EE 638 Project: Compressive recovery of a low-rank matrix

Arunabh Ghosh

November 26, 2018

1 Introduction

In many practical problems of interest, one would like to estimate a matrix from a sampling of its entries. Formally, we may view this problem as follows. We are interested in estimating a data matrix $\mathbf{M}$ with n_1 rows and n_2 columns but only get to observe a number m of its entries which is comparably much smaller than $n_1 n_2$, the total number of entries. Can one recover the matrix $\mathbf{M}$ from m of its entries? In general, everyone would agree that this is impossible without some additional information.

In many instances, however, the matrix we wish to estimate is known to be structured in the sense that it is low-rank or approximately low-rank. (We recall for completeness that a matrix with n_1 rows and n_2 columns has rank r if its rows or columns span an r-dimensional space.) Given below is a practical scenario where one would like to be able to recover a low-rank matrix from a sampling of its entries.

The Netflix problem - In the area of recommender systems, users submit ratings on a subset of entries in a database, and the vendor provides recommendations based on the user's preferences [1, 2]. Because users only rate a few items, one would like to infer their preference for unrated items. A special instance of this problem is the now famous Netix problem [3]. Users (rows of the data matrix) are given the opportunity to rate movies (columns of the data matrix), but users typically rate only very few movies so that there are very few scattered observed entries of this data matrix. Yet one would like to complete this matrix so that the vendor (here Netix) might recommend titles that any particular user is likely to be willing to order. In this case, the data matrix of all user-ratings may be approximately low-rank because it is commonly believed that only a few factors contribute to an individual's tastes or preferences.

In fact, this problem occurs in many areas of engineering and applied science such as machine learning [4, 5], control [6] and computer vision, see [7]. In this project, I present an Alternating Direction Method of Multipliers (ADMM) algorithm that has been widely used for solving several convex and non-convex optimization problems by breaking them into smaller sub-problems. It has also been applied successfully to a number of statistical problems [8], image restoration and denoising [9, 10] and support vector machines [11].

Further, I even went beyond and improved the results presented in the paper [12] by imposing a sparsity constraint on the matrix in the DCT basis. This is a common condition for natural images, and I demonstrate a significant improvement in reconstruction results.

2 Mathematical Formulation and Literature Survey

The mathematical formulation of the matrix completion (MC) problem, that aims to find the lowest rank matrix from its known entries, is as follows:

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

where $rank(\mathbf{X})$ denotes the rank of $\mathbf{X}$ and Ω is the set of observed entries of the unknown matrix. Different algorithms are proposed in the literature to find the solution of (1), for example, see [13]. All these algorithms use singular value decomposition (SVD) that is expensive when the size of the matrix increases. Thus, other algorithms based on simple factorization are widely used for model (1) instead of SVD, for example in [14, 15]. The authors in [14] proposed to solve

$$\begin{aligned} &min\ \frac{1}{2}||XY - Z||_F^2 \\ &s.t\ \ Z_{ij} = M_{ij}, \quad (i,j) \in \Omega \end{aligned} \tag{2}$$

In this project, instead of model (2), we propose to solve the following problem:

$$\begin{aligned} &min\ \frac{1}{2}||XUY - Z||_F^2 \\ &s.t\ \ Z_{ij} = M_{ij}, \quad (i,j) \in \Omega \end{aligned} \tag{3}$$

where M is a matrix with known entries with indices in set Ω, $X \in R^{m\times s}$, $U \in R^{s\times s}$, $Y \in R^{s\times n}$ and $Z \in R^{m\times n}$, s is integer denoting our estimated rank. Let the rank of data matrix M be a and

$$P_\Omega(M) = M_{ij} \quad if \quad (i,j) \in \Omega \quad else \quad 0 \tag{4}$$

Model (3) is a non-convex quadratic optimization problem because of XUY term in the objective function. The combination of three matrices XUY instead of just two matrices XY in objective function allows us to control the rank of XUY with dimension of U, because U is a full rank square matrix and $s < min\{m,n\}$. The rank estimate s should ideally be equal to the rank of the data matrix M because the correct rank is generally unknown. In the case of uncertainty, we may experiment on multiple rank estimates and choose the one which gives the best reconstruction result

To solve model (3), we present an Alternating Direction Method of Multipliers (ADMM) algorithm that has been widely used for solving several convex and non-convex optimization problems by breaking them into smaller subproblems. In Section 2 we present the ADMM algorithm. Comparison of ADMM algorithm with Opts [16] is given in Section 3 on several test problems.

