Nuclear Norm and Alternating Minimization for Matrix Completion

Soo Min Kwon*

Abstract

In many practical problems, such as recommendation systems, one would like to recover a data matrix given a sampling of its entries. This problem, widely known as the matrix completion problem, has gained massive popularity with its overlap in literature with compressed sensing, as well as prized awards such as the Netflix problem. In this paper, we investigate matrix completion in detail, primarily showing that these "unfinished" matrices can easily be computed through convex optimization. We study the types of matrices that can be completed and explore the different methods of convex optimization schemes that can solve for these matrices. We also observe the matrix completion optimization problem in the dual domain, and conduct a residual analysis from the resulting matrices. We show results of matrix completion with different optimization formulas on both synthetic and real data.

1 Introduction

Many modern applications involve recovering partially observed matrices, examples including collaborative filtering [12, 8], compressed sensing [3], seismic data interpolation [1], and computer vision [9]. The problem of matrix completion is illustrated in Figure 1. To make sense of this illustration, consider a data matrix where row index users and columns index restaurants. Let the colored entries in the matrix be numerical ratings of users given to a restaurant, and let the white entries be empty entries in which users have not rated certain restaurants. The objective is to predict what ratings these users would have given to other restaurants, given their ratings to some restaurants. How can we make this type of inference? There are many ways to complete this problem, one significant method, which we investigate, is solving for low-rank matrices that essentially "build" the whole matrix.

Related Works. Matrix completion is a widely studied problem, specifically studied due to the Netflix prize problem [5]. The Netflix prize problem is similar to the scenario proposed previously, in which one (here Netflix) would like to complete

*Department of Electrical and Computer Engineering, Rutgers, The State University of New Jersey, New Brunswick, NJ 08854, smk330@scarletmail.rutgers.edu.

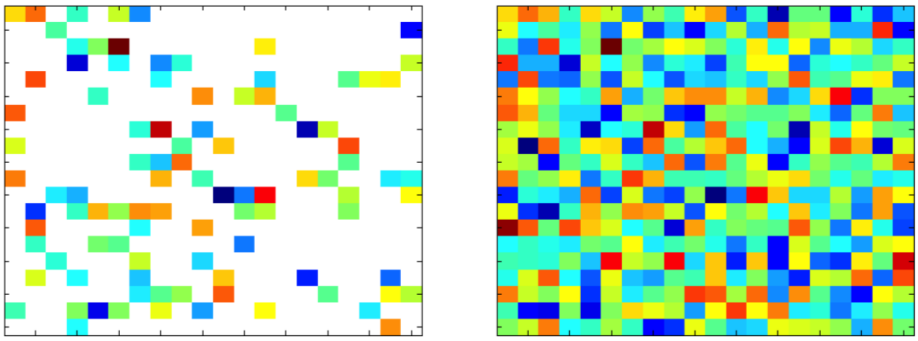

Figure 1: Left – example of a matrix with missing entries, right – matrix recovered through convex optimization

a matrix to recommend movies and shows to a user. In this case, the matrix to be completed can be interpreted as an approximately low rank matrix, as the recommendations only really depend on a certain subset of reviews. In literature, many researchers propose solutions to different methods that can solve these types of problems, such as alternating minimization and convex relaxations [7, 4]. Alternating minimization can be a better solution to these problems, as convex relaxations does not scale well. Iin this paper, we study the matrix completion problem using both the nuclear norm and the alternating minimization algorithm to solve for these matrices.

Our Contributions. This paper is merely to show results from solving matrices using a convex relaxation and alternating minimization of the original matrix completion problem. For detail, please refer to the paper by Candes et al. for derivations of properties for exact matrix completion with convex optimization [2]. The rest of this paper is organized as follows. We introduce matrix completion as an optimization problem and its convex form. We then derive and analyze its dual form. We then perform experiments on both synthetic and real data with both the nuclear norm form as well as the alternating minimization form. Through this paper, the questions we aim to answer are the following:

- How can we complete a data matrix by using only a certain sampling of its entries?
- What type of matrices can we complete using convex optimization?
- What does the dual form of the matrix completion optimization formula look like?
- How can we analyze the dual form of the optimization problem in context of a penalty function?

