Matrix Completion Project

Yunjuan Wang, Alex Bai, Nilay Thakor

April 2020

1 Introduction

Matrix completion and recovery is a common problem for real world settings such as recommendation system and sensor localization. One famous example is the Netflix challenge, which is a contest to develop algorithm to improve the prediction of movie rating. The incomplete rating matrix is a reflection of users' tastes on movies and the goal is to predict unseen user ratings. Another example is Internet of Things, where each of the sensor node need to communicate to adjacent node and send to data center. The goal is to recover the pairwise distance matrix if the absolute locations of a few sensor nodes are provided. Other settings like reconstruct 3D scene geometry and camera poses from multiple images, recovery signal from electronic control system, image compression and restoration, massive multiple input multiple output are also relevant.

In all of these settings, the matrices we are considering can be extremely large. So if we add no constraints on the number of freedom of the matrix, there's no hope to recovery the huge matrix with small error because the missing entries can be assigned to any number. But when the matrix is low rank, we are able to exploit the structure and perform the recovery. In fact, low rank matrix model is also a classical technique in machine learning such as principle component analysis and multitask learning.

The problem we are considering is formulated as below. We want to recover matrix M. The goal is to find the lowest rank matrix X which matches the observed entries Ω from M.

$$\begin{aligned} &\min_X \ \text{rank}(X) \\ &s.t. X_{i,j} = M_{i,j} \forall (i,j) \in \Omega \end{aligned}$$

Define $P_\Omega(X) = \begin{cases} X_{i,j}(i,j) \in \Omega \\ 0(i,j) \notin \Omega \end{cases}$, which is the projection operator on the observed index set Ω. When the matrix is corrupted by noise, we need to keep the noise level within an appropriate range.

$$\begin{aligned} &\min_X \ \text{rank}(X) \\ &s.t. P_\Omega(X) - P_\Omega(M) \leq \delta \end{aligned}$$

where $\delta > 0$ is the tolerance error budget. Unfortunately, the rank minimization problem is NP-hard and cost expensively to solve it completely. Thus, the state of the art algorithms are attempting to solve an approximate problem. Candès and Recht [2009] solved the problem using nuclear norm as a way of convex relaxation. Nuclear norm minimization problem can be recast as a semidefinite programming problem, which has several efficient algorithms to solve it such as SDPT3 Toh et al. [1999] and SeDeMi Sturm [1999]. Another way of solving this is the singular value thresholding Cai et al. [2010], and it has some extensions like fast singular value thresholding (FSVT) Cai and Osher [2013], fixed point continuation with approximate SVD (FPCA)Ma et al. [2011], accelerated proximal gradient with line-search-like acceleration (APGL) Toh and Yun [2010].

In fact, matrix completion problem has close relationship with matrix factorization problem. Take Netflix problem as an example. Consider X_{ij} represents the rating assigned to a movie i by a user j. We make an assumption that some movies are more popular in general than others, and some users are more generous. Thus, we can say that $X_{ij} = a[i] \cdot b[j]$, where features a and b capture the respective contributions of the movie and the user to the ranking. If $a[i]$ is large, movie i receives good ratings in general. If $b[j]$ is large, user j give good ratings in general. In this case, we only consider one factor, so X can be represented as $X = ab$, where $a \in \mathbb{R}^m, b \in \mathbb{R}^n$. Now we generalize the problem and consider r factors that capture the dependence between the rating s and the movie. Then the problem can be transformed as bilinear factorization.

$$X \approx AB, A \in \mathbb{R}^{m \times r}, B \in \mathbb{R}^{r \times n}$$

We can further prove that solving

$$\begin{aligned} &\min_X \|X\|_* \\ &s.t. X = LR^\top \\ &\quad P_\Omega(X) = P_\Omega(M) \end{aligned}$$

is equivalent to solve

$$\begin{aligned} &\min_{L,R,X} \|P_\Omega(M) - P_\Omega(X)\|_F^2 + \frac{\lambda}{2}(\|L\|_F^2 + \|R\|_F^2) \\ &s.t. X = LR^\top \end{aligned}$$

where λ controls the effect of nuclear norm regularization. Higer λ leads to better low rank solutions.