3 ADMM Algorithm

In mathematical optimization, the Karush–Kuhn–Tucker (KKT) conditions, also known as the Kuhn–Tucker conditions, are first derivative tests (sometimes called first-order) necessary conditions for a solution in nonlinear programming to be optimal. Therefore the solution must satisfy the KKT conditions.
The KKT conditions for (3) are as follow:

$$\begin{aligned} &\frac{\partial L}{\partial X} = (XUY - Z)(UY)^T = 0 \\ &\frac{\partial L}{\partial Y} = (XU)^T(XUY - Z) = 0 \\ &\frac{\partial L}{\partial U} = (X)^T(XUY - Z)(Y)^T = 0 \\ &\frac{\partial L}{\partial Z} = (Z - XUY) + \Lambda = 0 \\ &\frac{\partial L}{\partial \Lambda} = P_\Omega(Z - M) = 0 \end{aligned} \tag{5}$$

Now consider the augmented Lagrangian associated to (3) as follows:

$$L_\rho(X,U,Y,Z,\Lambda) = \frac{1}{2}||XUY - Z||_F^2 + \Lambda P_\Omega(Z - M) + \frac{\rho}{2}||P_\Omega(Z - M)||_F^2 \tag{6}$$

where $\Lambda \in R^{m\times n}$ is the Lagrange multiplier and $\rho > 0$ is the penalty parameter.

The steps of ADMM can be outlined as follow:

$$X_{k+1} \leftarrow \arg\min_{X} L_\rho(X, U_k, Y_k, Z_k, \Lambda_k),$$

$$Y_{k+1} \leftarrow \arg\min_{Y} L_\rho(X_{k+1}, U_k, Y, Z_k, \Lambda_k),$$

$$U_{k+1} \leftarrow \arg\min_{U} L_\rho(X_{k+1}, U, Y_{k+1}, Z_k, \Lambda_k),$$

$$Z_{k+1} \leftarrow \arg\min_{Z} L_\rho(X_{k+1}, U_{k+1}, Y_{k+1}, Z, \Lambda_k),$$

$$\Lambda_{k+1} = \Lambda_k + \gamma\rho P_\Omega(Z_{k+1} - M).$$

Since Y, U, Z and Λ are fixed matrices, the above problems are convex quadratic optimization problem. Specifically, the above steps are the following due to the convexity:

$$X_{k+1} = (Z_k (U_k Y_k)^T)((U_k Y_k)(U_k Y_k)^T)^{-1},$$

$$Y_{k+1} = ((X_{k+1} U_k)^T (X_{k+1} U_k))^{-1}((X_{k+1} U_k)^T Z_k),$$

$$U_{k+1} = (X_{k+1}^T X_{k+1})^\dagger X_{k+1}^T Z_k Y_{k+1}^T (Y_{k+1} Y_{k+1}^T)^\dagger,$$

$$Z_{k+1} = X_{k+1} U_{k+1} Y_{k+1} + P_\Omega(M - X_{k+1} U_{k+1} Y_{k+1}) - \frac{1}{\rho}\Lambda_k,$$

$$\Lambda_{k+1} = \Lambda_k + \gamma\rho P_\Omega(Z_{k+1} - M),$$

(7) Iterative steps for solving the ADMM algorithm

where $A^\dagger$ is the Moore-Penrose pseudoinverse of A and γ is the step size. Now, the ADMM algorithm for solving 3 can be outlined as follows.

