Simultaneous Recovery of A Series of Low-rank Matrices by Locally Weighted Matrix Smoothing

Liangbei Xu and Mark A. Davenport
Georgia Institute of Technology
Email: {lxu66, mdav}@gatech.edu

***Abstract*—Low-rank matrix factorizations arise in a wide variety of applications – including recommendation systems, topic models, and source separation, to name just a few. There exist both empirical and theoretical results showing that, under some dynamic models, significant improvements can be obtained by incorporating temporal information and allowing for the possibility that the underlying matrix is time-varying. In this paper we propose the S-LOWEMS estimator, which simultaneously recovers a series of low-rank matrices based on the locally weighted matrix smoothing (LOWEMS) framework. Our synthetic simulations and real world experiments show that, compared to the original LOWEMS estimator, the proposed S-LOWEMS estimator not only recovers a series of low-rank matrices with a small computational overhead, but also improves the recovery accuracy and reduces the sample complexity.**

I. Introduction

In recent years there has been a significant amount of progress in our understanding of how to recover a low-rank matrix X from incomplete observations even when the number of observations is much less than the number of entries in X. (See [1] for an overview of this literature.) Nearly all of this existing work assumes that the underlying low-rank matrix X remains fixed throughout the measurement process. In many practical applications, this is a tremendous limitation. For example, users' preferences for various items may change (sometimes quite dramatically) over time. Modelling such drift of user's preference has been proposed in the context of both music and movie recommendation as a way to achieve higher recommendation accuracy [2]. Another example in signal processing is dynamic non-negative matrix factorization for the blind signal separation problem [3]. In these and many other applications, explicitly modeling the dynamic structure in the data has led to superior empirical performance.

There has been some recent theoretical progress in the context of dynamic low-rank matrix recovery, including [4], which provides recovery guarantees when one of the factor matrices of the underlying low-rank matrix is changing over time and [5], which uses a temporal regularizer to exploit the temporal dependence. In this paper we extend the approach of [4] by designing a two-stage estimator in the context of estimating a sequence of low-rank matrices simultaneously under a discrete random walk model.

II. Problem Setup

The underlying assumption throughout this paper is that our low-rank matrix is changing over time throughout the measurement process, that is, rather than observations of a fixed matrix X, we are given observations of a sequence of (related) matrices $X^1, \ldots, X^d$. Note that we can impose the low-rank constraint explicitly by factorizing X^t as

$$X^t = U^t \left(V^t\right)^T,$$

where $U^t \in \mathbb{R}^{n_1 \times r}, V^t \in \mathbb{R}^{n_2 \times r}$.

In general both U^t and V^t may be changing over time. However, in many applications it is reasonable to assume that only one set of factors is changing. For example, in a recommendation system where our matrix represents user preferences, if the rows correspond to items and the columns correspond to users, then U^t contains the latent properties of the items and V^t models the latent preferences of the users. In this context it is reasonable to assume that only V^t changes over time (see [2]), and that there is a fixed matrix U. In fact, without loss of generality we may always assume that U is fixed and orthonormal (since otherwise one can find another equivalent factorization satisfying this assumption, e.g., via a QR factorization of U).

Now suppose that V satisfies the following discrete random walk model up to d time steps:

$$V^{t+1} = V^t + \epsilon^t, t = 1, \ldots, d-1, \tag{1}$$

where ϵ^t is the perturbation (or process) noise. We observe X^t via the following linear measurement model:

$$y^t = \mathcal{A}^t(X^t) + z^t, \quad y^t, z^t \in \mathbb{R}^{m_0}, \tag{2}$$

where z^t is measurement noise, m_0 is the number of measurements per time step, and $\mathcal{A}^t$ is a measurement operator, which is a linear mapping from $\mathbb{R}^{n_1 \times n_2}$ to $\mathbb{R}^{m_0}$. We can represent the i^{th} entry of $\mathcal{A}(X)$ by $[\mathcal{A}(X)]_i = \langle A_i, X \rangle$, where A_i represents i^{th} sensing matrix A_i. In this paper we consider a special case where $\mathcal{A}$ samples a subset of entries of X, it is known as the *matrix completion* problem.