2 Theory

Assumption 1 (Low rank assumption)**.** For $n_1 \times n_2$ matrix M of rank r, assume that $\min\{n_1, n_2\} \gg r$.

Theorem 2.1. Vershynin [2018] Consider fixed $n \times n$ matrix X with $rank(X) = r$, where $r \ll n$. Each entry X_{ij} is revealed to us independently with probability $p \in (0, 1)$. We only observe matrix Y, $Y_{ij} = \delta_{ij} X_{ij}$, $\delta_{i,j} \sim Ber(p)$. Choose $p = \frac{m}{n^2}$. Let $\hat{X}$ be a best rank r approximation to $p^{-1}Y$. Then

$$\mathbb{E}\frac{1}{n}\|\hat{X} - X\|_F \leq C\sqrt{\frac{rn\log n}{m}}\|X\|_\infty$$

as long as $m \geq n \log n$.

Theorem 2.2 (Eckart-Young)**.** Let $A = U\Sigma V^T$ be a singular-value decomposition of A where $\text{rank}(A) = r$ and Σ is diagonal with entries $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$, then under $\|\cdot\|_X = \|\cdot\|_F$, $A_k = U_k \Sigma_k V_k^T$ is the minimizer to low rank approximation problem where U_k and V_k are the first k columns of U and V and $\Sigma_k = \text{diag}(\sigma_1, \ldots, \sigma_k)$.

Definition (Restricted Isometry Property)**.** Let $\mathcal{A}$: $\mathbb{R}^{m \times n} \to \mathbb{R}^p$ be a linear map. W.l.o.g, assume $m \leq n$. For every integer r with $1 \leq r \leq m$, define the r-restricted isometry constant to be the smallest number $\delta_r(\mathcal{A})$ such that

$$(1 - \delta_r(\mathcal{A}))\|X\|_F \leq \|\mathcal{A}(X)\| \leq (1 + \delta_r(\mathcal{A}))\|X\|_F$$

holds for all matrices X of rank at most r.

Definition (coherence)**.** Let U be a subspace of $\mathbb{R}^n$ of dimension r and P_U be the orthogonal projection onto U. Then the coherence of U is defined as

$$\mu(U) = \frac{n}{r}\max_{1 \leq i \leq n}\|P_U e_i\|^2$$

This can be thought of as a mesure of "how well can the subspace represent standard basis e_i. High coherence means the value are correlated, localized in matrix, while low coherence means the value are spreading out though the matrix. We are interested in the subspace with low coherence and saying that if a matrix has row and column spaces that are incoherent with the standard basis, the nuclear norm minimization can recover this matrix from a random sampling of a small number of entries.

Assumption 2. The coherences obey $\max(\mu(U), \mu(V)) \leq \mu_0$ for some positive μ_0.

Assumption 3. The SVD of $n_1 \times n_2$ matrix M is given by $M = \sum_{1 \leq k \leq r} \sigma_k u_k v_k^*$, then the $n_1 \times n_2$ matrix $\sum_{1 \leq k \leq r} u_k v_k^*$ has a maximum entry bounded by $\mu_1 \sqrt{r/(n_1 n_2)}$ in absolute value for some positive μ_1.

The idea of these two assumptions are to define low coherence matrix mathematically.

Theorem 2.3 (Candès and Recht [2009])**.** Let M be a $n_1 \times n_2$ matrix of rank r satisfying assumption 2 and 3, choose $n = \max(n_1, n_2)$. Suppose we observe m entries of M with locations sampled uniformly at random. Then there exist constants C, c, such that if

$$m \geq C \max(\mu_1^2, \mu_0^{0.5}\mu_1, \mu_0 n^{0.25}) nr(\beta \log n)$$

for some $\beta > 2$, then the optimal solution to the norm minimization problem is unique and equal to M with probability at least $1 - cn^{-\beta}$. Furthrermore, for low rank case $r \leq \mu_0^{-1} n^{0.2}$, with same probability provided that

$$m \geq C\mu_0 n^{1.2} r(\beta \log n).$$

3 Algorithm

3.1 Convex relaxation

Since $rank(X)$ is nonconvex, we use nuclear norm $\|X\|_*$ as a convex surrogate for $rank(X)$. Thus, the problem is transferred as

$$\begin{aligned} &\min_X \|X\|_* \\ &s.t. P_\Omega(X) = P_\Omega(M) \end{aligned} \tag{3.1}$$

