IINTERNSHIP REPORT

A study on Matrix completion, associated algorithms and applications

Submitted by

GURSIMRAN SINGH

Under the Guidance of

MURUGESAN VENKATAPATHI
Assistant Professor,
SuperComputer Education and Research Canter,
INDIAN INSTITUTE OF SCIENCE, BANGALORE.

Kishore Vaigyanik Protsahan Yojana

IIT Bombay cell

Abstract

In this transcript an attempt has been made to study the problem of matrix completion, which asks whether it is possible to recover a matrix given only a subset of its entries. Under no other assumption, the problem seems impossible because each such missing entry can take any value on the real number line R [1] which make an infinite matrices on $\mathbb{R}^{n\times n}$, all equally probable candidates for original matrix. So the problem was rather an ill-posed one, only after Candes and Retch in 2008 proved mathematically that in case of nxn matrix[2], we require only $m \geq C\, n^{1.2} r\, log(n)$ entries to recover the whole matrix with high probability, given only the assumption that the matrix is low rank and possesses certain properties so called weak coherence property. Interesting point is that in practical scenarios a majority of matrices satisfy these expectations and are subject to exact recovery, even if a small number of entries are revealed. This problem is of considerable practical interest as in most of practical scenarios having large data matrices, there are only a few independent directions that can actually represent the whole matrix. So, often there is either a low rank or an approximately low rank structure lying hidden in that higher dimensional space. Some of many examples include Netflix prize problem [1] movie-user data matrix, sensor net distance matrix and quantum tomography [2].

Apart from theoretical guarantees it can also be shown that the unknown matrix X can be obtained from the partially revealed matrix by solving a unique convex optimisation problem called the nuclear norm minimisation. Nuclear norm minimisation introduced by Fazel in her Phd thesis is the tightest convex relaxation of the rank minimisation problem which is the solution to matrix completion problem. This is similar to using the popular l_1 norm as a proxy for l_0 norm in compressed sensing and medical imaging literature. The idea is that in both the cases the former is computationally more tractable than the latter. The dual of above nuclear norm minimisation can be casted into a semi definite program which can be solved by SDP solvers. However SDP solvers aren't very efficient for large problems so different research groups have been proposing more efficient, but often limited in scope, alternative algorithms for solving matrix completion problem. These include low rank matrix factorization by Srebro etal and Monteiro etal, FPCA (Fixed Point Continuation with approximation SVD) by Chen etal, SVT (Singular Value Thresholding) by Candes etal and OptSapce by Keshavan and Oh. The further study on matrix completion problem has taken two routes, one improving on theory for better and closer results the other finding and improving new algorithms which can converge fast enough and solve a broader domain of large practical problem.

[1] Complex number are also allowed, however without losing generality reals are taken for simplicity.
[2] Rectangular matrices are also allowed but square ones are simple to consider for analysis.

1. Preliminary

1.1. Singular value decomposition

Singular value decomposition is a factorization of any rectangular matrix M into three factors the orthonormal factor U, a diagonal matrix ∑ and the transpose of the orthonormal matrix V.

$$M = U\sum V^T$$

Where

$$U = \{u_1, u_2, u_3 \ldots . u_m\} \quad \textit{are the left singular vectors of MM}$$

$$V = \{v_1, v_2, v_3 \ldots . v_n\} \quad \textit{are the right singular vectors of M}$$

$$\begin{bmatrix} s_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & s_r \end{bmatrix} \quad \textit{are the r singular values of M}$$

Such that

$$Mu_i = s_i v_i$$

Satisfying these constraints

$$U^T U = I \qquad \textit{U is orthonormal}$$

$$V^T V = I \qquad \textit{V is orthonormal}$$

$$s_{ij} = 0\ ;\ if\ i \neq j \qquad \textit{i.e S is diagnol matrix}$$

In case of full singular value decomposition, the row space of V is spanned by those columns which are associated with non-zero singular values. On the other hand, the columns of V which correspond to zero singular values provide an orthonormal basis for null space. Similarly the column space is spanned by columns of U associated with non-zero singular values.