Algorithm 1 ADMM algorithm

```
$U_0, Y_0, Z_0 \leftarrow$ **input** matrices
$U_0, Y_0 \leftarrow$ Identity matrix
$Z_0 \leftarrow P_\Omega(M)$

Set $\rho > 0, maxIter > 0$ and $k = 0$

**for** $k = 1 : maxIter$ **do**
    Perform steps outlined in (7)
**end for**

**Return** $X_{k+1}, U_{k+1}, Y_{k+1}, Z_{k+1}, \Lambda_{k+1}$
```

4 Results

In this section, the algorithm presented above is tested on incomplete images. The problem statement is this - We test our algorithm on a partially observed image and attempt to estimate the original image. In our experiment, the image size is 200×200 and the rank of the image is assumed to be 30. The number of elements observed is increased from 30% to 60% and the reconstruction result is observed. Here is what the original image looks like -

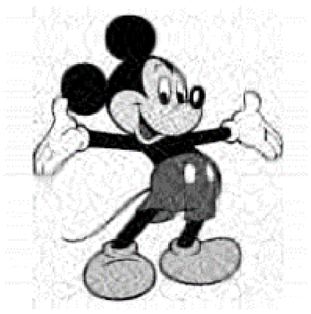
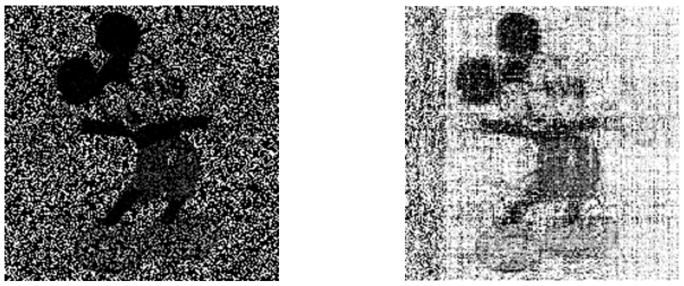


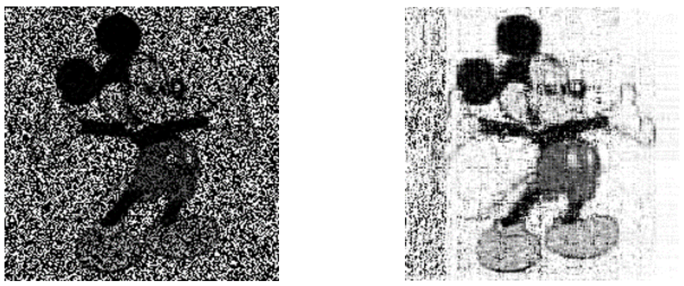
Figure 1: Original rank 30 image

The error metric used is the Relative Mean Squared Error between the reconstruction and the test image. This is defined as follows: $\text{RMSE}(z, \hat{z}) = \frac{\|z - \hat{z}\|_2}{\|z\|_2}$ where $\hat{z}$ is the reconstructed estimate for z. Here is what the reconstruction results look like.



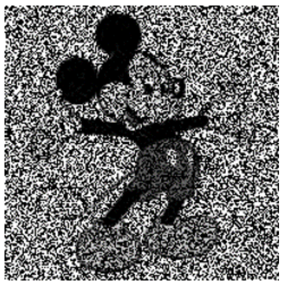
(a) Observed Image (b) Reconstructed Image

Figure 2: 30% elements observed, RMSE = 0.4621

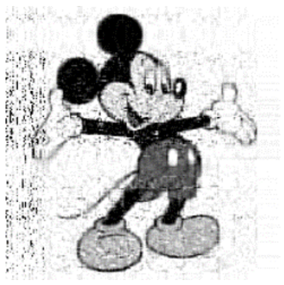


(a) Observed Image (b) Reconstructed Image

Figure 3: 40% elements observed, RMSE = 0.3129

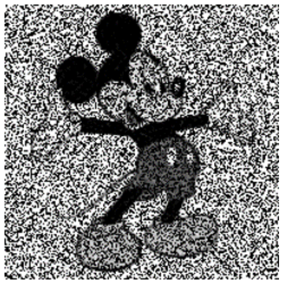


(a) Observed Image

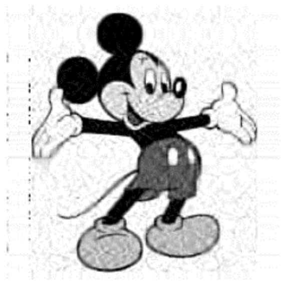


(b) Reconstructed Image

Figure 4: 50% elements observed, RMSE = 0.0890



(a) Observed Image



(b) Reconstructed Image

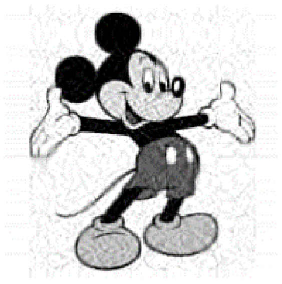
Figure 5: 60% elements observed, RMSE = 0.0430