Since spectral norm is dual norm of nuclear norm, which can be represented as $\max\{\langle X, Y\rangle : \|Y\| \leq 1\}$. Using Shcur complement, $\|Y\| \leq 1$ is equivalent to $\begin{bmatrix} I & Y \\ Y^\top & I \end{bmatrix} \succeq 0$. Thus, our problem can be rewritten as

$$\begin{aligned} &\min_X \max_Y \langle X, Y\rangle \\ &s.t. \begin{bmatrix} I & Y \\ Y^\top & I \end{bmatrix} \succeq 0 \\ &\quad P_\Omega(X) = P_\Omega(M) \end{aligned} \tag{3.2}$$

We still consider the dual problem of this problem. Let SVD of X be $X = U\Sigma V^\top$. Define $W_1 = U\Sigma U^\top$, $W_2 = V\Sigma V^\top$. Thus, our problem can be again transferred as

$$\begin{aligned} &\min_{W_1, W_2} \frac{1}{2}(Tr(W_1) + Tr(W_2)) \\ &s.t. \begin{bmatrix} W_1 & X \\ X^\top & W_2 \end{bmatrix} \succeq 0 \\ &\quad P_\Omega(X) = P_\Omega(M) \end{aligned} \tag{3.3}$$

This is a standard Semi-Definite Programming problem, and can be solved by classical SDP algorithm.

3.2 Singular value thresholding

We reformulate the problem. Given a matrix $M \in \mathbb{R}^{m\times n}$, we only observe the entries $M_{i,j}, (i,j) \in \Omega$. In order to recover matrix M, we need to solve

$$\min_{X\in\mathbb{R}^{m\times n}} F(X) = \frac{1}{2}\|P_\Omega(M) - P_\Omega(X)\|_F^2 + \lambda\|X\|_*$$

where $\|X\|_*$ is nuclear norm of X, i.e. $\|X\|_* = \sum_{i=1}^r \sigma_i(X)$. $r = rank(X), \sigma_1(X) \geq \sigma_2(X) \geq \cdots \geq \sigma_r(X) \geq 0$. After reformulation, this become a convex problem. The first term is quadratic, the second is a norm, which is convex.

Using projected gradient descent method to think of this problem. For the first term, the gradient is $-(P_\Omega(M) - P_\Omega(X))$. For the second term, we consider the proximal operator

$$prox_t(X) = \arg\min_{Z\in\mathbb{R}^{m\times n}} \frac{1}{2t}\|X - Z\|_F^2 + \lambda\|X\|_*$$

.

Lemma 3.1. $prox_t(X) = S_{\lambda t}(X)$, matrix soft-thresholding at the level λ. Here $S_\lambda(X)$ is defined by $S_\lambda(X) = U\Sigma_\lambda V^\top$, where $X = U\Sigma V^\top$ is an SVD, and Σ_λ is diagonal with $(\Sigma_\lambda)_{ii} = \max\{\Sigma_{ii} - \lambda, 0\}$.

Hence the proximal gradient update step is:

$$X^{k+1} = prox_t(X^k + \delta_t(P_\Omega(M) - P_\Omega(X^k))) = S_{\lambda t}(X^k + \delta_t(P_\Omega(M) - P_\Omega(X^k))) \tag{3.4}$$

where δ_t is step size. If we choose fixed step size $\delta_t = 1$, update step is:

$$X^{k+1} = S_\lambda(P_\Omega(M) + P_\Omega^\perp(X^k))$$

where $P_\Omega^\perp(X)$ projects onto unobserved set, $P_\Omega(X) + P_\Omega^\perp(X) = X$. See the formal algorithm 1 for details.

Algorithm 1 SVT Algorithm

Input: observed entries $P_\Omega(M)$,
a sequence of positive step sizes $\{\delta_k\}_{k\geq 0}$,
a regularization parameter λ,
termination criteria
Initialize: $Y_0 = 0_{n_1\times n_2}$
while *termination criteria is not reached* **do**
  $[U_k, \Sigma_k, V_k] = \text{svd}(Y_{k-1})$
  $X_{k+1} = U_k \text{diag}(\{\sigma_i(\Sigma_k) - \lambda)_+\}_i) V_k^\top$
  $Y_{k+1} = Y_k + \delta_k(P_\Omega(M) - P_\Omega(X_{k+1}))$
  $k = k + 1$
end
Output: X_k

The converge analysis can be found in theorem 3.2.