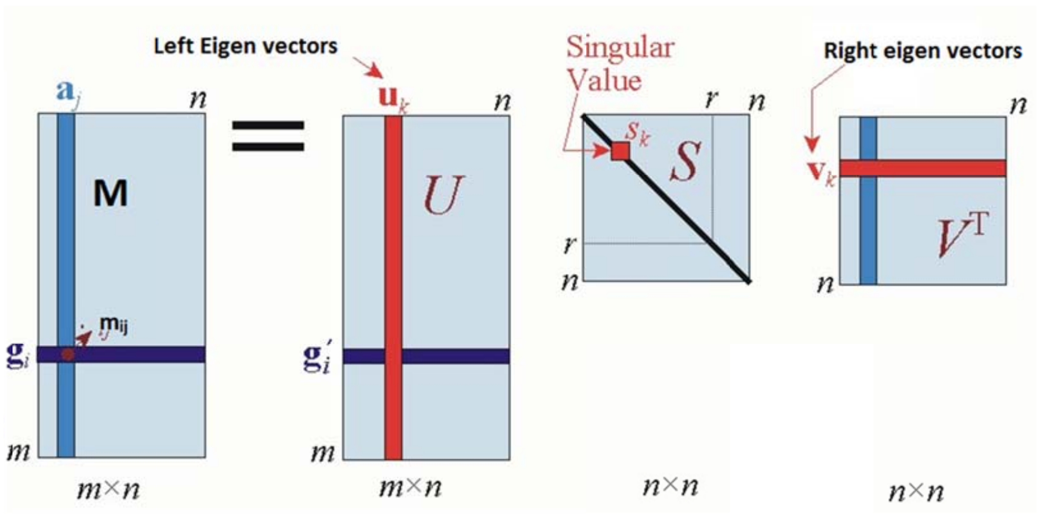


Figure 1.1 - Partial singular value decomposition

In yet another prospective, the matrix M can be thought to be linear map acting on vectors to produce other vectors in some other space. The action of M can be very easily understood and described when factorized in SVD form. [1]

- Rotation (Performed by V^T)
- Scaling (Performed by $\sum$)
- Rotation (Performed by U)

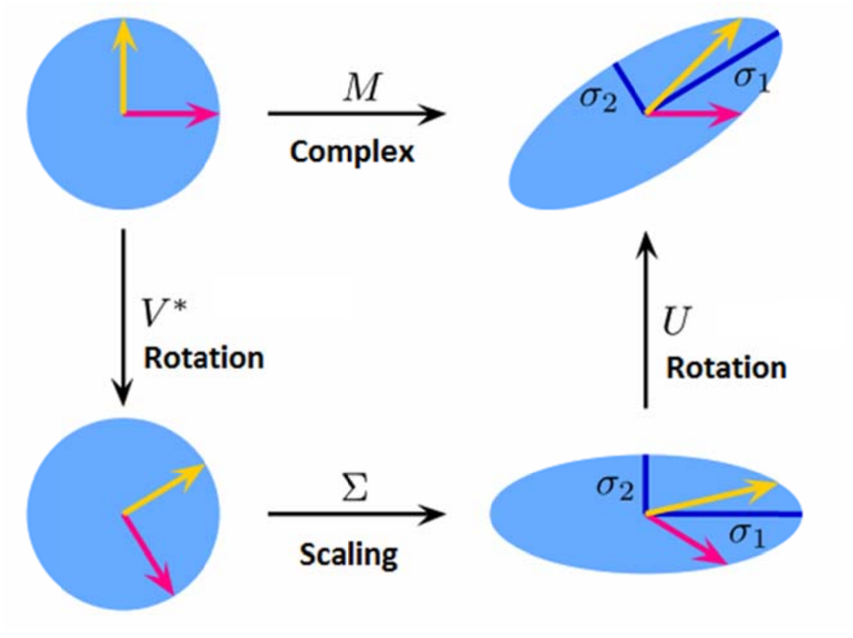


Figure 1.2 – Explaining the effect of matrix M as a linear operator

1.2. Principal component analysis

In many practical experiments the variables of the system can be unwieldy and may be deceptive which leads to greater problem in deciding the number of variables. Often experimentalists tend to measure more variables than those required to accurately define the dynamics of the system, which leads to redundancy and unclear data garbled with noise. Thus the data acquired is often high dimensional which makes it increasingly difficult to understand the underlying reality of science.

