat{\Phi_t^u}]_j}{[\hat{\Phi_t^g}]_j + [\hat{\Phi_t^u}]_j}; \quad [\Phi_t^g]_j = \frac{[\hat{\Phi_t^g}]_j}{[\hat{\Phi_t^g}]_j + [\hat{\Phi_t^u}]_j}. \tag{14}$$

It is worth noting that the above processing corresponds to the E step of EM. Next, we focus on updating the parameters of the mixed model including σ_t^2, π_t^u and π_t^g, which can be achieved via [Dempster *et al.*, 1977]:

$$\begin{aligned} \pi_t^u &= \frac{\sum_j [\Phi_t^u]_j}{m}; \quad \pi_t^g = \frac{\sum_j [\Phi_t^g]_j}{m}; \\ \sigma_t^2 &= \frac{\sum_j [\Phi_t^g]_j ([\boldsymbol{y}_t]_j - \boldsymbol{U}_{j\cdot}^{(t-1)}\boldsymbol{v}_t)^2}{\sum_j [\Phi_t^g]_j}. \end{aligned} \tag{15}$$

In turn, $\boldsymbol{v}_t$ can be obtained through optimizing the following:

$$\begin{aligned} \boldsymbol{v}_t &= \underset{\boldsymbol{v}_t}{\text{argmin}} \sum_{j=1}^{m} [\Phi_t^g]_j \log \frac{[\Phi_t^g]_j}{\pi_t^g \mathcal{N}([\boldsymbol{y}_t]_j | \boldsymbol{U}_{j\cdot}^{(t-1)}\boldsymbol{v}_t, \sigma_t^2)} \\ &= \underset{\boldsymbol{v}_t}{\text{argmin}} \|\Omega_t \odot (\boldsymbol{y}_t - \boldsymbol{U}^{(t-1)}\boldsymbol{v}_t)\|_2^2, \end{aligned} \tag{16}$$

where $[\Omega_t]_j \equiv \sqrt{[\Phi_t^g]_j}$. This is in the same form with (5), and can be solved in the same way. Please notice that the updating of Θ_t and $\boldsymbol{v}_t$ can be viewed as the M step of EM. Φ_t, Θ_t and $\boldsymbol{v}_t$ are iteratively updated via the procedure above until converge. Moreover, the element of Φ_t is in $(0, 1)$ instead of binary, although very close to either 0 or 1. To make the weight Φ_t^g consistent with the binary support (LRMC), we pre-define a threshold (*e.g.* 0.5) to binarize Φ_t^g as $\hat{\boldsymbol{w}}_t$. With the well constructed support $\hat{\boldsymbol{w}}_t$ and $\boldsymbol{v}_t$, we are ready to update the basis $\boldsymbol{U}$. For completeness, the solver of ORLRMF is summarized in Algorithm 3.

3 Experimental Verification

In this section, we conduct experiments on synthetic data to reveal the convergence speed and the robustness to outliers of the proposed ORLRMR, and on real data for showing the advantages of our method compared against alternative approaches. Because OLRMC is a special case of ORLRMR, we concentrate on ORLRMR. Please note that ORLRMR has one free parameter λ for balancing the outlier sparsity. From Eqs. (12) and (13), we can see that the effect of λ is actually to pull down $\frac{1}{\epsilon}$ to $\frac{1}{\epsilon\exp(\lambda)}$. No doubt that when λ grows, the probability for a residual of being assigned to the outlier component decreases. But, the margin between the tail of Gaussian distribution and $\frac{1}{\epsilon}$ is usually large, therefore ORLRMR can perform stably with a relatively wide range of λ, especially for Gaussian noises with small variances. For the experiments shown in this paper, we uniformly set λ to 2. The experiments are carried out on a MacBook Pro running OS X 64bit operating system with Intel Core i.7 2.8GHz CPU and 16GB RAM. Our code is implemented in Matlab.