Theorem 3.2 (Cai et al. [2010])**.** Suppose that the sequence of step sizes obeys $0 < \inf \delta_t \leq \sup \delta_t \leq 2$, then the sequence $\{X^k\}$ obtained via (3.4) converges to the unique solution of

$$\min_X \lambda\|X\|_* + \frac{1}{2}\|X\|_F^2$$
$$s.t. P_\Omega(X) = P_\Omega(M)$$

Based on this thought, various SVT-based techniques have been proposed such as iterative hard thresholding (IHT) Tanner and Wei [2013], Jain et al. [2010]. Consider hard thresholding operator H_r as $H_r(X) = U\Sigma_r V^\top$ where $\Sigma_r(i,i) = \begin{cases} \Sigma(i,i) & i \le r \\ 0 & i > r \end{cases}$. Consider sensing operator $\mathcal{A}$, $b_l = \mathcal{A}(X)_l = \text{trace}(A_l^* X)$ for $l = 1, 2, \cdots, p$. $\mathcal{A}^*(\cdot)$ is defined as $A^*(y) = \sum_{l=1}^p y(l)\mathcal{A}_l$. See algorithm 2 for details.

Algorithm 2 Normalized IHT Algorithm

Input: Sensing operator $\mathcal{A}$, $b = \mathcal{A}(M)$, r, termination criteria
Initialize: $X_0 = H_r(\mathcal{A}^*(b))$, $j = 0$, U_0 as the top r left singular vecotr of X_0
while *termination criteria is not reached* **do**
  Set the projection operator $P_U^k = U_k U_k^\top$
  Compute the step size $\delta_k = \frac{\|P_U^k \mathcal{A}^*(b - \mathcal{A}(X_k))\|_F^2}{\|\mathcal{A}(P_U^k \mathcal{A}^*(b - \mathcal{A}(X_k)))\|_F^2}$
  Set $X_{k+1} = H_r(X_k + \mu_k \mathcal{A}^*(b - \mathcal{A}(X_k)))$
  Let U_{k+1} be the top r left singular vectors of X_{k+1}
  $k = k + 1$
end
Output: X_k

One drawback of SVT algorithm is that we need to compute SVD in each iteration, which takes $O(mn^2)$ time and can be computationally expensive. To handle this problem, a fast SVT approach is proposed to compute $S_\lambda(X)$ without using SVD at each iteration.

3.3 Alternating Least Squares Minimization

We assume that M is of low rank $k \ll \min\{m, n\}$ and represent our estimation matrix $X \in \mathbb{R}^{m \times n}$ in a bi-linear form $X = UV^\dagger$ where $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$. The low rank assumption makes it so that the matrices U, V are much smaller than X and thus computationally easier to optimize. The problem we seek to optimize is then:

$$\min_{U \in \mathbb{R}^{m \times k}, V \in \mathbb{R}^{n \times k}} ||P_\Omega(UV^\dagger) - P_\Omega(M)||_F^2 \tag{3.5}$$

However, the bilinear form of X makes (3.1) a non-convex problem. Alternating minimization seeks to approximately solve (3.1) by alternatively optimizing for U and V. In Jain et al. [2013], they show that once we assume that the matrix M satisfies a incoherence property, we can approximately solve the problem efficiently.

Definition (μ-incoherence)**.** A matrix $M \in \mathbb{R}^{m \times n}$ is *incoherent with parameter* μ if:

$$||u^{(i)}||_2 \le \frac{\mu\sqrt{k}}{\sqrt{m}} \; \forall i \in [m], \quad ||v^{(j)}||_2 \le \frac{\mu\sqrt{k}}{\sqrt{n}} \; \forall j \in [n] \tag{3.6}$$

where $M = U\Sigma V^\dagger$ is the SVD of M and $u^{(i)}, v^{(j)}$ denote the i^{th} row of U and j^{th} row of V respectively.