Principal component analysis is the technique to find hidden correlations in data and the corresponding directions. Thus PCA finds a new set of orthonormal basis in which the data can be represented in fewer dimensions in comparison to the naïve original basis. Often re-expressing the data in the new meaningful basis leads to noise removal from the dataset which is an added advantage. [2]

1.2.1. Basis change equation

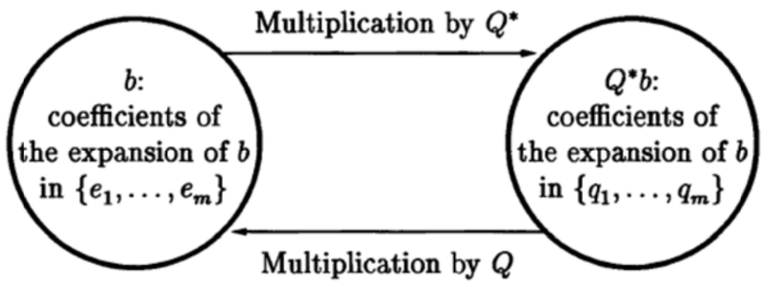

Multiplication by Q^*
b: coefficients of the expansion of b in $\{e_1, \ldots, e_m\}$
Q^*b: coefficients of the expansion of b in $\{q_1, \ldots, q_m\}$
Multiplication by Q

Figure 1.2 Basis change equation

$$b' = Q^* b$$

b' *is the new dataset in new orthonormal basis* Q

b *is the original dataset in* I

Columns of Q *contains new basis vectors*

1.2.2. Truncated SVD – The optimality property

There is a rather deep relationship between principal component analysis and singular value decomposition. In fact the easiest way of obtaining the directions with maximum variance are the eigen vectors corresponding to the largest eigen values of the covariance matrix. What is interesting here is that these eigen values and eigen vectors of the covariance matrix is directly obtained from SVD in the following way.

- The left singular vectors of M, the columns U are the eigen vectors of MM^T.
- The right singular vectors of M, the rows of V are the eigen vectors of M^TM.
- The non-zero singular values of M, diagonal elements of $\sum$ are the square root of the non-zero eigen values of MM^T, which is same as that of M^TM.

The **optimality property** is based on a theorem by Eckart and Young [3], which shows that the *most optimal* kth rank approximation of a matrix M is obtained by dropping all but the first k singular values in the SVD of the matrix M along with the corresponding columns and rows of U and V^T. Mathematically,

$$||M - M_k||_F \leq ||M - B||_F$$

Where

$$||X||_F = \sum_i \sum_j x_{ij}{}^2$$

$$M\ is\ the\ original\ matrix$$
$$M_k\ is\ the\ k^{th} rank\ approximation\ of\ the\ matrix\ M$$
$$B\ is\ any\ other\ matrix\ of\ rank\ k$$

This property makes way for principal component analysis, in which we want to reduce the rank of the data matrix to find underlying low rank structure. It shows that we cannot hope to find any other matrix B which is closer (in regard to Frobenius norm) to matrix M than M^k. So by taking the kth rank approximation by dropping singular values in SVD, we can automatically assure ourselves that we have found the best lower dimensional structure to the matrix.

1.3. Experiment

To construct data matrix G we randomly sampling from the normal distribution and scaling one dimension by a factor of 10 to study correlations and PCA. G is obtained by rotating the data by 15 degrees with the help of rotation matrix R.

```
a=10* randn(1000,1);
b= randn(1000,1);
R = [cosd(15) -sind(15); sind(15) cosd(15)]
G = R *[a,b]';

[U S V] = svd(G);   % svd of the matrix
x= G (1,:);
y= G (2,:);

plot(a,b,'.')
ylim([-40,40])
xlim ([-40 40])
hold on
plot(x,y,'g.')
```