3.1 Synthetic Data

Data Preparation. For fair comparison, we generate data in a similar way with PCP [Candès *et al.*, 2011] and ORPCA [Feng *et al.*, 2013]. That is, a set of n clean data points is produced by $\boldsymbol{Y} \in \mathbb{R}^{m\times n} = \boldsymbol{U}\boldsymbol{V}^T$, where $\boldsymbol{U} \in \mathbb{R}^{m\times r}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times r}$. The entries of both $\boldsymbol{U}$ and $\boldsymbol{V}$ are i.i.d. sampled from the $\mathcal{N}(0, 1/n)$ distribution, and then the data are projected into $[-1, 1]$. The basis of the generated subspace is $\boldsymbol{U}$, and r is the intrinsic rank (dimension) of the spanned subspace. As for the corruption, we contaminate the data by replacing a fraction p_o of $\boldsymbol{Y}$ with outliers drawn from a uniform distribution over the interval of $[-1, 1]$.

Task and Quantitative Metric. We evaluate the ability of correctly recovering the subspace of corrupted observations under varying settings of the error density and intrinsic subspace rank. The $\boldsymbol{U}^{(0)}$ that feeds to the competitors is randomly initialized in the same fashion as data generation. To measure the similarity between the recovered subspace $\hat{\boldsymbol{U}}$ and the ground-truth $\boldsymbol{U}$, the Expressed Variance (E.V.) [Xu *et al.*, 2010; Feng *et al.*, 2013] is employed as our metric, the definition of which is:

$$\mathrm{E.V.}(\hat{\boldsymbol{U}}, \boldsymbol{U}) \equiv \frac{\mathrm{tr}\left(\mathrm{orth}(\hat{\boldsymbol{U}})^T\boldsymbol{U}\boldsymbol{U}^T\,\mathrm{orth}(\hat{\boldsymbol{U}})\right)}{\mathrm{tr}(\boldsymbol{U}\boldsymbol{U}^T)}, \quad (17)$$

where $\mathrm{orth}(\cdot)$ and $\mathrm{tr}(\cdot)$ stand for the orthogonalizing and trace operators, respectively. The averaged results from 10 independent trials are finally reported.

Competitors. We compare our ORLRMR with recently proposed online techniques, *i.e.* GRASTA and OR-PCA. The code of GRASTA is downloaded from the authors' website, the core of which is implemented in C++. Its most important parameter is the sub-sampling ratio, to obtain its best possible results, we turn off the sub-sampling to take into account whole information. The code of OR-PCA is not available when this paper is prepared. So we implement it in Matlab by strictly following the pseudo code given in [Feng *et al.*, 2013], and adopt the parameters suggested by the authors.

Performance Analysis. We provide the performance curves of ORLRMR, OR-PCA and GRASTA against the number of samples in Fig. 1, under the setting of $m = 400$ and $n = 10^4$, to see the robustness of the contestants to corruptions and how the performance is improved when more samples are revealed. As can be seen from the pictures, all the three methods achieve better results with more observations. For

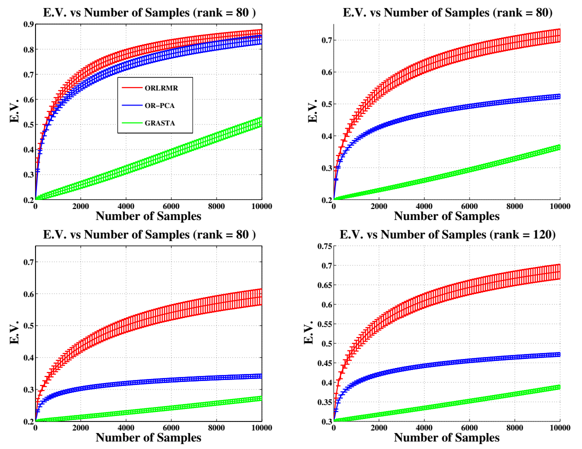

Figure 1: Expressed variance versus number of samples. **Top-Left**: intrinsic rank $r = 80$ and outlier ratio $p_o = 0.2$. **Top-Right**: $r = 80$ and $p_o = 0.3$. **Bottom-Left**: $r = 80$ and $p_o = 0.4$. **Bottom-Right**: $r = 120$ and $p_o = 0.3$.

