Thesis Proposal: Large-Scale Matrix Completion and Recommender Systems

Lily Amadeo

July 3, 2015

1 Motivation

We are in an age where data is growing at immense speeds, and solving problems has become as much about the technique for analysis as it has handling the data. Making algorithms work for data sets that are very large, (even wanting to consider exabyte scale of 10^{18} bytes) is not only interesting, but absolutely necessary for continued advancement in the field. Principal Component Analysis (PCA) is one of the foundational algorithms of data science, and it is an extremely popular statistical technique. Given how applicable it is, PCA has been widely researched and has had numerous extensions created to handle various circumstances that a user encounters within a data set. In particular, it is often used for dimensionality reduction, where it has been utilized for image processing, data compression, network mapping, and many more real-world applications [1–3]. In particular, PCA is a foundational element in solving the *matrix completion problem* [4]. Exploring techniques for making these algorithms work on a large-scale is the focus of my thesis.

Matrix completion is important, because it allows data to be represented in a new, often smaller, set of variables, and it allows PCA to be done in a scalable way. Matrix completion is the situation when a matrix is simply missing some, or many, entries. The user is aiming to fill in those missing entries using some principled approach. It is a hard problem, but under some assumptions and conditions on the matrix, it can be accomplished with high probability of success. This situation comes up many times in practical business problems. Classic examples of such problems include recommender systems [5]. A famous example of recommender systems is the Netflix Prize[6]. Given a set of user's ratings of movies, we wish to predict what users will rate movies they have not seen. Essentially, the problem

becomes completing a matrix of partially observed data, where the missing entries are movies that a user has not rated. In this thesis, we work on both the theory and practice of such methods, and one of my tests on real-world data will be an example of such a recommender system. In addition, among other examples, we will treat mapping of computer networks based upon hop distance. Between these two problems, we expect to demonstrate the applicability of large-scale matrix completion to problems of practical interest.

2 Background

The matrix completion problem has been explored extensively in the literature[4]. In a paper presented in 2010 by Emmanuel Candes and Terence Tao,[7], the idea of using *convex relaxation* made the problem tractable for large-scale data. Of course, one might wonder, given a matrix M, and only a sample m of its entries, is it possible to recover the remaining entries and complete the matrix? Intuition says no, of course not. The missing entries could be anything. How can they be recovered? Obviously, one has to make some kind of modeling assumption, and a standard modeling assumption, in this domain, is that M is *low-rank*. When M is low-rank, it in fact is very possible to recover the missing entries given the set m. If M is approximately low-rank, and enough entries m are given, then there is a high probability that M can be recovered. This problem is solved efficiently using convex optimization. Interestingly, PCA is commonly viewed as being solved using Singular Value Decomposition (SVD), and is commonly done so using this linear algebra technique. On the other hand, the original definition of PCA was from an optimization viewpoint. Solving a matrix rank minimization problem (while adhering to the constraint that the given entries of M match the low-rank component's entries) is another version of PCA. PCA can be set up as the following optimization problem: Given a matrix M, with a low rank component L_0 and a noise component N_0, ie $M = L_0 + N_0$, solve:

$$\begin{aligned} L = \arg\min \|M - L\|_F^2, \\ \text{s.t. } \operatorname{rank}(L) \leq k, \end{aligned} \tag{1}$$

where $\|M - L\|_F$ is the Frobenius norm, or the sum of the squares of the entries. Surprisingly, there are many advantages to using the original optimization version of the problem. Take for example a brief look at speed. The SVD version of solving this problem takes $O(n^3)$ operations. For a matrix with 10^6 entries, computing the SVD of such a data set would take, leaving

room for a constant adjustment, approximately 10.6 years (assuming a 3 gH processor). A linear time version of the problem, which is offered by our current work, and is $O(n)$, would take under one second (with room for a constant time adjustment). Clearly this is a substantial improvement, but, it comes with a cost, and that cost is a detailed understanding of the interaction between convex optimization and the particular problems of interest. It is the study of this interaction that is the core of my thesis. Modern extensions of PCA typically work on this optimization problem as a starting point, as it allows for these very fast computational approaches. Extending this technique to problems where the matrix includes corrupted data gives us Robust Principal Component Analysis (RPCA), also solved using an optimization problem. Lastly, the eRPCA technique aims to solve a similar problems for data which has some corruption or uncertainty in all points, and potentially large amounts of it.

3 Methods

The applicability of the work I am completing in this thesis will be shown in the utilization of the algorithm. This will require all the experimentation on improving the convergence rate of the algorithm to be applied to real-world data sets. So far, my results have been extremely positive in improving run time and algorithm iteration count on synthetic data. Extending these results into data from two different real-world domains will demonstrate the use of this pursuit. We will describe two real-world data sets in what follows.