2 Matrix Completion: Properties & Formulation

We dedicate this section to discuss the properties that admit matrix completion as well as the optimization formulation, as shown in Candes et al. [2]. Consider a matrix M of dimensions $M \in \mathbb{R}^{n \times n}$ with rank R, where R takes a positive integer value. This matrix has a total of n^2 entries, and turns out it also has a total of (2n - R)R degrees of freedom. This value can be calculated by looking at the number of left and right singular vectors of M.

Can we recover all matrices of dimensions $M \in \mathbb{R}^{n \times n}$ that have this many degrees of freedom? It turns out we can recover most matrices, however, we need to have at least one value is each row/column. For example, consider a matrix M that is generated through the outer product between two canonical basis vectors, e_1 and e_n:

$$M = \begin{bmatrix} 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix}.$$

Let the 0's in M be the values not yet observed and 1 be the value that is observed. This matrix cannot be completed, as there are rows and columns not observed (at all) in the matrix. Logically, this makes sense – going back to the initial example, if there was a user who did not rate any restaurants at all, it would be impossible to predict a rating to a restaurant for that user. Thus, it is not possible to recover *all* low rank matrices (M has rank one), but most of them can be if they hold the above properties.

If we have enough observations in the matrix, we can hope to recover this low rank matrix with the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \text{rank}(X) \\ \text{subject to} \quad & X_{ij} = M_{ij} \qquad (i,j) \in \Omega, \end{aligned} \tag{1}$$

where X is the matrix to be estimated and Ω is the set of entries we have in the matrix. However, in theory, this optimization problem would work if there was a unique low rank solution to fill the matrices. However, not only is this problem is actually NP-hard, but it is non-convex! Instead, if a matrix has rank R, then it has R nonzero singular values so that we can observe the non-vanishing singular values. Mathematically, we can relax the constraints by looking at the *nuclear norm*:

$$||X||_* = \sum_{i=1}^{n} \sigma_i X. \tag{2}$$

Now with the nuclear norm from (2) and the original matrix completion problem

in (1), we can formulate the following relaxed convex optimization problem:

$$\begin{aligned} \text{minimize} \quad & ||X||_* \\ \text{subject to} \quad & X_{ij} = M_{ij} \qquad (i,j) \in \Omega. \end{aligned} \tag{3}$$

2.1 Equivalent Forms of Matrix Completion

Nuclear norm minimization is a widely studied heuristic for matrix completion. However, if the data matrix that we're looking at is symmetric and positive semi-definite (which may be the case for real-world problems), then the nuclear norm of data matrix M would be the sum of the eigenvalues, or the trace of X [2]. Thus for a positive semi-definite matrix, we can formulate an equivalent optimization problem:

$$\begin{aligned} \text{minimize} \quad & \text{trace}(X) \\ \text{subject to} \quad & X_{ij} = M_{ij} \quad (i,j) \in \Omega. \end{aligned} \tag{4}$$

Even if the data matrix M was not positive semi-definite or symmetric, we can formulate the problem to be an equivalent form of (4) by solving for the following semi-definite program (SDP):

$$\begin{aligned} \text{minimize} \quad & \text{trace}(W_1) + \text{trace}(W_2) \\ \text{subject to} \quad & X_{ij} = M_{ij} \quad (i,j) \in \Omega \\ & \begin{bmatrix} W_1 & X \\ X^{\mathrm{T}} & W_2 \end{bmatrix} \succeq 0, \end{aligned} \tag{5}$$

where the optimization variables are W_1, W_2, and X.

However, the convex relaxation form as stated in (3) and (4) may not scale well for matrices of larger size. In this case, we can adopt to using an **alternating minimization** algorithm, which can be mathematically formulated as the following optimization problem:

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} (X_{i,j} - (UV^\top)_{i,j})^2, \tag{6}$$

where U, V are initialized from the top-k singular vectors of the singular value decomposition of X and Ω is the set of indices in which X is observed. We summarize this algorithm in Algorithm 1. In essence, we fix all the variables and optimize of U on one iteration, and then fix to update V on the next iteration. The stopping criteria is if the objective function value doesn't change based on threshhold. This iterates over an arbitrary T value. Since the multiplication of two variables is non-convex, the whole problem can be seen as non-convex, but is convex in each iteration if we use this alternating minimization scheme.