the four cases, the performance of GRASTA is much inferior to the others, possibly because the intrinsic dimensions versus the ambient are relatively large for GRASTA, as well as the basis updating by GRASTA only takes one gradient descent step while both ORLRMR and OR-PCA have closed-form solutions. We will see that GRASTA attains reasonably well results when the intrinsic rank is relatively small later. In the case of intrinsic rank $r = 80$ and outlier ratio $p_o = 0.2$, the plots of OR-PCA and ORLRMR are very close, both of which reach about $\mathrm{E.V.} = 0.9$ after processing 10^4 samples. As the outlier ratio increases, the convergence speed drops. Even though, ORLRMR gives $\mathrm{E.V.} \approx 0.7$ and $\mathrm{E.V.} \approx 0.6$ for $p_o = 0.3$ and $p_o = 0.4$, respectively, which outperforms OR-PCA by about 0.2. This indicates that ORLRMR can bear more corruptions than the others. In addition, we show a case of $r = 120$ and $p_o = 0.3$ to view the effect of r. The curves tell us that OR-PCA and ORLRMR perform similarly with those of $r = 80$ and $p_o = 0.3$ with very slight depression, which means the change of r influences not much to them. Please note that, the E.V. of the initialized basis increases as the intrinsic rank approaches to the ambient dimension.

3.2 Real Data

Dataset. The Star dataset consists of 9 real world surveillance videos, which has a variety of scenarios including Escalator (*Esc: indoor, dynamic background, crowd*), Hall (*Hal: indoor, static background, crowd*), BootStrap (*BS: indoor, static background, crowd*), Lobby (*Lob: indoor, light switch, people*), ShoppingMall (*SM: indoor, static background, people*), Campus (*Cam: outdoor, dynamic background, cars*), WaterSurface (*WS: outdoor, dynamic background, lingering person*), Fountain (*Fou: outdoor, dynamic background, people*), and Curtain (*Cur: indoor, dynamic background, people*).

Task and Quantitative Metric. This task is to separate foregrounds from surveillance videos, in which the ground-truth foreground is given but the ground-truth background model is not available. Moreover, different to the simulation, the

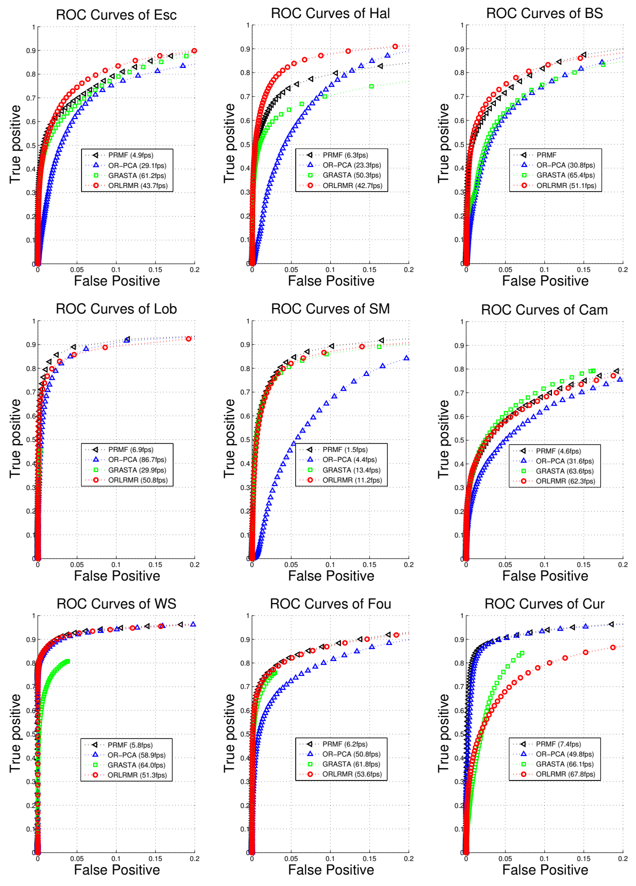

Figure 2: Quantitative results on the Star dataset.

background may comprise dynamic factors, such as fountains and waving curtains. To assess the performance on foreground detection, we use the Receiver Operating Characteristic (ROC) curve as our metric. As the rank of the background of surveillance videos is typically small, we empirically set it to 5 for all the 9 sequences.