First, we will work on the movie lens data set [8]. This is a data set collected by the GroupLens research group at the University of Minnesota, Twin Cities. It offers various sizes of movie rating data over different periods of time. Included is demographic information about the users, though we are mainly concerned with using the ratings for a matrix completion problem.

Second, we will work on internet tomography. This is the study that deals with mapping the internet. The internet can be thought of as a series of interconnected nodes that send and receive information from each other. There are various distance measures that can be used to describe this node graph. Euclidean distance is one. Hop distance, or, the number of nodes data must pass through to reach a target node, is another. Transmission or latency time is another measure that can be used as a proxy for distance. Currently, theorems have been developed stating that the Euclidean distance between nodes form a matrix with at most rank 4 (assuming the nodes lie on a plane) [9]. This is a low-rank matrix, which is exactly the type my work

has focused on. However, there are not currently similar theorems about hop distance or transmission time, though preliminary results indicate these problems are low-rank as well. Both of these problems offer incomplete matrices that I will be applying the algorithm to in order to fill in the matrix, make predictions, and show convergence.

4 Work So Far

Since matrix completion is such a rich algorithm to explore, I looked into a number of different pieces of extensions, such as eRPCA, that would allow for improved matrix completion results, and accurate results that required *less time and algorithm iterations*. These included: parameters within the algorithm and how they could be changed based on the size of the data, exploring how the size and sparsity of the data affected the algorithm's performance at a large scale, and how different error allowances changed the results from the algorithm. The main parts of the algorithm I explored were two parameters that help to control for convergence. With some manipulation detailed in [10] and [11], the optimization problem for eRPCA becomes the following relaxed Lagrangian problem (up to a constant):

$$\mathcal{L}(L, S, Y, \mu) = \|L\|_* + \lambda\|\mathcal{S}_\epsilon(P_\Omega(S))\|_1 + \frac{\mu}{2}\|P_\Omega(M - L - S) + \frac{1}{\mu}Y\|_F^2 \quad (2)$$

μ is a multiplier that changes the algorithm's focus from the objective (the optimization problem), to the constraint (that the recovered matrix's entries must match those given). ρ is a parameter that helps to change how the value of μ grows. In the algorithm's initial iterations, μ needs to be relatively small, so that the objective is met. At each iteration, μ may potentially increase, if the objective value is reasonably minimized. ρ controls for this potential update of μ. A large μ value means the focus of the problem is solely on satisfying the constraint. A larger μ value will always produce a faster algorithm.

By varying parameters within the eRPCA algorithm, I was able to substantially improve the time it takes to solve the optimization problem, while maintaining accuracy. I was able to take the original values for μ and ρ (based on [12]), and through empirical study, reduce algorithm completion time and iteration count to levels that are much more approachable and able to be used on large data sets.

Figure 1 is an example of the work I have done so far. The best value for the matrix with .1% of values observed, and that with 1% observed, were

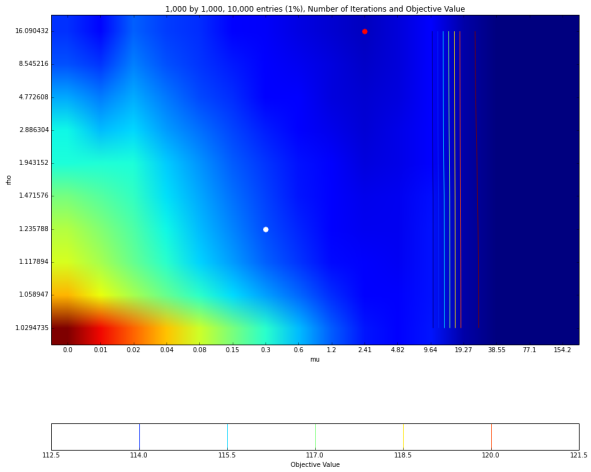

Figure 1: Results for a 1,000 by 1,000 matrix with 10,000 entries. The color scale shows that the algorithm runs with the least amount of iterations (dark blue) at the largest values of μ and ρ. However, the contour lines show that the objective value is no longer minimized when that μ value is too large. The white dot represents the default μ ρ pair. The red dot represents the μ and ρ pair that requires the least iterations while maintaining the minimum objective. As the thesis progresses, we plan to treat much larger problems.

the largest values of μ and ρ tested, when "best" refers to number of iterations or solving time. However, for values that large, one would expect the algorithm to fail, as it would be focusing solely on the constraint for such a large μ. This does indeed occur. The contour lines in plot 1 show that past a certain large μ value, the objective is no longer at a minimum. The problem then becomes finding the best value of μ and ρ that not only minimizes solving time and iterations, but also solves the problem correctly. In this case, the original values for μ and ρ (based upon theoretical considerations) required 58 iterations of the algorithm, while the best pairing (based upon a data analysis) only required 18 iterations. Since our algorithm is $O(n)$, this means that even for such a small problem, our algorithm allows us to treat

data which is three times larger, in the same amount of time. We expect that the improvement will be even more for larger problems.

