Summary and discussion of: "Weighted Low Rank Matrix Approximation and Acceleration"

ZHANG Tianyu

1 Summary

This paper [1] proposed an element-wise weighted generalization of low-rank matrix completion (LRMC) problem, which is called weighted low-rank matrix approximation (WLRMA). An algorithm as well as two acceleration technique are proposed to solve the WLRMA problems. Further, a non-SVD version of the algorithm are proposed to handle extremely high-dimensional data. The algorithm as well as the acceleration technique are applied on a simulation dataset and a real dataset.

In this report, a brief overview of the proposed problem and the algorithms are given. Then the results of experiments on both simulation data and real data are presented. Finally a conclusion is given.

2 A Brief Overview of the Paper

2.1 From LRMC to WLRMA

A low-rank matrix completion problem (LRMC) is described as follows:

$$\text{minimize}_{X\in\mathbb{R}^{n\times p}} \|W * (M - X)\|_F^2 \quad \text{subject to } \text{rk}(X) \leq k. \tag{1}$$

where $W \in \mathbb{R}^{n\times p}$ is a binary matrix with

$$W_{ij} = \begin{cases} 0, \text{ if } (i,j) \notin \Omega \\ 1, \text{ if } (i,j) \in \Omega \end{cases} \tag{2}$$

where Ω is the index set of observed entries and * denotes the Hadamard product. The convex relaxation of LRMC is

$$\text{minimize}_{X\in\mathbb{R}^{n\times p}} \frac{1}{2}\|W * (M - X)\|_F^2 + \lambda\|X\|_* \tag{3}$$

where $\|X\|_*$ denotes the nuclear norm of X and λ is a tuning parameter.

The author generalized the LRMC problem by introducing a weight matrix $W \in \mathbb{R}^{n\times p}$ with non-negative entries. The generalized problem is called weighted low-rank matrix approximation (WLRMA) problem, which is described as follows:

$$\text{minimize}_{X\in\mathbb{R}^{n\times p}} \|\sqrt{W} * (M - X)\|_F^2 \quad \text{subject to } \text{rk}(X) \leq k. \tag{4}$$

Here $W \in [0,1]^{n \times p}$ is a matrix of weights (not necessary binary) and the objective is simply weighted residual sum of squares $\sum_{i=1}^{n} \sum_{j=1}^{p} W_{ij} \left(M_{ij} - X_{ij}\right)^2$. The convex relaxation of WLRMA is

$$\text{minimize}_{X \in \mathbb{R}^{n \times p}} \frac{1}{2} \|\sqrt{W} * (M - X)\|_F^2 + \lambda \|X\|_* \tag{5}$$

Next, the author proposed an algorithm to solve the WLRMA problem. Besides, two acceleration techniques are proposed to speed up the convergence of the algorithm.

2.2 The Algorithm and Acceleration Techniques

The problems (4) and (5) can be solved with proximal gradient descent. The baseline algorithm is described as follows:

Algorithm 1 Baseline Solver for (4) and (5)

1. Initialize $X^{(0)}$ and t.
2. Repeat until convergence:

$$X^{(i+1)} := \text{SVD}_k \left(tW * M + (1 - tW) * X^{(i)} \right) \text{ for equation (4)}$$

or

$$X^{(i+1)} := \text{S}_{t\lambda} \left(tW * M + (1 - tW) * X^{(i)} \right) \text{ for equation (5)}$$

Based on the baseline algorithm, the author proposed two acceleration techniques. The first one is Nesterov acceleration. The algorithm is described as follows:

Algorithm 2 Nesterov Accelerated Solver for (4) and (5)

1. Initialize $X^{(0)}$ and t.
2. Repeat until convergence:
   (a) Calculate

$$V^{(i)} = X^{(i)} + \frac{i-1}{i+2} \left(X^{(i)} - X^{(i-1)} \right)$$

   (b) Update $X^{(i+1)}$ by

$$X^{(i+1)} := \text{SVD}_k \left(tW * M + (1 - tW) * V^{(i)} \right) \text{ for equation (4)}$$

or

$$X^{(i+1)} := \text{S}_{t\lambda} \left(tW * M + (1 - tW) * V^{(i)} \right) \text{ for equation (5)}$$