Algorithm 1 Alternating Minimization for Matrix Completion

Require: Data matrix X that satisfies matrix completion properties

1: **Initialize:** U and V from top-k singular vectors from SVD(X), where X takes values from observed values and 0 otherwise
2: **repeat**
3: **for** $t = 1, \ldots, T$ **do**
4: $V_t = \underset{V_t}{\text{argmin}} \quad \ell(U_{t-1}, V, X)$
$U_t = \underset{U_t}{\text{argmin}} \quad \ell(U, V_t, X)$
5: **end for**
6: **until** $\ell(\theta^{(t+1)}) - \ell(\theta^{(t)}) < \epsilon$

2.2 Duality

The dual form for this type of problem is rather difficult to derive as the nuclear norm is non-differentiable. In this paper, we will be doing a residual analysis, so we do not *need* to derive the dual form, however we think it is an interesting derivation worth mentioning. This is well summarized in a blog post by researchers at Yale, as well as by Candes and Recht [2]. Suppose we were trying to solve the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(x, y) \\ \text{subject to} \quad & g(x, y) = c. \end{aligned} \tag{7}$$

A necessary condition for the Lagrangian duality is that there exists a solution at (x, y) with some λ for the equation,

$$L(x, y, \lambda) = \nabla_{x,y} f - \lambda \nabla_{x,y} g = 0 \tag{8}$$

From the equation in (7), we have a linear combination of the objective function and the equality constraints. In our problem, the equality constraints are that the observations that we have in matrix M are the same as the entries for the solved matrix in X, and can be formulated as

$$||X||_* = \sum_{k=1}^{m} \lambda_k (X_{i,j} - M_{i,j}), \tag{9}$$

where m is the total number of observations given in matrix M. Thus, taking gradient with respect to X, we have the dual form as the following:

$$\nabla ||X||_* - \sum_{k=1}^{m} \lambda_k e_{i,j} = 0, \tag{10}$$

where e denotes the basis vectors. However, the difficulty lies in that we need to use sub-gradients for the nuclear norm. This is outside the scope of this project, but

Figure 2: Left – 5×5 synthetically generated matrix with integer entries, right – matrix with missing values, where NaN indicates missingness

1.00002	5.00002	5.99996	8.99998	3
4.00001	1.00001	4.99992	3.00006	7.99997
8.00005	5.12268e-05	4.00003	7.99995	8.99998
8.99993	1.99995	6.00001	7	4.00003
2.99995	4.99993	1.00006	2.99997	5.99999

Figure 3: Matrix solved using CVXPY with optimization problem stated in (4)

can be further explored in the previously mentioned citations. I thought this was something cool worth mentioning from what we learned in class (assuming it was derived correctly).

3 Experimental Results

We used two types of data for our experiments: synthetic data and the MovieLens dataset [6]. The objective of both experiments is to investigate the matrix completion optimization problem as defined in (3) and (6). For all experiments, we use a Python and MATLAB environment on a Macbook Pro with 2.2 GHz Intel Core i7 and 16 GB RAM.

3.1 Synthetic Data

We first test the matrix completion method as described in (3) and (4) on some synthetically generated data. Let $X \in \mathbb{R}^{16\times 16}$, where all the entries of X are positive values. We construct X so that we can solve the optimization problem in (4) to get the missing values of the matrix as shown in Figure 2. In Figure 2, we can see a portion of the matrix, with the filled and missing entries. After setting up the problem in CVX, we obtain the matrix solution as shown in Figure 3. We can observe that the resulting matrix is close, with a squared error of 0.0002. This was to show that the nuclear norm minimization and the trace minimization formula works

0.160294	0.619608	-3.71606	0.549835
-0.240334	-0.342301	-0.268995	-0.0642022
0.434015	-0.0467792	7.10525	-0.745615
-0.505507	-0.667647	-1.08678	-0.0672313
0.637654	1.14648	-1.65844	0.479086
1.45146	1.68891	5.39124	-0.102512

0.160294	0.619608	-3.71606	0.549835
-0.240334	-0.342301	-0.268995	-0.0642022
0.434015	-0.0467792	7.10525	-0.745615
-0.505507	-0.667647	-1.08678	-0.0672313
0.637654	1.14648	-1.65844	0.479086
1.45146	1.68891	5.39124	-0.102512

Figure 4: Left – snippet of true 100×100 matrix, right – snippet of solved 100×100 matrix using alternating minimization as in (6)

efficiently for matrix completion.