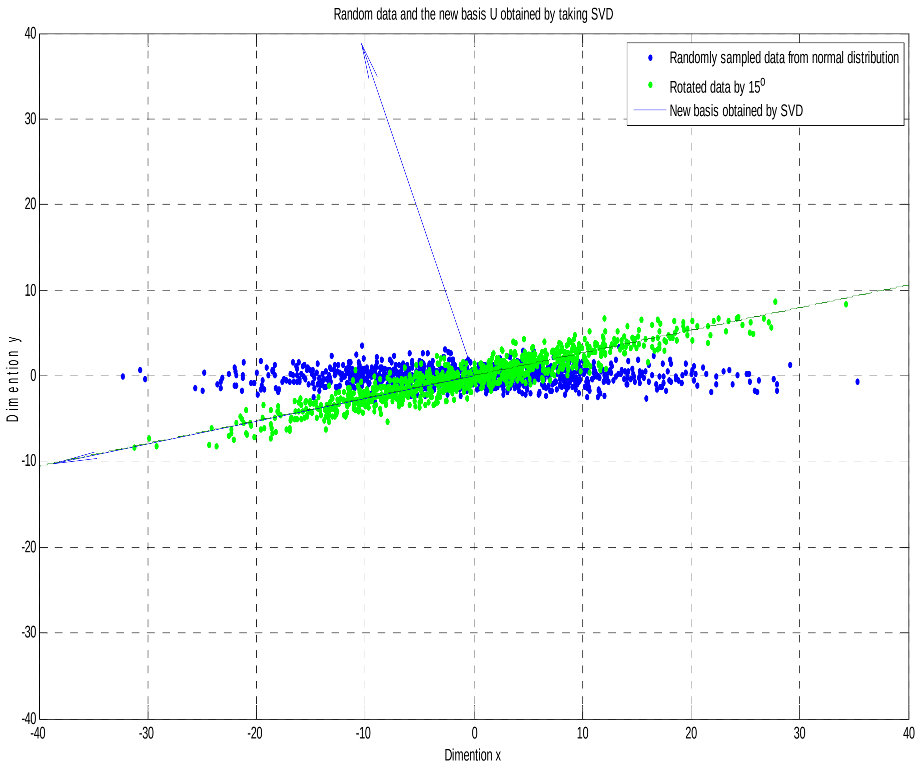


Figure 1.3 Data visualization

Noise removal by dropping columns

```
[U S V] = svd(G);   % svd of the matrix
G_svd = U * S* V';
VT = V';
G_svd_drop = U(:,1:r) * S(1:r,1:r) * VT(1:r,:);
figure
plot (G_svd_drop(1,:),G_svd_drop(2,:),'.');
ylim([-40,40])
xlim ([-40 40])
```