Finally, we will also assume that both ϵ^t and z^t are i.i.d. zero-mean Gaussian noise with variance σ_2^2 and σ_1^2 respectively. Our problem is to recover the sequence $\{X^t\}_{t=1}^d$ from $\{y^t\}_{t=1}^d$.

III. S-LOWEMS estimator

A. Maximum likelihood estimator

The first approach is to consider the recovery problem as a latent factor learning problem. A maximum likelihood estimator (MLE) is given by minimizing the following negative log-likelihood:

$$\mathcal{L}(U, V^1, \ldots, V^d) = \frac{1}{\sigma_1^2} \sum_{t=1}^{d} \left\| \mathcal{A}^t \left(UV^t \right) - y^t \right\|_2^2 + \frac{1}{\sigma_2^2} \sum_{t=2}^{d} \left\| V^t - V^{t-1} \right\|_F^2 . \tag{3}$$

The above cost function consists of two terms: the first term quantifies data fidelity and the second term quantifies the dynamic constraint on V. Although minimizing (3) is a nonconvex optimization problem, we can attempt to solve it via the alternating least squares (ALS) algorithm over U and $\{V^t\}_{t=1}^d$. Note however that in this case the convergence of the ALS algorithm is not (known to be) guaranteed and the computational burden is quite heavy (especially when d is large, since we need to update all V^t's at each update).

B. A fast estimator based on weighted smoothing

In this section we use the idea of weighted smoothing from [4] to form a fast estimator of $\{X^t\}_{t=1}^d$. We first introduce the LOWEMS estimator proposed in [4], which is an algorithm that aims to produce an estimate of only X^d from $\{y^t\}_{t=1}^d$. The LOWEMS estimator consists of solving the following optimization program:

$$\widehat{X}^d = \underset{X \in \mathbb{C}(r)}{\arg\min} \frac{1}{2} \sum_{t=1}^{d} w_t \left\| \mathcal{A}^t (X) - y^t \right\|_2^2, \tag{4}$$

where $\mathbb{C}(r) = \{X \in \mathbb{R}^{n_1 \times n_2} : \text{rank}(X) \leq r\}$, and $\{w_t\}_{t=1}^d$ are non-negative weights with constraint $\sum_{t=1}^d w_t = 1$. If we define $\kappa := \sigma_2^2/\sigma_1^2$ and set $p_t = (d-t), 1 \leq t \leq d$, then one can calculate the optimal weights as [4]:

$$w_t^* = \frac{1}{\sum_{i=1}^d \frac{1}{1+p_i\kappa}} \frac{1}{1+p_t\kappa}, \quad 1 \leq t \leq d. \tag{5}$$

The parameter κ measures how strong the perturbation noise is compared to the observation noise.

Note that one can modify LOWEMS to recover X^s for any $s \in [d]$ by solving the following similar program:

$$\widehat{X}^s = \underset{X \in \mathbb{C}(r)}{\arg\min} \frac{1}{2} \sum_{t=1}^{d} w_t^s \left\| \mathcal{A}^t (X) - y^t \right\|_2^2, \tag{6}$$

where $\{w_t^s\}_{t=1}^d$ are a different set of weights to be used when estimating X^s. Following similar arguments in [4], the optimal weights in this case are:

$$w_t^{s*} = \frac{1}{\sum_{i=1}^d \frac{1}{1+p_i^s\kappa}} \frac{1}{1+p_t^s\kappa}, \quad 1 \leq t \leq d, \tag{7}$$

where $p_t^s = |t-s|$.

A naïve extension of the LOWEMS method to recover X^s for all $s \in [d]$ is to perform program (6) independently for each $s \in [d]$. However this approach does not take into account the fact that for all $s \in [d]$, X^s should share the same U. This clearly leaves some room for potential improvement. Moreover, because the weights in (7) are selected specifically to minimize the recovery error for a particular X^s, the weights necessarily "downweight" previous/future observations. This can be helpful in obtaining a more accurate estimate of V^s, but this can actually be harmful in terms of our estimate of U (since it is essentially using only a small subset of the data in its estimate).