In Jain et al. [2012], they propose the following variant of alternating minimization:

Algorithm 3 AltMinComplete

Input: observed set Ω, values $P_\Omega(M)$
Partition Ω into $2T+1$ subsets $\Omega_0, \ldots, \Omega_{2T}$ with each element of Ω belonging to one of the Ω_t with equal probability (sampling with replacement)
Initialize: $\hat{U}^0 = SVD(\frac{1}{p}P_{\Omega_0}(M), k)$ i.e., top-k left singular vectors of $\frac{1}{p}P_{\Omega_0}(M)$
Clipping step: Set all elements of $\hat{U}^0$ that have magnitude greater than $\frac{2\mu\sqrt{k}}{\sqrt{n}}$ to zero and orthonormalize the columns of $\hat{U}^0$
for *t = 0,. . . ,T-1* **do**
  $\hat{V}^{t+1} \longleftarrow \text{argmin}_{V \in \mathbb{R}^{n\times k}} ||P_{\Omega_{t+1}}(\hat{U}^t V^\dagger - M)||_F^2$
  $\hat{U}^{t+1} \longleftarrow \text{argmin}_{U \in \mathbb{R}^{m\times k}} ||P_{\Omega_{T+t+1}}(U(\hat{V}^{t+1})^\dagger - M)||_F^2$
end
Return $X = \hat{U}^T(\hat{V}^T)^\dagger$

They then presented the following result:

Theorem 3.3. Let $M = U^*\Sigma^*V^{*\dagger} \in \mathbb{R}^{m\times n}$ $(n \geq m)$ be a rank-k incoherent matrix, i.e., both U^* and V^* are μ-incoherent. Also, let each entry of M be observed uniformly and independently with probability,

$$p > C\frac{(\frac{\sigma_1^*}{\sigma_k^*})^2\mu^2k^{2.5}\log n \log\frac{k||M||_F}{\epsilon}}{m\delta_{2k}^2},$$

where $\delta_{2k} \leq \frac{\sigma_k^*}{12k\sigma_1^*}$ and $C > 0$ is a global constant. Then with high probability for $T = C'\log\frac{||M||_F}{\epsilon}$, the outputs $\hat{U}^T$ and V^T of AltMinComplete, with input $(\Omega, P_\Omega(M))$ satisfy: $||M - \hat{U}^T(V^T)^\dagger||_F \leq \epsilon$

The theorem implies that alternating minimization may require a larger sample complexity Ω than convex optimization techniques. Explicitly, alternating minimization requires $|\Omega| = O((\frac{\sigma_1^*}{\sigma_k^*})^4\mu^2k^{4.5}n\log n\log\frac{k||M||_F}{\epsilon})$ and can recover the matrix M in $O(\log\frac{1}{\epsilon})$ steps.

4 Experiment

We did experiments with simulated and real world data to verify some of the theorems and compare the relative performance of algorithms. The synthetic data generation is done via following method:
To sample k-rank $M \in \mathbb{R}^{m\times n}$

$$\begin{aligned} L \in \mathbb{R}^{m\times k}, L_{ij} &\sim \mathcal{N}(0,1) \\ R \in \mathbb{R}^{n\times k}, R_{ij} &\sim \mathcal{N}(0,1) \\ M &= LR^T \\ P_\Omega(M) = \Delta \otimes M, \quad \Delta_{i,j} &= 1 \text{ w.p. } p \end{aligned}$$

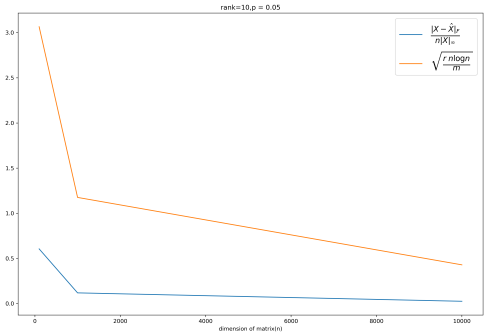
For the rest of the discussion we define the errors as following:

- Reconstruction Error $= \|P_\Omega(M - \hat{M})\|_F$
- Testing error $= \sqrt{\frac{\sum_{i,j\in(\text{Test Data})}(M_{i,j}-\hat{M}_{i,j})^2}{\#i,j\in(\text{Test Data})}}$