Further, we test the methods of alternating minimization as described in (6). We make a new data matrix, with entries drawn $\mathcal{N}(0, 1)$ with dimensions 100×100. We initialize U and V to be the top–2 singular vectors from the singular value decomposition (SVD) of the data matrix. Upon solving the optimization problem, programmed in Python using CVXPY, we can see the results in Figure 4. From the entries in Figure 4, we can see that the we almost get an exact approximation of the solved data matrix. The squared error for this experiment was nearly 0, but that doesn't necessarily mean that it outperformed the previous trace norm experiment. Our entries in Figure 4 were significantly smaller than those in Figure 2, and can just "luckily" get a smaller error.

3.2 MovieLens Dataset

For the real dataset experiment, we use the MovieLens Dataset [6]. This dataset is comprised of 100,000 user ratings of 1 to 5 from 943 users on 1682 movies. The data matrix has been pre-processed so that each user has at least rated 20 different movies. To perform matrix completion on this data matrix, we randomly take out some values from the **ratings** column of the matrix and use CVX to solve for the missing entries. We can observe a snippet of these matrices in Figure 5, and the empty + solved matrices in Figure 6. In Figure 5, we have a snippet of the data matrix that is originally of dimensions $(100,000 \times 4)$, where the columns are the user ID number, movie ID number, rating, and timestamp (when the rating was made), respectively. In Figure 6, we can observe the results from CVX. The squared error was approximately 0.0015, however, we can see that the ratings were not as accurate as the other entries. For example, user 3 had a true rating of 1, but the matrix completer gave a rating of 3.4. We believe this could be based on the size of the matrix, as maybe it will not generalize to data matrices of higher dimensions.

Index	user_id	item_id	rating	timestamp
0	196	242	3	881250949
1	186	302	3	891717742
2	22	377	1	878887116
3	244	51	2	880606923

Figure 5: Snippet of data matrix from MovieLens dataset

196	242	3.79912	8.81251e+08
185.999	302.001	3.77992	8.91718e+08
21.9965	377.002	3.4171	8.78887e+08
244	51	4.02175	8.80607e+08
165.996	346.002	3.69212	8.86397e+08
298	474	3.78127	8.84183e+08

196	242	3	881250949
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923
166	346	1	886397596
298	474	4	884182806

Figure 6: Left – Solved matrix using CVX, right – true matrix from MovieLens dataset

4 Residual Analysis

In this section, we explore residual analysis of the results from the synthetic case for the $X \in \mathbb{R}^{16\times 16}$ data matrix, as it had better results. Since the optimization problem in (6) is a norm minimization (sum of squares can easily be represented as ℓ-2 norm), we can define the penalty function as $\phi(u) = u^2$. Moreover, we also use different types of penalty functions to analyze the residuals as defined in the next section.

4.1 Penalty Functions

We dedicate this section to define the penalty functions to be used to analyze our residuals. We use these functions to plot a residual histogram, similar to the histograms displayed in the Boyd text [10].