Competitors. Besides GRASTA and OR-PCA, we further employ a batch method, *i.e.* PRMF [Wang *et al.*, 2012], as our reference. PRMF is a probabilistic approach to robust matrix factorization, which holds an equivalent relation with PCP. PCP [Candès *et al.*, 2011] is not used, mainly because its costs in both memory and time are too expensive to process long sequences. The Matlab code of PRMF is downloaded from the authors' website, the parameters are set as the values provided in the implementation package released by the authors. In addition, to obtain best possible performance of GRASTA and OR-PCA, the first 5 frames of each sequence are used to initialize the basis for GRASTA and OR-PCA[2]. To show the insensitivity of ORLRMR to initialization, its basis is randomly generated as the previous experiment do.

[2]Without a good initialization, OR-PCA can not give reasonable results in this experiment. The reason possibly is that the subspace rank versus the ambient dimension is too small and the number of samples is inadequate for converging.

Performance Analysis. Figure 2 summarizes the ROC plots for the 9 videos. From the top row, we can see that our ORLRMR outperforms GRASTA, OR-PCA and even the batch approach PRMF. The reason is that these three videos are in presence of crowd, and ORLRMR can handle heavy outliers better than the others. For the rest videos, PRMF attains the best results (except for the Campus sequence, in which GRASTA slightly exceeds PRMF), while ORLRMR competes very favorably with PRMF in most cases. The only exception is the Curtain video, the background of which contains intensively moving curtains. Although the ground-truth foreground excludes these curtains, from the perspective of low rank recovery, they are more like outliers, and ORLRMR treats these as outliers. Please notice that GRASTA performs reasonably well on these real videos, since the subspace dimension is relatively small. As regards OR-PCA, its inferior performance for most cases may be because of the large fractions of foreground and/or the insufficient number of streaming samples for achieving convergence. Together with the simulation, we can conclude that ORLRMR is better at dealing with heavily polluted data and is of faster convergence speed than OR-PCA. In terms of time, PRMF costs, without surprise, the most (5.7 frames per second on average over the 9 sequences) among the contestants, due to its batch nature. ORLRMR achieves 48.3 fps, which outperforms OR-PAC 40.6 fps and is competitive with GRASTA 52.9 fps, even though the core of GRASTA is implemented in C++. The fps numbers corresponding to each sequence can be found in the legends of Fig. 2. We would like to emphasize that the experiments in this part do not intend to verify our ORLRMR wins over those special designs on the task of foreground detection. The aim is to demonstrate the merits of our general model without involving domain knowledge. It is certain that equipping with proper priors for specific problems can further improve the performance of any of the general models like GRASTA, OR-PCA, ORLRMR and PRMF.

4 Conclusion

In this paper, we have proposed to address the low rank matrix completion and recovery tasks in an online fashion, which scales up gracefully to large data sets. The mixture model consisted of a Gaussian and a uniform distributed components has shown its significant improvement on the robustness to heavy outliers. The theoretical and experimental results have demonstrated the properties and advantages of our scheme compared alternative methods. It is worth noting that our technical framework is ready to embrace additional specific constraints for further boosting the performance on different tasks. For instance, the spatial smoothness can be introduced to accurately detect foregrounds in surveillance videos. It is positive that our proposed technique can be widely applied to tabulated data processing applications.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61402467 and in part by the Excellent Young Talent Program through the Institute of Information Engineering, Chinese Academy of Sciences.

References

[Balzano *et al.*, 2011] L. Balzano, R. Nowak, and B. Recht. Online identification and tracking of subspaces from highly incomplete information. *arXiv:1006.4046*, 2011.