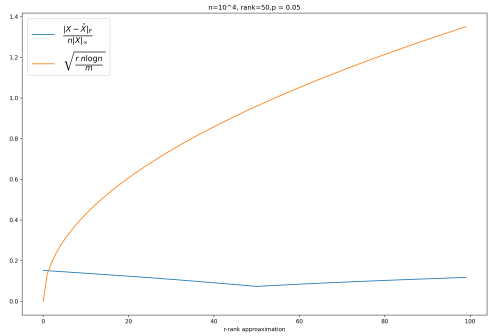
4.1 Experiments on synthetic data

k-rank approximation using svd is defined as $X_k = \sum_{i=1}^{k} s_i u_i v_i^T$ From Vershynin [2018] we get the following bound on k-rank approximation of a matrix.

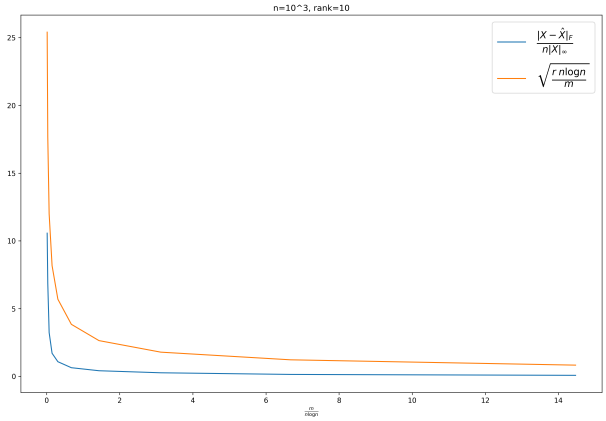
$$\mathbb{E}\frac{1}{n}\|\hat{X} - X\|_F \leq C\sqrt{\frac{rn\log n}{m}}\|X\|_\infty$$



(a)



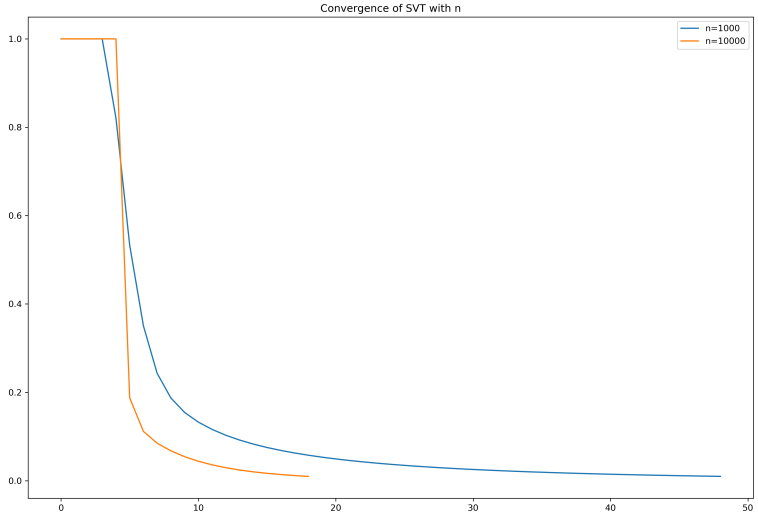
(b)



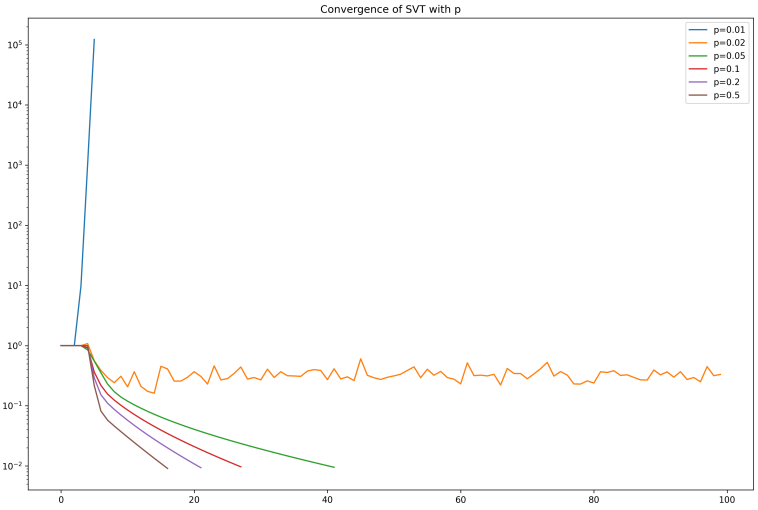
(c)