5 Compressed sensing

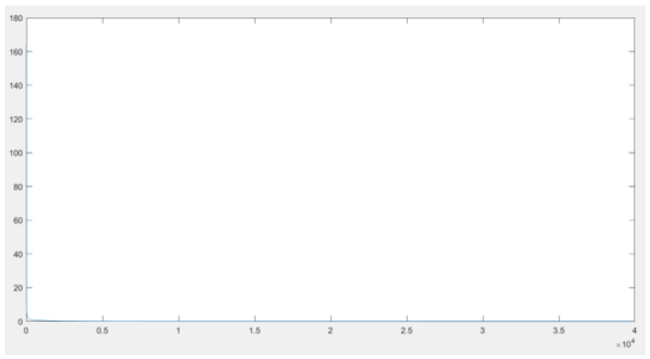
Compressed sensing or CS is a novel sensing/sampling paradigm that goes against the common wisdom in data acquisition. CS theory asserts that one can recover certain signals and images from far fewer samples or measurements than traditional methods use. To make this possible, CS relies on a principle called sparsity.

Sparsity expresses the idea that the "information rate" of a continuous time signal may be much smaller than suggested by its bandwidth, or that a discrete-time signal depends on a number of degrees of freedom which is comparably much smaller than its (finite) length. More precisely, CS exploits the fact that many natural signals are sparse or compressible in the sense that they have concise representations when expressed in the proper basis.

Consider, for example, the image in Figure 6(a) and its DCT transform in (b). Although nearly all the image pixels have nonzero values, the DCT coefficients offer a concise summary: most coefficients are small, and the relatively few large coefficients capture most of the information.



(a) Original Image



(b) DCT coefficients sorted from highest to lowest

Figure 6: Sparsity of DCT coefficients of natural images

5.1 Incorporating the sparsity constraints

We modify the previous optimization problem to the following –

$$L_\rho(X,U,Y,Z,\Lambda) = \frac{1}{2}||XUY - Z||_F^2 + P_\Omega(Z-M) + \frac{\rho}{2}||P_\Omega(Z-M)||_F^2 + ||u||_1 \tag{7}$$

where u is is the DCT coefficients of Z. We impose a sparsity constraint on Z by imposing the L1-norm penalty factor in the DCT basis.

Now all the steps remain the same as before, except for the update step of Z which now has an additional penalty factor. We update Z in two steps, Forward step and Backtracking step. The forward step simply optimizes the previous optimization problem as follows -

$$Z_{k+1} = X_{k+1}U_{k+1}Y_{k+1} + P_\Omega(M - X_{k+1}U_{k+1}Y_{k+1}) - \frac{\Lambda}{\rho} \tag{8}$$

I add an additional backtracking step which solves the following optimization problem -

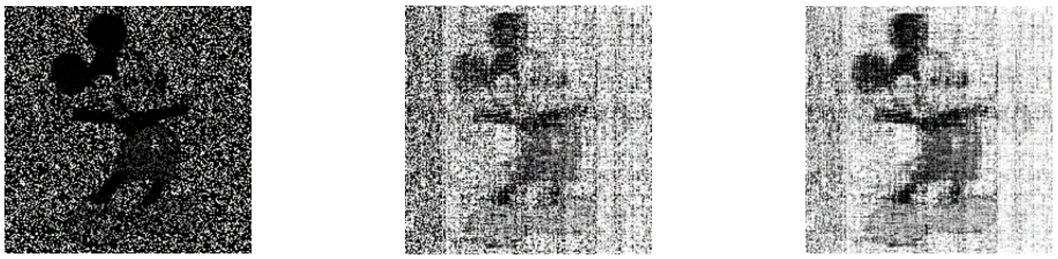
$$min||u||_1 + ||Du - Z_{k+1}||^2 \tag{9}$$