Inspired by this observation, we consider an alternative method which, although still quite simple, has the potential to improve on the naïve approach described above. Specifically, we conjecture that an equal weighting will yield an improved estimate of U (or more precisely, the column space of U) compared to the results of using (7) for any particular choice of s. Thus, we first estimate U from $\{y^t\}_{t=1}^d$, and we then follow this step by estimating $\{V^t\}_{t=1}^d$ by solving (6) using (7) while holding U fixed. This approach is summarized as follows:

Algorithm 1 S-LOWEMS: Simultaneously Locally Weighted Matrix Smoothing

1: Given d, κ, $\{y^t\}_{t=1}^d$ and $\{\mathcal{A}^t\}_{t=1}^d$
2: Solve (6) with equal weights to obtain $\widehat{U}$
3: For each $s \in [d]$, solve (6) via least-squares with $U = \widehat{U}$ and w_t^{s*} in (7) to obtain $\widehat{V}^s$
4: Output the estimate $\widehat{X}^s = \widehat{U}(\widehat{V}^s)^T$ for all $s \in [d]$

Remark 1. One can solve (6) in step 2 via alternating minimization (see e.g., [6]) or gradient descent (see e.g., [7]) based on matrix factorization.

Remark 2. Compared to MLE, Algorithm 1 is solving a bi-convex relaxation of the objective in (3).

Remark 3. Compared to the naïve extension of LOWEMS, the computational complexity of S-LOWEMS is actually quite small. Instead of increasing the computational complexity over a single LOWEMS by a factor of d, we need only perform a single LOWEMS and then the only additional computational overhead involves solving d least-squares problems. The same is true when comparing to MLE-ALS; we do not need to update all V^t's at each update, which saves both storage and computation.

Remark 4. We conjecture that for each $t \in [d]$, the recovery error of S-LOWEMS is smaller than that of a single LOWEMS estimator. However, we leave the proof of this conjecture for future work.

IV. Simulations and Experienments

A. Synthetic simulations

In the following synthetic simulations we restrict our attention to matrix completion, although we expect similar

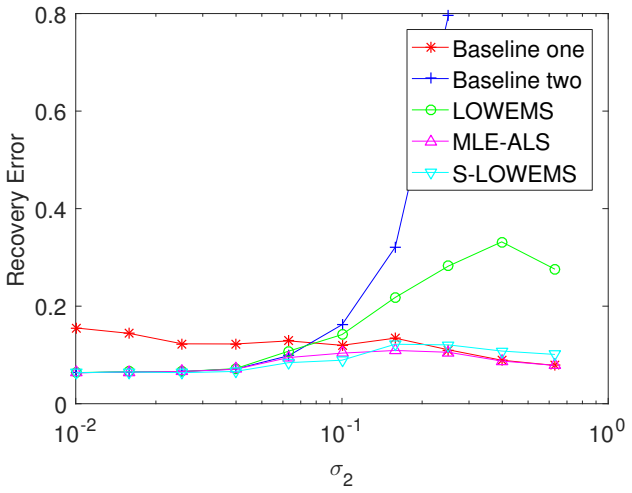

Fig. 1: Recovery error under different levels of perturbation noise.