Figure 1: (a) Dimension of matrix vs reconstruction error (b) Approximation rank vs reconstruction error Dimension(c) Sparsity vs Reconstruction error

In Figure 1 we track the theoretical bound for k-rank approximation for various scenarios. In (a) we tracked reconstruction error for increasing matrix size and as seen from the graph its trajectory resembles the upper bound. In (b) reconstruction error computed for increasing rank. The rank of original matrix is 50. So error is decreasing till then and increasing again as expected. In (c) we compare the error for different p values.



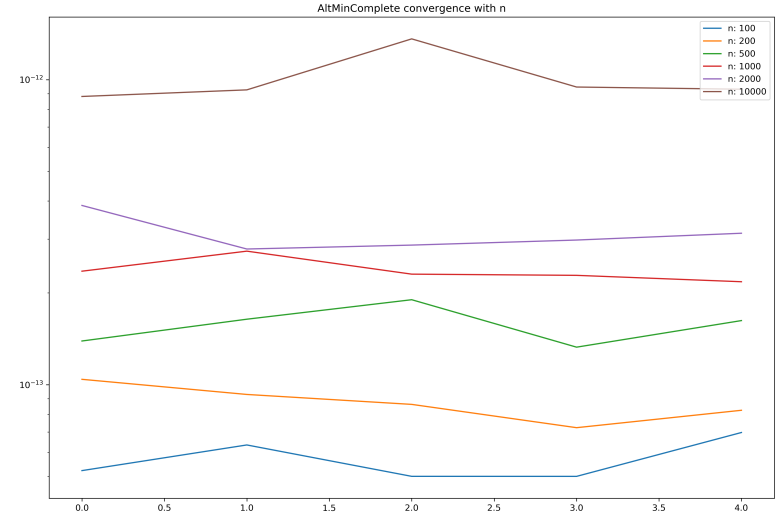
(a)



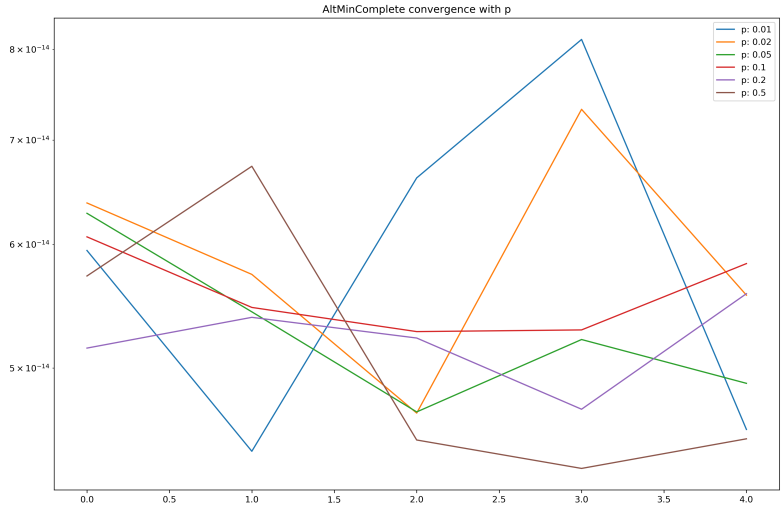
(b)

Figure 2: (a) Convergence of SVT for different matrix sizes (b) reconstruction error vs p

In Figure 2 we did the convergence analysis for SVT algorithm. As seen in (a) it converges quickly for higher values of n. Similarly it converges very quickly for higher value of p.