where D is the DCT operator. In this operation, we are essentially trying to find the best sparse representation of the updated value of Z in the DCT basis. We write out the new updated Z after the backtracking step as follows –

$$Z_{final} = D^{-1}u \tag{10}$$

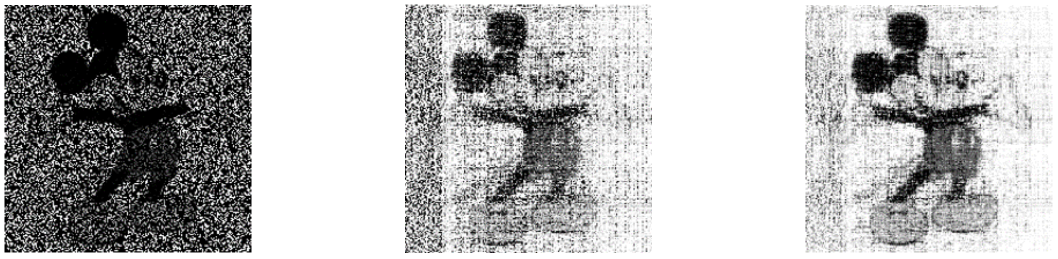
6 Results - with compressed sensing

Now we put in the sparsity constraint and again test our algorithm on partially observed images. The L1-norm optimization in the backtracking step is solved using the L1-LS package which is an interior-point method for l_1-regularized least squares problem [16]. Given below are the reconstructions in the case of imposing the sparsity constraint and not imposing the sparsity constraint.



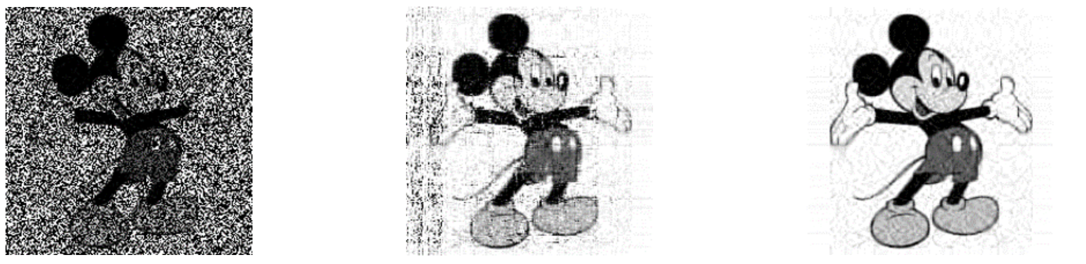
(a) Original Image (b) Without sparsity constraint (c) With sparsity constraint

Figure 7: 25% elements observed, RMSE(b) - 0.5721, RMSE(c) - 0.3434



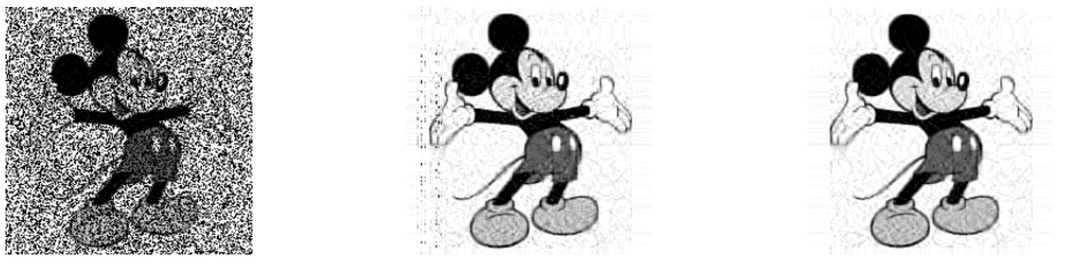
(a) Original Image (b) Without sparsity constraint (c) With sparsity constraint