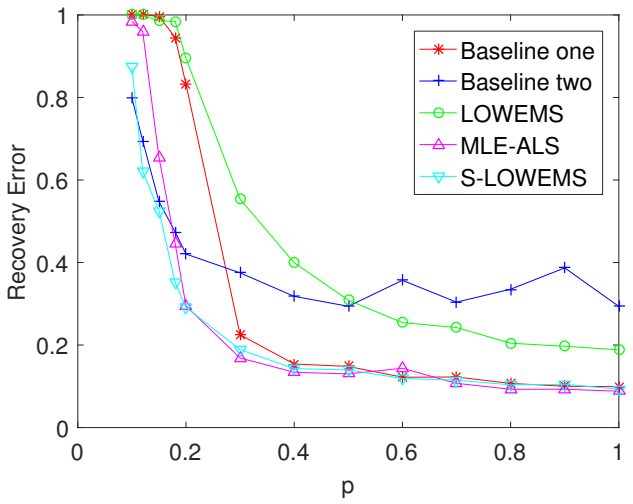

Fig. 2: Recovery error under different percentages of missing entries.

results for other observation models. We use the relative recovery error (RRE) at time d, i.e., $\|\widehat{X}^d - X^d\|_F^2/\|X^d\|_F^2$, as our recovery accuracy metric (similar results are obtained when we look at the full sequence $\{X^t\}_{t=1}^d$). We set $n_1 = 100$, $n_2 = 50$, $d = 4$ and $r = 5$. We first generate entries of U and V^d uniformly from $[-0.5, 0.5]$ and $\{V^t\}_{t=d-1}^1$ according to (1). We orthonormalize U afterwards and generate y^t according to (2). For the purpose of illustration we consider two additional baselines besides LOWEMS and MLE-ALS: **baseline one** is the MLE assuming all V^t's have no dynamic constraints (hence σ_2 is infinity); **baseline two** is the MLE assuming all V^t's are the same (hence σ_2 is zero).

1). Recovery error. We set $\sigma_1 = 0.05$. In the first simulation, we vary the perturbation noise level σ_2 while keeping $m_0 = 4000$. For every σ_2 we perform 10 trials, and show the average RRE. As one can see from Figure 1, when σ_2 is small, all the three estimator LOWEMS, MLE-ALS and S-LOWEMS achieve almost the same RRE as baseline two. As σ_2 grows, the RRE of LOWEMS will increase due to the perturbation noise. However in this case both S-LOWEMS and MLE-ALS achieve smaller RRE compared to LOWEMS. Notice that only when σ_2 is relatively large (compared to the matrix V itself, say 0.3) the RRE of S-LOWEMS is slightly larger than that of MLE-ALS. We also note that increasing perturbation noise (from 0.2 to 0.6) decreases the RRE of MLE-ALS. The reason is that the perturbation noise is large enough to help the recovery of U, and in turn reduce the RRE of recovering X^d (though we suspect this would be rare in practice).

In the second simulation, we vary the fraction of observed entries $p := m_0/(n_1 n_2)$ while keeping $\sigma_2 = 0.2$ (moderate). From Figure 2, we can see that S-LOWEMS almost achieves the best RRE (comparable to MLE-ALS) under various p.

2). Sample complexity. In this simulation we vary p to

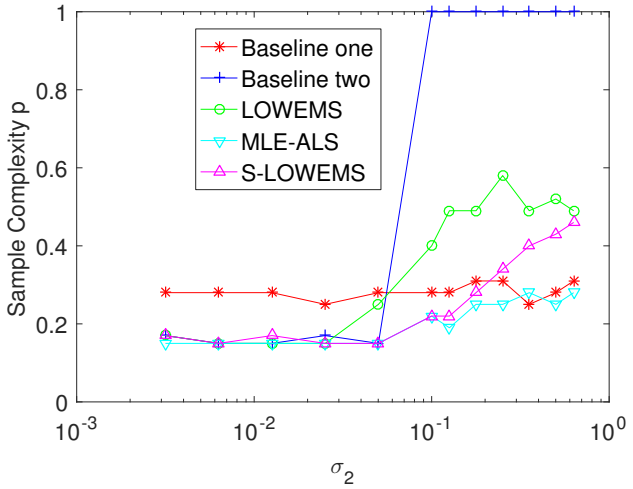

Fig. 3: Sample complexity under different levels of perturbation noise.

empirically find the minimum sample complexity required to guarantee successful recovery ($RRE \leq 0.06$). We compare the sample complexity of LOWEMS, MLE-ALS and S-LOWEMS under various σ_2 (σ_1 is set as 0.02). For a fixed σ_2, the RRE is averaged over 10 trials. From Figure 3, we can see when the perturbation noise is small (less than 0.04), the sample complexities of LOWEMS, MLE-ALS and S-LOWEMS are almost the same as baseline two. When the perturbation noise increases, the RRE of the three estimators will increase due to the perturbation noise and hence the sample complexity increases. As we can see, in this case S-LOWEMS achieves a smaller sample complexity compared to LOWEMS and a bit larger than that of MLE-ALS (the price paid for not forming a MLE).