(a)



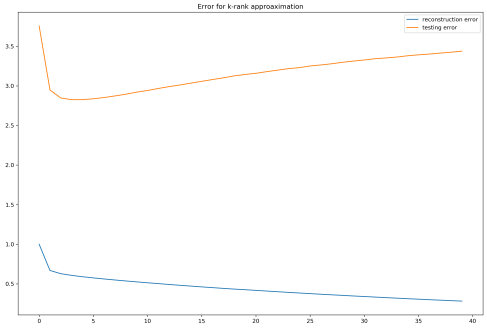
(b)

Figure 3: (a) Convergence of AltMinComplete for different matrix sizes (b) reconstruction error vs p

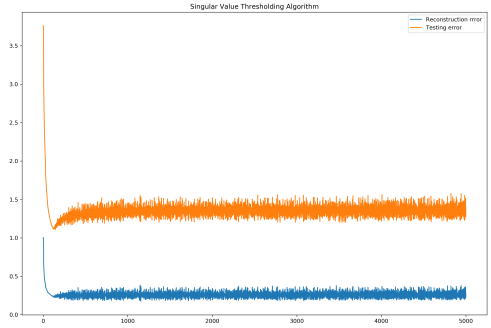
As seen in Figure 3 Algorithm 3.3 converges in first iteration itself. Observing the y-axis in figure 3.3 we can say that convergence of AltMinComplete is not affected significantly by size of the matrix.

4.2 Experiments on real world data

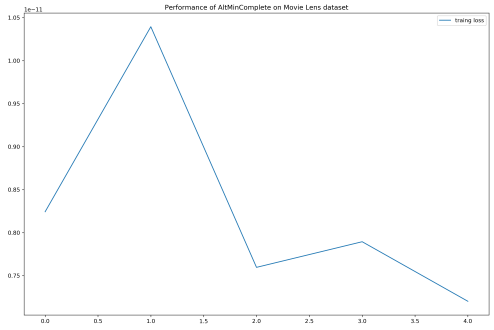
For analysis on real world data we used Movie Lens-100k dataset. It is essentially a movie rating database where each users has review few movies and each movie has few reviewers. The dataset has 943 unique viewers and 1682 movies. So we have 943×1682 dimension matrix with each entry of the matrix being the ratings. We present the convergence analysis for k-rank approximation, SVT and AltMinComplete.



(a)



(b)



(c)

Figure 4: (a) k-rank approximation to get the rank (b) convergence in SVT (c) convergence in AltMinComplete

As seen from the figure 4 For k-rank approximation we see a over fitting curve. This exercise is useful in estimating the rank of the matrix. in (b) SVT reduces the error initially very fast but does not change much after that. On the other hand AltMinComplete error goes to almost close to zero in first iteration itself.

5 Conclusion

In this report, we first discuss the background and previous literature of matrix completion problem. We give some preliminary theory, saying that under the observation of certain entries, the matrix can be recovered under low rank and low coherence properties. Then we convexify the problem and transfer to different formula based on the algorithm. We mainly show two kind of algorithms: singular value thresholding and alternating least squares minimization. Finally we implement these two algorithms and test them using synthetic data and real world data.

Our implementation of the algorithms at be found on https://github.com/nthakor/matrix_completion

6 Contribution

Yunjuan Wang: Section 1: Introduction; Section 2: Theory; Section 3 Algorithm: 3.1 Convex relaxation, 3.2 Singular value thresholding; Section 5 Conclusion
Alex Bai: 3.3 Alternating Least Squares Minimization
Nilay Thakor: Experiments

References

Jian-Feng Cai and Stanley Osher. Fast singular value thresholding without singular value decomposition. *Methods and Applications of Analysis*, 20(4):335–352, 2013.

Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on optimization*, 20(4):1956–1982, 2010.

Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717, 2009.

Prateek Jain, Raghu Meka, and Inderjit S Dhillon. Guaranteed rank minimization via singular value projection. In *Advances in Neural Information Processing Systems*, pages 937–945, 2010.

Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674, 2013.

Shiqian Ma, Donald Goldfarb, and Lifeng Chen. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128(1-2):321–353, 2011.

Jos F Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization methods and software*, 11(1-4):625–653, 1999.

Jared Tanner and Ke Wei. Normalized iterative hard thresholding for matrix completion. *SIAM Journal on Scientific Computing*, 35(5):S104–S125, 2013.

Kim-Chuan Toh and Sangwoon Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of optimization*, 6(615-640):15, 2010.

Kim-Chuan Toh, Michael J Todd, and Reha H Tütüncü. Sdpt3—a matlab software package for semidefinite programming, version 1.3. *Optimization methods and software*, 11(1-4):545–581, 1999.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.