Figure 8: 30% elements observed, RMSE(b) - 0.4584, RMSE(c) - 0.2444



(a) Original Image (b) Without sparsity constraint (c) With sparsity constraint

Figure 9: 45% elements observed, RMSE(b) - 0.2265, RMSE(c) - 0.0266



(a) Original Image (b) Without sparsity constraint (c) With sparsity constraint

Figure 10: 60% elements observed, RMSE(b) - 0.0713, RMSE(c) - 0.0101

7 Conclusion and Future Work

Thus we have seen how to recover a structured low-rank matrix from just a subset of the observations. Further, if the matrix has a sparse representation in a basis, we have developed an algorithm based on the principles of compressed sensing, to improve reconstruction results. The code for this project can be found here - *https://github.com/Arunabh98/matrix-completion.*

There are still multiple avenues in which the work presented here, may be taken forward. This algorithm requires us to have prior knowledge about the rank of the original matrix. Although it maybe estimated from the observed matrix, a deviation from the actual rank may deteriorate our results. Secondly, we have to recognize the basis in which the matrix has a sparse representation. Natural images, mostly have a sparse representation in the DCT basis but our algorithm will not just be applied to natural images. For example, in the Netflix problem it may be difficult to recognize an appropriate basis. In such cases, we may learn an appropriate basis through dictionary learning [17]. It would be interesting to see how accurate reconstructions we might get, if we combine dictionary learning along with the algorithm presented here.

References

[1] J. D. M. Rennie and N. Srebro, "Fast Maximum Margin Matrix Factorization for Collaborative Prediction," tech. rep., 2005.

[2] N. Srebro, "Learning with Matrix Factorizations," tech. rep., 2004.

[3] "ACM SIGKDD and Netflix."

[4] J. Abernethy, F. Bach, and J.-P. Vert, "Low-rank matrix factorization with attributes," tech. rep., 2006.

[5] "Uncovering Shared Structures in Multiclass Classification," tech. rep.

[6] M. Mesbahi and G. Papavassilopoulos, "On the rank minimization problem over a positive semidefinite linear matrix inequality," *IEEE Transactions on Automatic Control*, vol. 42, no. 2, pp. 239–243, 1997.

[7] C. Tomasi and T. Kanade, "Shape and Motion from Image Streams under Orthography: a Factorization Method," Tech. Rep. 2, 1992.

[8] J. M. Bioucas-Dias and M. A. T. Figueiredo, "Alternating direction algorithms for constrained sparse regression: Application to hyperspectral unmixing," in *2010 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, pp. 1–4, IEEE, 6 2010.

[9] M. A. T. Figueiredo and J. M. Bioucas-Dias, "Restoration of Poissonian Images Using Alternating Direction Optimization," *IEEE Transactions on Image Processing*, vol. 19, pp. 3133–3145, 12 2010.

[10] M. K. Ng, P. Weiss, and X. Yuan, "Solving Constrained Total-variation Image Restoration and Reconstruction Problems via Alternating Direction Methods," *SIAM Journal on Scientific Computing*, vol. 32, pp. 2710–2736, 1 2010.

[11] P. A. Forero, A. Cano, and G. Edu, "Consensus-Based Distributed Support Vector Machines Georgios B. Giannakis," tech. rep., 2010.

[12] R. Taleghani and M. Salahi, "AMO-Advanced Modeling and Optimization. Low rank matrix completion by alternating direction method of multipliers," *AMO-Advanced Modeling and Optimization*, vol. 20, no. 2, 2018.

[13] R. Meka, P. Jain, and I. S. Dhillon, "Guaranteed Rank Minimization via Singular Value Projection *," tech. rep., 2009.

[14] Z. Wen, W. Yin, and Y. Zhang, "SOLVING A LOW-RANK FACTORIZATION MODEL FOR MATRIX COMPLETION BY A NONLINEAR SUCCESSIVE OVER-RELAXATION ALGORITHM," tech. rep.

[15] J. P. Haldar and D. Hernando, "Rank-Constrained Solutions to Linear Matrix Equations Using PowerFactorization," 2009.

[16] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An Interior-Point Method for Large-ScalèScalè 1-Regularized Least Squares," *IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING*, vol. 1, no. 4, 2007.

[17] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, vol. 54, no. 11, 2006.