In general, our synthetic simulations demonstrate that the proposed S-LOWEMS achieves better performance (in terms of recovery error and sample complexity) than LOWEMS, and comparable performance as MLE-ALS

with less computation and storage.

B. Real world experiments

We next test LOWEMS, MLE-ALS, and S-LOWEMS in the context of a recommendation system using the (truncated) Netflix dataset. We eliminate those movies with few ratings and those users rating few movies, and generate a truncated dataset with 3199 users, 1042 movies, and 2462840 ratings. In this case the fraction of visible entries in the rating matrix is ≈ 0.74. All the ratings are distributed over a period of 2191 days.

For the sake of robustness, we additionally impose a Frobenius norm penalty on the factor matrices U and V. We keep the latest (in time) 10% of the ratings as a testing set. The remaining ratings are split into a validation set and a training set for the purpose of cross validation. We divide the remaining ratings into $d \in \{1, 3, 6, 8\}$ bins respectively according to their timestamps so that each bin contains the same number of ratings (see Figure 4). We use 5-fold cross validation, and we keep 20% of the ratings from the d^{th} bin as a validation set. The number of latent factors r is set to 10. The Frobenius norm regularization parameter γ is set to 1. We also note that in practice one likely has no prior information on σ_1, σ_2 and hence κ. However, we use model selection techniques like cross validation to select the best κ incorporating the unknown prior information on measurement/perturbation noise. We use root mean squared error (RMSE) to measure prediction accuracy. Since alternating minimization uses a random initialization, we generate 10 test RMSE's. Figure 5 shows that all the three temporal estimators LOWEMS, MLE-ALS and S-LOWEMS improve the testing RMSE with appropriate κ compared to the static baseline (when $d = 1$). In addition, the testing RMSE of S-LOWEMS is lower than that of LOWEMS and MLE-ALS in general. Our results show that exploiting the fact that the user factor matrix V is changing yields improved prediction performance.

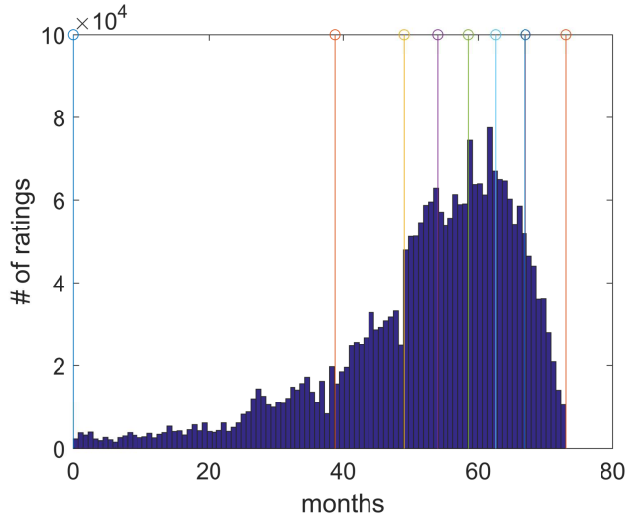

Fig. 4: Ratings divided into 7 bins (6 for training and 1 for testing) on the truncated Netflix dataset.