5 Evaluation

Evaluating the success or failure of my work will be accomplished in multiple ways. For the initial tests determining if I can improve the algorithm's performance on synthetic data, I have been relying on mostly numerical, empirical results. The actual improvement of the algorithm can be expressed in percentage speed up, or reduction in algorithm iterations. So far, this has shown to give positive, worthwhile results. For the real-world data applications that will follow, I will be both testing algorithm performance in time and iterations, and in *prediction accuracy*.

6 To Complete

As it stands, most of my experimentation on synthetic data is complete. I have results and am continuing to put them into context, and to understand exactly how impactful they are. In addition, I am continuing to actually write about my work in its final form. I will continue to work on some additional detailed, side pursuits on this synthetic data, and see if there are any other interesting insights to be gleaned from the experiments. This will continue until my thesis is complete and turned in, and I will include anything that proves sufficiently interesting.

The main piece of work I have left is to apply my results to two real-world problems, including large-scale recommender systems and internet tomography. I will be starting this work this week (July 1st) and continue to evaluate results for at least one week. I will be writing about any results I find and incorporating them into my thesis. The last two weeks of work will be filled with checking my work, continuing to write, and ensuring I have all my data and code organized so that others may repeat my work if they desire.

References

[1] Khaled Labib and V. Rao Vemuri. An application of principal component analysis to the detection and visualization of computer network attacks. September 2004.

[2] Seunghee Park, Jong-Jae Lee, and Chung-Bang Yun Daniel J. Inman. Electro-mechanical impedance-based wireless structural health monitoring using pca-data compression and k -means clustering algorithms. *Journal of Intelligent Material Systems and Structures*, 19, April 2008.

[3] Emmanuel J Candes, Xiodong Li, and John Wright. Robust principal component analysis? 2009.

[4] Emmanuel J. Candes and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9, December 2009.

[5] Emmanuel J. Cands Jian-Feng Cai and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SAIM Journal on Optimization*, 20, March 2010.

[6] Wikipedia. Netflix prize –wikipedia, the free encyclopedia, 2015. URL `https://en.wikipedia.org/wiki/Netflix_Prize`. [Online; accessed 29-July-2015].

[7] Emmanuel J Candes and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56, May 2010.

[8] GroupLens. Movie lens data set, April 1998. URL `http://grouplens.org/datasets/movielens/`.

[9] Nathan Krislock and Henry Wolkowicz. *Euclidean Distance Matrices and Applications*. Springer, September 2011.

[10] Randy Paffenroth, Philip du Toit, Ryan Nong, Louis Scharf, Anura Jayasumana, and Vidarshana Bandara. Space-time signal processing for distributed pattern detection in sensor networks. *Journal of Selected Topics in Signal Processing*, 6, January 2013.

[11] Randy C. Paffenroth, Ryan Nong, and Philip C Du Toit. On covariance structure in noisy, big data. *SPIE*, September 2013.

[12] Zhouchen Lin, Minming Chen, and Yi Ma. The augemented lagrange multiplier method for exact recovery of corrupted low-rank matrices. 2013.

[13] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 3, September 1936.

[14] N. Halko, P G Martinsson, and J A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53, May 2011.

[15] Benjamin Recht, Maryam Fazel, and Pablo A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equationsvia nuclear norm minimization. *SIAM Review*, 52, August 2010.

[16] Wikipedia. Exabyte - wikipedia, the free encyclopedia, 2015. URL `https://en.wikipedia.org/wiki/Exabyte`. [Online; accessed 29-July-2015].

[17] Wikipedia. Lagrangian relaxation- wikipedia, the free encyclopedia, 2014. URL `https://en.wikipedia.org/wiki/Lagrangian_relaxation`. [Online; accessed 29-July-2015].

[18] Tom Coughlin. In 10 years a single movie could generate close to 1 exabyte of content. *Forbes Magazine*, October 2014.

[19] Carly Page. Ee expects to handle an exabyte of data per year by 2018. *The Inquirer*, May 2015.

[20] Nikhil Swaminathan. Is dna the next frontier in pirvacy? *AlJazeera America*, May 2015.

[21] Jordan Novet. How facebook's cold storage preserves your photos. *Venture Beat*, May 2015.

[22] Hans-Peter Kriegal, Peer Kroger, Erich Schubert, and Arthur Zimek. A general framework for increasing the robustness of pca-based correlation clustering algorithms. In Bertram Ludscher and Nikos Mamoulis, editors, *Scientific and Statistical Database Management*, page 18. Springer, July 2008.