The second is Anderson acceleration. The algorithm is described as follows:

Algorithm 3 Anderson Accelerated Solver for (4)

1. Initialize
   (a) Initialize $Y^{(0)}$ (e.g. column mean imputation) and $y^{(0)} = \text{vec}\left(Y^{(0)}\right)$.
   (b) Compute $X^{(0)} = \text{SVD}_k\left(Y^{(0)}\right)$.
2. Repeat until convergence:
   (a) Compute $f^{(i)} = \text{vec}\left(W * M + (1 - W) * X^{(i)}\right)$.
   (b) Set $r^{(i)} = f^{(i)} - y^{(i)}$
   (c) Concatenate $R^{(i)} = \left[R^{(i-1)} : r^{(i)}\right], \quad F^{(i)} = \left[F^{(i-1)} : f^{(i)}\right]$.
   (d) Solve OLS:
      i. solve ${R^{(i)}}^\top R^{(i)}\theta = 1$ w.r.t. θ,
      ii. normalize $\alpha = \frac{\theta}{\mathbf{1}^\top\theta}$.
   (e) Compute $y^{(i+1)} = F^{(i)}\alpha$ and set $Y^{(i+1)} = \text{matrix}\left(y^{(i+1)}\right)$.
   (f) Update $X^{(i+1)} = \text{SVD}_k\left(Y^{(i+1)}\right)$.
   (g) Drop the first column of both $F^{(i)}$ and $R^{(i)}$ if they contain more than m columns.

For (5), just replace SVD_k with $\text{S}_{t\lambda}$ in the algorithm above.

2.3 An Adaptation for High-Dimensional Data

The algorithms above are based on SVD, which is not suitable for high-dimensional data. To solve this problem, the author proposed a non-SVD version of the algorithm, borrowing idea from [2]. The WLRMA problem is restate as follows:

$$\text{minimize}_{A\in\mathbb{R}^{n\times k},B\in\mathbb{R}^{p\times k}} \left\|\sqrt{W} * \left(M - AB^T\right)\right\|_F^2 \tag{6}$$

The convex relaxation of the problem is

$$\text{minimize}_{A\in\mathbb{R}^{n\times k},B\in\mathbb{R}^{p\times k}} \frac{1}{2}\left\|\sqrt{W} * \left(M - AB^\top\right)\right\|_F^2 + \frac{\lambda}{2}\|A\|_F^2 + \frac{\lambda}{2}\|B\|_F^2 \tag{7}$$

The algorithm is described as follows:

Algorithm 4 WLRMA-ALS-sparse for (6)

1. Initialize
   (a) Set $A^{(0)} \in \mathbb{R}^{n\times k}$ and $B^{(0)} \in \mathbb{R}^{p\times k}$ at random.
2. Repeat until convergence:
   (a) Compute sparse $S^{(i)} = W * \left(M - A^{(i)}B^{(i)^T}\right)$.
   (b) Compute $H_A = A^{(i)}\left(A^{(i)^T}A^{(i)}\right)^{-1}$ and set $B^{(i+1)} = B^{(i)}A^{(i)^T}H_A + S^{(i)^T}H_A$.
   (c) Update sparse $S^{(i)} = W * \left(M - A^{(i)}B^{(i+1)^T}\right)$.
   (d) Compute $H_B = B^{(i+1)}\left(B^{(i+1)^T}B^{(i+1)}\right)^{-1}$ and set $A^{(i+1)} = A^{(i)}B^{(i+1)^T}H_B + S^{(i)}H_B$

For convex relaxation (7) simply use

$$H_A = A^{(i)}\left(A^{(i)^T}A^{(i)} + \lambda I\right)^{-1} \text{ and } H_B = B^{(i+1)}\left(B^{(i+1)^T}B^{(i+1)} + \lambda I\right)^{-1} \tag{8}$$

The Nesterov acceleration can also be applied to the algorithm above, which is given as follows:

Algorithm 5 Nesterov Accelerated WLRMA-ALS-sparse for (6) and (7)