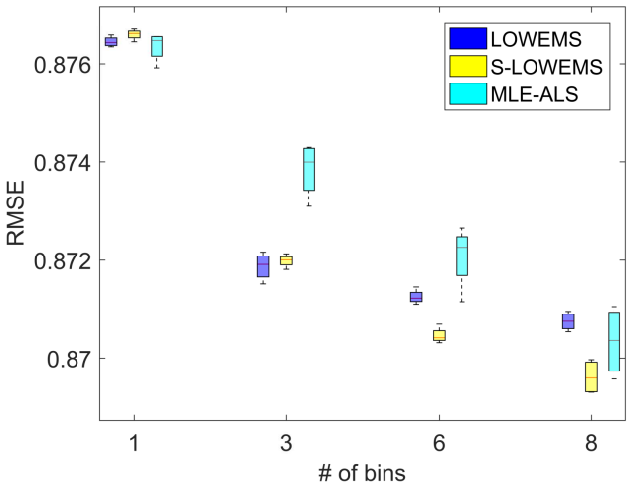

Fig. 5: Experimental results on the truncated Netflix dataset: prediction RMSE vs. number of time bins.

V. Conclusion

In this paper we investigate the low-rank matrix recovery problem under a random walk setting. We propose the S-LOWEMS estimator and analyze its recovery performance by synthetic simulations and test it on the truncated Netflix dataset. Our results show that, compared to original LOWEMS estimator, the proposed S-LOWEMS estimator not only recovers a series of low-rank matrices simultaneously with a small computational overhead, but also improves the recovery accuracy and sample complexity. Furthermore, the proposed S-LOWEMS achieves almost the same statistical efficiency as the MLE (especially when the perturbation noise is small or moderate) and consumes significantly less storage and computational resources. However our model has several limitations. For example, we assume a random walk model on one of the factor matrix. Some possible future extensions of this work include incorporating more sophisticated dynamic models, more general observation models (for example, one-bit observation in [8] and [9]), and a theoretical analysis to obtain provable recovery guarantees.

Acknowledgements

This was completed with support from the grants AFOSR FA9550-14-1-0342 and NSF CCF-1350616 as well as support from the Alfred P. Sloan Foundation.

References

[1] Mark Davenport and Justin Romberg, "An overview of low-rank matrix recovery from incomplete observations," *IEEE J. Select. Top. Signal Processing*, vol. 10, no. 4, pp. 608–622, 2016.

[2] Yehuda Koren, "Collaborative filtering with temporal dynamics," *Comm. ACM*, vol. 53, no. 4, pp. 89–97, 2010.

[3] Nasser Mohammadiha, Paris Smaragdis, Ghazaleh Panahandeh, and Simon Doclo, "A state-space approach to dynamic nonnegative matrix factorization," *IEEE Trans. Signal Processing*, vol. 63, no. 4, pp. 949–959, 2015.

[4] Liangbei Xu and Mark Davenport, "Dynamic matrix recovery from incomplete observations under an exact low-rank constraint," in *Proc. Adv. in Neural Processing Systems (NIPS)*, Barcelona, Spain, Dec. 2016.

[5] Hsiang-Fu Yu, Nikhil Rao, and Inderjit Dhillon, "Temporal regularized matrix factorization for high-dimensional time series prediction," in *Proc. Adv. in Neural Processing Systems (NIPS)*, Barcelona, Spain, Dec. 2016, pp. 847–855.

[6] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi, "Low-rank matrix completion using alternating minimization," in *Proc. ACM Symp. Theory of Comput.*, Stanford, CA, June 2013.

[7] Stephen Tu, Ross Boczar, Max Simchowitz, Mahdi Soltanolkotabi, and Benjamin Recht, "Low-rank solutions of linear matrix equations via procrustes flow," *arXiv preprint arXiv:1507.03566*, 2015.

[8] Mark Davenport, Yaniv Plan, Ewout van den Berg, and Mary Wootters, "1-bit matrix completion," *Inform. Inference*, vol. 3, no. 3, pp. 189–223, 2014.

[9] Liangbei Xu and Mark Davenport, "Dynamic one-bit matrix completion," in *Proc. Signal Processing with Adaptive Sparse Structured Representations (SPARS)*, Lisbon, Portugal, June 2017.