[Bertsekas, 1999] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[Bottou and Bousquet, 2008] L. Bottou and O. Bousquet. The trade-offs of large scale learning. In *Proceedings of NIPS*, pages 161–168, 2008.

[Candès *et al.*, 2011] E. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):1–37, 2011.

[Chandrasekaran *et al.*, 2011] V. Chandrasekaran, S. Sanghavi, P. Parrilo, and A. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011.

[Chen *et al.*, 2014] C. Chen, X. Zheng, Y. Wang, F. Hong, and Z. Lin. Context-aware collaborative topic regression with social matrix factorization for recommender systems. In *Proceedings of AAAI*, pages 9–15, 2014.

[Dempster *et al.*, 1977] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[Feng *et al.*, 2013] J. Feng, H. Xu, and S. Yan. Online robust pca via stochastic optimization. In *Proceedings of NIPS*, pages 1–9, 2013.

[Gu *et al.*, 2014] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with applications to image denoising. In *Proceedings of CVPR*, pages 2862–2869, 2014.

[Guo *et al.*, 2013] X. Guo, X. Cao, X. Chen, and Y. Ma. Video editing with temporal, spatial and appearance consistency. In *Proceedings of CVPR*, pages 2283–2290, 2013.

[Guo *et al.*, 2014a] X. Guo, X. Cao, and Y. Ma. Robust separation of reflection from multiple images. In *Proceedings of CVPR*, pages 2195–2202, 2014.

[Guo *et al.*, 2014b] X. Guo, X. Wang, L. Yang, X. Cao, and Y. Ma. Robust foreground detection using smoothness and arbitrariness constraints. In *Proceedings of ECCV*, pages 535–550, 2014.

[He *et al.*, 2012] J. He, L. Balzano, and A. Szlam. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *Proceedings of CVPR*, pages 1568–1575, 2012.

[Jamali and Ester, 2011] M. Jamali and M. Ester. A transitivity aware matrix factorization model for recommendation in social networks. In *Proceedings of IJCAI*, pages 2644–2649, 2011.

[Liu *et al.*, 2013] J. Liu, X. Tai, H. Huang, and Z. Huan. A weighted dictionary learning model for denoising images corrupted by mixed noise. *IEEE TIP*, 22(3):1108–1120, 2013.

[Mairal *et al.*, 2010] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *JMLR*, 11(1):19–60, 2010.

[Park *et al.*, 2013] S. Park, Y. Kim, and S. Choi. Hierarchical bayesian matrix factorization with side information. In *Proceedings of IJCAI*, pages 1593–1599, 2013.

[Pearson, 1901] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 1901.

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

[Salakhutdinov and Mnih, 2008] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Proceedings of NIPS*, pages 1–8, 2008.

[Shen and Wu, 2012] X. Shen and Y. Wu. A unified approach to salient object detection via low rank matrix recovery. In *Proceedings of CVPR*, pages 853–860, 2012.

[Shi *et al.*, 2013] J. Shi, N. Wang, Y. Xia, D. Yeung, I. King, and J. Jia. SCMF: Spase covariance matrix factorization for collaborative filtering. In *Proceedings of IJCAI*, 2013.

[Wang and Zhang, 2012] S. Wang and Z. Zhang. Colorization by matrix completion. In *Proceedings of AAAI*, pages 1169–1174, 2012.

[Wang *et al.*, 2012] N. Wang, T. Yao, J. Wang, and D. Yeung. A probabilistic approach to robust matrix factorization. In *Proceedings of ECCV*, pages 126–139, 2012.

[Xu *et al.*, 2010] H. Xu, C. Caramanis, and S. Sanghavi. Principle component analysis with contaminated data: The high dimensional case. In *Proceedings of COLT*, 2010.

[Zhang *et al.*, 2012] Z. Zhang, A. Ganesh, X. Liang, and Y. Ma. TILT: Transform invariant low-rank textures. *IJCV*, 99(1):1–24, 2012.