Dynamic one-bit matrix completion

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

I. INTRODUCTION

In recent years there has been a significant amount of progress in our understanding of how to recover a rank-$r$ matrix from incomplete observations, even when the number of observations is much less than the number of entries in the matrix. (See [4] for an overview of this literature.) In this work we consider a new setting where we aim to recover an underlying and dynamically evolving low-rank matrix from binary observations. This problem arises in a variety of applications. For example, low-rank models have been used in the context of personalized learning systems (see [6]), but in such a context we can expect a student’s knowledge/skill to change (and hopefully improve) throughout the learning process as a result of lectures, homeworks, and so on. Moreover, in such a scenario we may only have access to binary responses (right/wrong) for their answers to the assigned questions from which we hope to learn. Our goal is to unite the recent work in the area of one-bit matrix completion [3, 2, 1] with recent efforts in the context of dynamic matrix completion, including [8], which provides recovery guarantee when one of the factor matrices of the underlying low-rank matrix is changing over time and [9], which use a temporal regularizer to exploit temporal dependence.

II. THE DYNAMIC ONE-BIT MATRIX COMPLETION PROBLEM

We wish to consider the case where we have a low-rank matrix changing over time during the measurement process. At time $t$ we have a rank-$r$ matrix $X^t \in \mathbb{R}^{n_1 \times n_2}$ with factorization $X^t = U(V^t)\top$. Here we assume a random walk dynamic model on the right factor matrix $V$:

$$V^t = V^{t-1} + \epsilon^t, \quad t = 2, \ldots, d,$$

where each entry of $\epsilon^t$ follows $N(0, \sigma^2)$. We assume that we only have one-bit observations on a subset of the entries at each time-step, i.e., we observe

$$y_{i,j}^t = \begin{cases} 1 & \text{with prob. } f(X_{i,j}^t), \\ -1 & \text{with prob. } 1 - f(X_{i,j}^t) \end{cases} \quad \text{for } (i,j) \in \Omega^t,$$

where $f$ is fixed and known. Two common choices for $f$ are logistic function $f(x) = 1/(1 + e^{-x/\sigma_1})$ and the probit function $f(x) = \Phi(x/\sigma_2)$, where $\Phi(x)$ is the cumulative distribution function of standard Gaussian and $\sigma^2$ is the variance of zero-mean logistic (Gaussian) distribution. We also denote $p^t = |\Omega^t|/(n_1 n_2)$. Our goal is to recover $X^d$ from $\{Y^t, \Omega^t\}, t = 1, \ldots, d$.

III. ONE-BIT LOWEMS

The negative log-likelihood for the given problem at time $t$ is

$$\mathcal{L}(X; \Omega^t, Y^t) = -\sum_{(i,j) \in \Omega^t} \left\{ y_{i,j}^t \log(f(X_{i,j}^t)) + (1 - y_{i,j}^t) \log(1 - f(X_{i,j}^t)) \right\},$$

We additionally assume that the underlying matrix $X^d$ satisfies $\|X^d\|_{\infty} \leq \alpha$, which will make the recovery well-posed.

The proposed one-bit LOWEMS (Locally Weighted Matrix Smoothing) is formulated as the following optimization program:

$$\hat{X}^d = \arg\min_{X \in \mathcal{C}(r, \alpha)} \mathcal{F}(X) = \arg\min_{X \in \mathcal{C}(r, \alpha)} \sum_{t=1}^d w_t \mathcal{L}(X; \Omega^t, Y^t),$$

where $\mathcal{C}(r, \alpha) := \{X \in \mathbb{R}^{n_1 \times n_2} : \text{rank}(X) \leq r, \|X\|_{\infty} \leq \alpha\}$ and $\{w_t\}_{t=1}^d$ are non-negative weights. The optimal weights can be computed as follows:

$$w^*_j = \frac{1}{\sum_{i=1}^d \frac{1}{1 + (d - j)\kappa}}, \quad 1 \leq j \leq d,$$

provided $\kappa := \sigma^2 / \sigma_i^2$ is known. (See [8], Sec 3.1.)

IV. CONSTRAINED ALTERNATING GRADIENT DESCENT

The program in (4) can be reformulated as

$$\hat{X}^d = \arg\min_{X=UVT} \mathcal{F}(UV\top),$$

where $U \in \mathbb{R}^{n_1 \times r}, V \in \mathbb{R}^{n_2 \times r}$. We use alternating gradient descent to minimize $\mathcal{F}(U,V)$, which alternatively applies a gradient descent step over $U$ (or $V$) while holding $V$ (or $U$) fixed until a stopping criterion is reached. Our choice of stepsize is the safe-guard LBB (long Barzilai-Borwein) stepsize [5]. We also rescale $U$ and $V$ following the gradient descent step so that $\|UV\top\|_{\infty} \leq \alpha$ is satisfied at each step.

V. SIMULATIONS AND EXPERIMENTS

We set $n_1 = 100, n_2 = 50, d = 4, r = 2, p^t = 0.8$ for all $t$, and use the logistic function for $f$. We consider two baselines: baseline one is only using $y^d$ to recover $X^d$ and simply ignoring $y^1, \ldots, y^{d-1}$; baseline two is using $\{y^t\}_{t=1}^d$ with equal weights. Note that both of these can be viewed as special cases of one-bit LOWEMS with weights $(0, \ldots, 0, 1)$ and $(\frac{1}{r}, \frac{1}{r}, \ldots, \frac{1}{r})$ respectively.

Figure 1 shows that the recovery performance is poor when noise is either too large or too small, a similar phenomenon as observed in [3]. Figure 2 illustrates that one-bit LOWEMS reduces the recovery error compared to our baselines, which is also observed in the continuous observation setting [8]. Figure 3 shows that one-bit LOWEMS reduces the sample complexity required to guarantee successful recovery (defined as a relative error $\leq 0.4$).

Furthermore, we test the one-bit LOWEMS approach in the context of personalized learning using the ASSISTment dataset (for a precise description, see [7]). We truncate the dataset by eliminating students/questions with less than 100 responses. We keep a portion (10%) of the most recent data as the testing set, and use the remaining data to learn the matrix. To exploit the dynamic constraint, we divide the training set into $d$ bins chronologically. As we can see from Figure 4, exploiting the dynamic constraint yields better prediction performance on this dataset.
Fig. 1. Recovery error vs. observation noise ($\sigma_2 = 0.1$).

Fig. 2. Recovery error vs. perturbation noise ($\sigma_1 = 0.1$).

Fig. 3. Sample complexity vs. perturbation noise ($\sigma_1 = 0.1$).

Fig. 4. Experimental results on ASSISTment dataset

ACKNOWLEDGMENTS

This work was supported by grants NRL N00173-14-2-C001, AFOSR FA9550-14-1-0342, NSF CCF-1409406, CCF-1350616, and CMMI-1537261.

REFERENCES