1. The **quadratic** penalty function is

$$\phi(u) = u^2$$

2. The **deadzone–linear** penalty function with $a > 0$ is given by

$$\phi(u) = \begin{cases} 0, & |u| \leq a \\ |u| - a, & |u| > a. \end{cases}$$

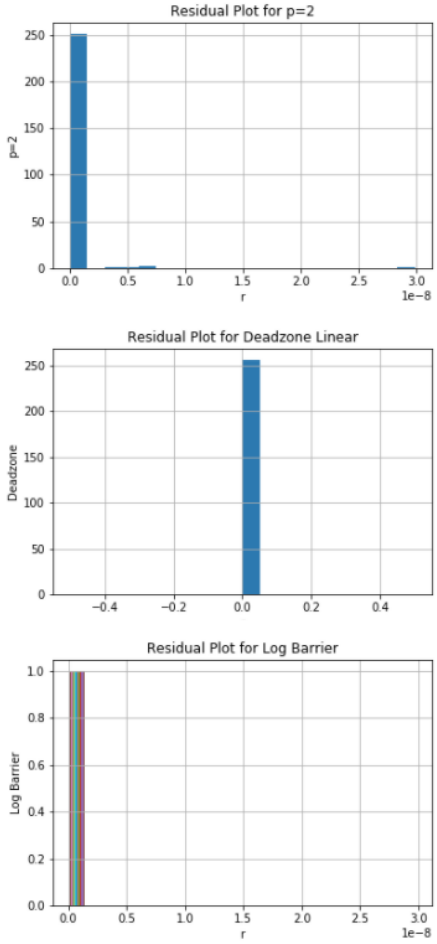

Figure 7: Residual plots with penalty functions defined

3. The **log barrier** penalty function with limit $a > 0$ is given by

$$\phi(u) = \begin{cases} -a^2 \log(1 - (\frac{u}{a})^2), & |u| < a \\ \infty, & |u| \geq a. \end{cases}$$

With the penalty functions defined above, we make the residual histogram as shown in Figure 7. In Figure 7, we can see that most of the penalty function values are centered around nearly 0, which is because our results were an almost exact approximation. Since the error was so low, we can predict most of our residuals to be around 0. The residuals are around zero, as they take values of approximately 1×10^{-7}. The only ones where we can see a slight more action is for the ℓ_2 penalty function, with slight values away from 0.

5 Discussion

In this paper, we investigate a problem of practical interest: matrix completion. Matrix completion has gained wide popularity, especially with the infamous Netflix prize problem. Through this, there are lots of researchers publishing work for the properties and algorithms for matrix completion, which we explored here. We view the nuclear norm minimization and the scheme of alternating minimization for matrix completion, and performed a residual analysis on synthetic data. We perform matrix completion on both synthetic and real data and display the results for both.

Link to MovieLens dataset: `https://grouplens.org/datasets/movielens/`.

References

[1] R. Baraldi, C. Ulberg, R. Kumar, K. Creager, and A. Aravkin, "Relaxation algorithms for matrix completion, with applications to seismic travel-time data interpolation," 08 2018.

[2] E. Candès and B. Recht, "Exact matrix completion via convex optimization," *Communications of the ACM*, vol. 9, pp. 717–772, 11 2008.

[3] Y. Chen and Y. Chi, "Robust spectral compressed sensing via structured matrix completion," *IEEE Transactions on Information Theory*, vol. 60, 04 2013.

[4] S. Gunasekar, A. Acharya, N. Gaur, and J. Ghosh, "Noisy matrix completion using alternating minimization," vol. 8189, pp. 194–209, 09 2013.

[5] C. Gómez-Uribe and N. Hunt, "The netflix recommender system," *ACM Transactions on Management Information Systems*, vol. 6, pp. 1–19, 12 2015.

[6] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," *ACM SIGIR Forum*, vol. 51, pp. 227–234, 08 2017.

[7] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," *Proceedings of the Annual ACM Symposium on Theory of Computing*, 12 2012.

[8] H. Liu, X. Kong, X. Bai, W. Wang, T. Bekele, and F. Xia, "Context-based collaborative filtering for citation recommendation," *IEEE Access*, vol. 3, pp. 1–1, 01 2015.

[9] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, pp. 2114 – 2121, 11 2009.

[10] H.-J. Lüthi, "Convex optimization: Stephen boyd and lieven vandenberghe." *Journal of the American Statistical Association*, vol. 100, pp. 1097–1097, 02 2005.

[11] W. Ma and G. H. Chen, "Missing not at random in matrix completion: The effectiveness of estimating missingness probabilities under a low nuclear norm assumption," 2019.

[12] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Matrix factorization and neighbor based algorithms for the netflix prize problem," *RecSys'08: Proceedings of the 2008 ACM Conference on Recommender Systems*, pp. 267–274, 01 2008.