1. Initialize
   (a) Set $A^{(0)} \in \mathbb{R}^{n\times k}$ and $B^{(0)} \in \mathbb{R}^{p\times k}$ at random.
2. Repeat until convergence:
   (a) Update $V_A^{(i)} = A^{(i)} + \frac{i-1}{i+2}\left(A^{(i)} - A^{(i-1)}\right)$ and $V_B^{(i)} = B^{(i)} + \frac{i-1}{i+2}\left(B^{(i)} - B^{(i-1)}\right)$.
   (b) Update $\left(A^{(i+1)}, B^{(i+1)}\right) := \Phi\left(V_A^{(i)}, V_B^{(i)}\right)$

where Φ is the transformation described by step 2 of the algorithm 4. Denote $Z = [A; B]$, then we have a fixed point iteration $Z^{(i+1)} = \Phi\left(Z^{(i)}\right)$. The Anderson accelerated version is given as follows:

Algorithm 6 Anderson Accelerated WLRMA-ALS-sparse for (6) and (7)

1. Initialize
   (a) Initialize $Z^{(0)}$ and $z^{(0)} = \text{vec}\left(Y^{(0)}\right)$.
2. Repeat until convergence:
   (a) Compute $f^{(i)} = \text{vec}\left(\Phi\left(Z^{(i)}\right)\right)$.
   (b) Set $r^{(i)} = f^{(i)} - y^{(i)}$
   (c) Concatenate $R^{(i)} = \left[R^{(i-1)} : r^{(i)}\right], \quad F^{(i)} = \left[F^{(i-1)} : f^{(i)}\right]$.
   (d) Solve OLS:
      i. solve ${R^{(i)}}^\top R^{(i)}\theta = 1$ w.r.t. θ,
      ii. normalize $\alpha = \frac{\theta}{\mathbf{1}^\top\theta}$.
   (e) Compute $z^{(i+1)} = F^{(i)}\alpha$ and set $Z^{(i+1)} = \text{matrix}\left(z^{(i+1)}\right)$.
   (f) Drop the first column of both $F^{(i)}$ and $R^{(i)}$ if they contain more than m columns.

2.4 Convergence Critera

The convergence criteria for the algorithms above are as follows:

$$\Delta^{(i)} = \left| \frac{\ell\left(X^{(i+1)}\right) - \ell\left(X^{(i)}\right)}{\ell\left(X^{(i)}\right)} \right| \tag{9}$$

where $\ell\left(X^{(i)}\right)$ is the corresponding objective function value at iteration i. The algorithm is terminated when $\Delta^{(i)}$ is smaller than a threshold ϵ.

3 Result and Discussion

In this section, a simulation study and a real data example are conducted to show the performance of the proposed algorithms. The algorithms are implemented with Python.

3.1 Simulation Study

In this simulation, algorithms 1, 2, 3 are applied to solve the WLRMA problem (4) and (5). The simulation data is generated as follows: first we draw $A \in \mathbb{R}^{n\times r}$ and $B \in \mathbb{R}^{p\times r}$ from standard normal distribution and compute $M = AB^T + E$. Here $E \in \mathbb{R}^{n\times p}$ is some matrix of errors drawn from $\mathcal{N}\left(0, \sigma^2\right)$. We further generate the matrix of weights $W \in \mathbb{R}^{n\times p}$ via uniform distribution. In our experiments we set $n = 1000, p = 100, r = 70$ and $\sigma = 1$. We initialize $X^{(0)}$ at zero matrix. The simulation results are as follows:

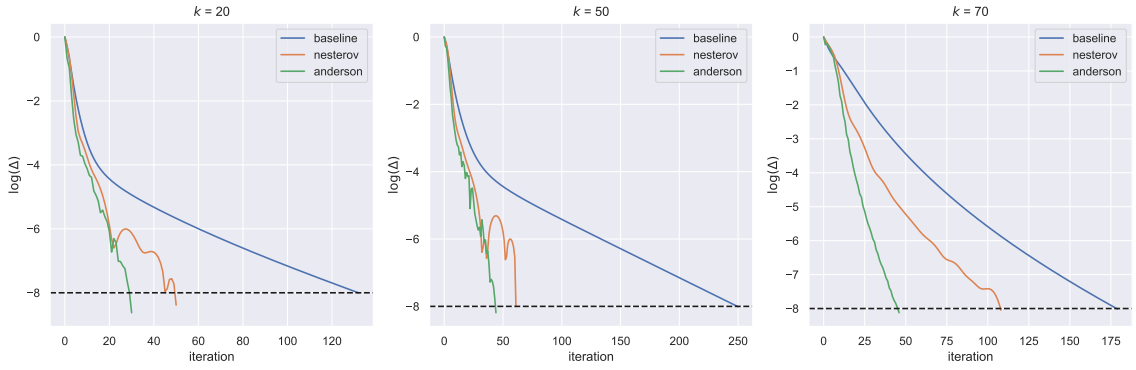

Figure 1: Solving the non-convex WLRMA problem with different ranks k.

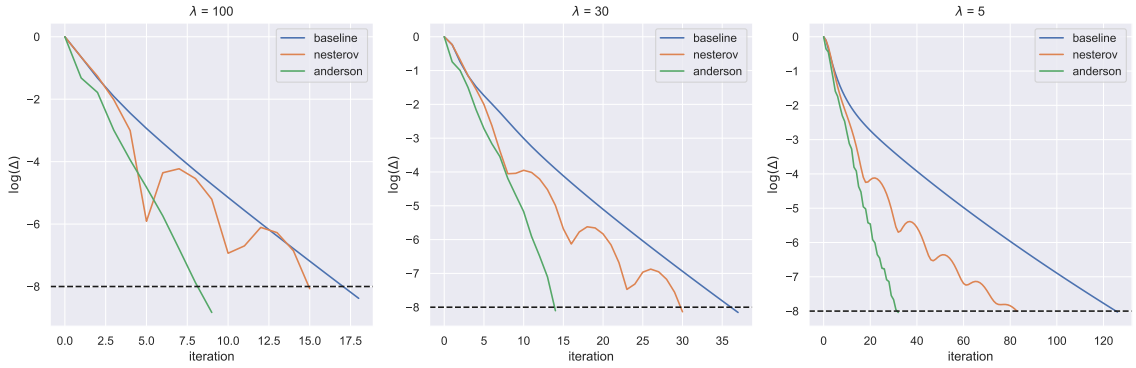

Figure 2: Solving the WLRMA convex relaxation problem with different penalty factors λ.

From the figures above we can see that the Nesterov accelerated algorithm converges faster than the baseline algorithm. The Anderson accelerated algorithm converges even faster than the Nesterov accelerated algorithm. The simulation results are consistent with those in the paper.

3.2 Real Data Example

In this example, algorithms 4, 5, 6 are applied to solve the WLRMA problem (6) and (7) to show the performance of algorithms on high-dimensional data. The dataset is MovieLens data, which describes a 5-star ratings. It contains 1 million ratings from 6000 users on 4000 movies that we store in a large 6000 $\times$ 4000 matrix M. Note that M is very sparse: some of the movies were rated by just a few users (if any) and only 5% of values are observed. For the initialization, a warm start is used. The simulation results are as follows:

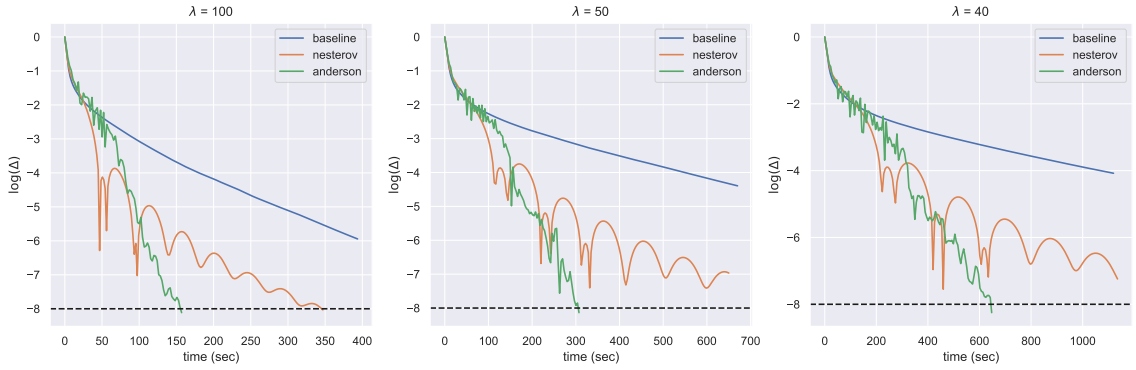

Figure 3: MovieLens Example: $\log(\Delta)$ v.s. iteration

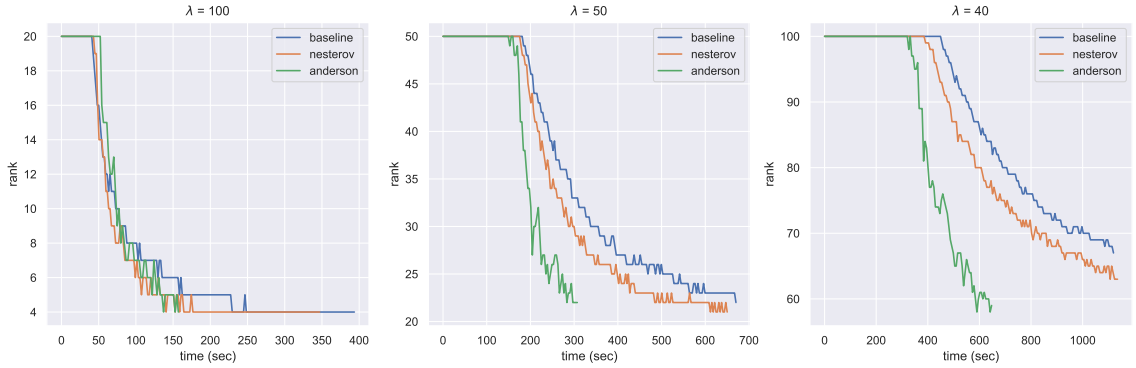

Figure 4: MovieLens Example: solution rank v.s. iteration

From the figures above we can see that the Nesterov accelerated algorithm converges faster than the baseline algorithm. The Anderson accelerated algorithm converges even faster than the Nesterov accelerated algorithm. However, for a looser tolerance, e.g. $\epsilon = 10^{-4}$, the Nesterov accelerated algorithm could converge faster than the Anderson accelerated algorithm. Overall the simulation results are consistent with those in the paper.

3.3 Discussion

From the simulation study and real data example we can see that the proposed algorithms are effective in solving the WLRMA problem. The acceleration techniques are also effective in speeding up the convergence of the algorithms. From the convergence process we can see that the Anderson accelerated algorithm are more stable than the Nesterov accelerated algorithm. However, the convergence of the algorithms are sensitive to the tolerance ϵ. For a looser tolerance, the Nesterov accelerated algorithm could converge faster than the Anderson accelerated algorithm.

Also note that although the author used these two examples to demonstrate the performance of the proposed algorithms, the reasons for generalizing from binary weights to

weights with real values have not been clearly explained. It seems that in missing value problem it's more reasonable to use binary weights, i.e. giving equal weights to observed entries and missing entries, respectively.

4 Conclusion

Here we summarize the main results of this report:

1. The author proposed a weighted low-rank matrix approximation (WLRMA) problem, which is a generalization of the low-rank matrix completion (LRMC) problem.

2. The author proposed an algorithm to solve the WLRMA problem. Two acceleration techniques are proposed to speed up the convergence of the algorithm. The author also proposed a non-SVD version of the algorithm to handle high-dimensional data.

3. A simulation study and a real data example are conducted to show the performance of the proposed algorithms in Python. The results are consistent with those in the paper.

4. Although the author used these two examples to demonstrate the performance of the proposed algorithms, the reasons for generalizing from binary weights to weights with real values have not been clearly explained.

5 Software

The code for the simulation study and real data example are available at `https://github.com/jonas-tzhang/MATH5472-Project`.

References

[1] Elena Tuzhilina and Trevor Hastie. Weighted low rank matrix approximation and acceleration. *arXiv preprint arXiv:2109.11057*, 2021.

[2] Trevor Hastie, Rahul Mazumder, Jason D Lee, and Reza Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *The Journal of Machine Learning Research*, 16(1):3367–3402, 2015.