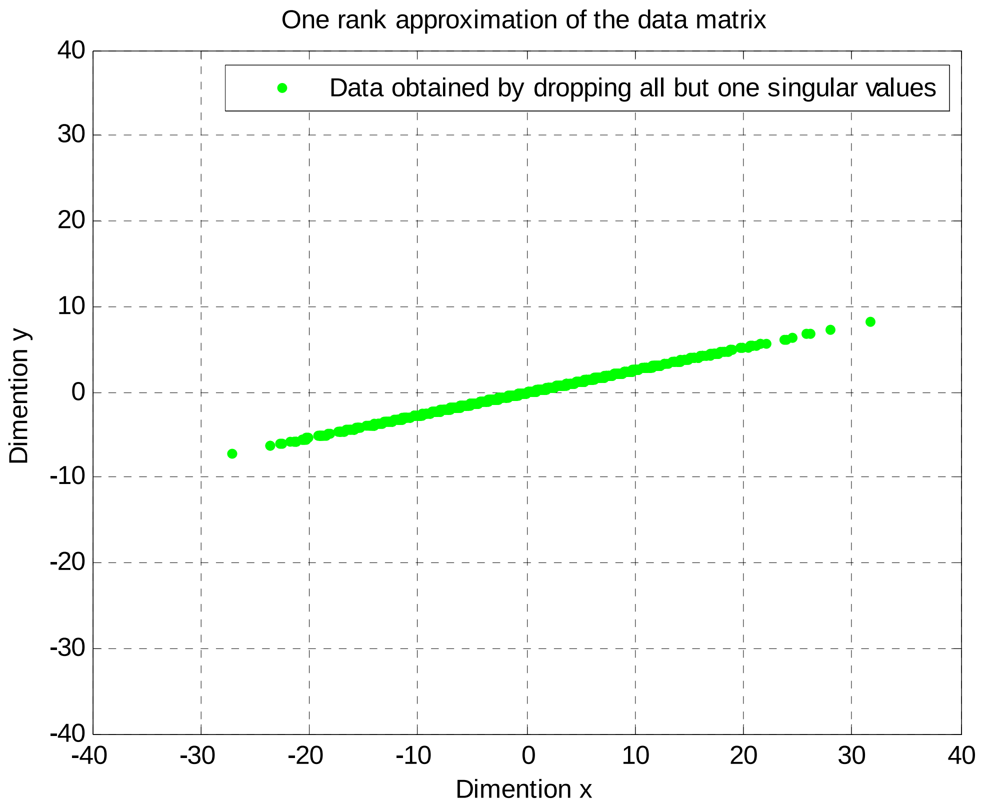


Figure 1.4 PCA by dropping in SVD

The other dimension contains only small variation of data and could be possibly noise, so it can be dropped and the underlying low-dimensional structure is obtained. Similarly we can also do so for images that are of low rank; many images are low rank compared to the dimension of the space in which they are represented. So reducing the dimension of the image using the above procedure is also a popular image compression algorithm.

2. Matrix Completion

2.1. Introduction

The Matrix completion problem asks whether it's possible and if so, under what conditions can we recover a full matrix given only a subset of its entries. The tendency to count and measure things in mathematical sciences further encourages us to investigate and quantify the error with which matrix completion is possible. Such a problem is of significant practical interest with many known applications which are increasing all the way.

$$\begin{bmatrix} \times & ? & ? & ? & \times & ? \\ ? & ? & \times & \times & ? & ? \\ \times & ? & ? & \times & ? & ? \\ ? & ? & \times & ? & ? & \times \\ \times & ? & ? & ? & ? & ? \\ ? & ? & \times & \times & ? & ? \end{bmatrix}$$

Figure 2.1 - Matrix completion

This problem had been investigated in literature before (using partly different methods and objectives), but it has gained significant importance among statisticians and computer scientists since the famous Netflix prize problem [4] which is explained in the following section.

However this was only an ill-posed problem since ever, when in 2008 Candes and Retch in the famous paper [5] proposed it as a well posed mathematical problem introducing fundamental limits on matrix recovery giving results when is matrix recovery impossible what-so-ever the algorithm be. Apart from these fundamental limits on when is matrix recovery not possible, they proved certain mathematical results that provide information on how much entries are required to complete a matrix and with how much probability and error.

In general, all must agree that matrix completion is impossible without some additional information. This is because each such missing entry can take any value on real number line R without any constraint. This makes infinite matrices equally probable candidates for original matrix. However it turns out that in spite of all entries being completely random and uncorrelated, a majority of matrices which we wish to recover in practise are known to be structured in the sense that the entries are somehow correlated. This means that in case we know all the entries of the matrix, we have extra redundant information about the matrix which we can afford to lose without losing any meaningful information about the matrix. Subsequently on demand these lost entries can later be recovered using matrix completion. In the following section we wish to have a look on some applications involving matrix which are structured and contains redundancy and following which we will formalize this notion of structured matrix.

After authors of [5] proved the matrix completion problem as a well posed problem, there are two broad topics in which research is being carried out in this area, the former studying the theoretical prospects of matrix completion and the latter investigating the experimental aspects. Candes and Retch apart from the theoretical results also proposed how we can solve the problem. They formulated a rank minimisation problem but it's a NP hard problem so they converted it onto a convex optimisation problem, nuclear norm minimisation problem.

2.2. Applications

2.2.1. Recommendation systems

Recommendation system is an intelligent system that somehow collects user's opinions/ profile and uses this collected data to present better recommended items in relation to user's interests and likings. Such a system is very common these days with all websites recommending items based on the result of such systems. eg – amazon recommending books based on recently purchased books, flipkart recommending items etc.

Basically there are two types of recommender system in literature. The first one is called the content based approach which maintains a profile of the user's personal information such as demographic/ sex/ interests etc. This information is used by an algorithm which makes decision what items to recommend. The other one, namely collaborative filtering provides recommendations based on users past behaviour and similar decisions made by other users. This system uses the similarity information to get latent features which describe users and items. This is based on the fact that only a few independent dimensions of a user decide what he likes/ dislikes and all users' uses these independent dimensions while deciding what he/she likes.

In CF based systems it is quite often that every user won't rate all items in the database, so if we arrange users as rows of a matrix and movies to be columns of the matrix it is clear that we have a matrix completion problem. However if a particular user likes movies x and y and yet another user like movies x, y and z then its high probability that the former user may like the movie z as well. This structure of the matrix makes us believe that there are some latent features that can be attributed to movies and each user decide his likings or dis-likings based on the likings and dis-likings of those latent features.

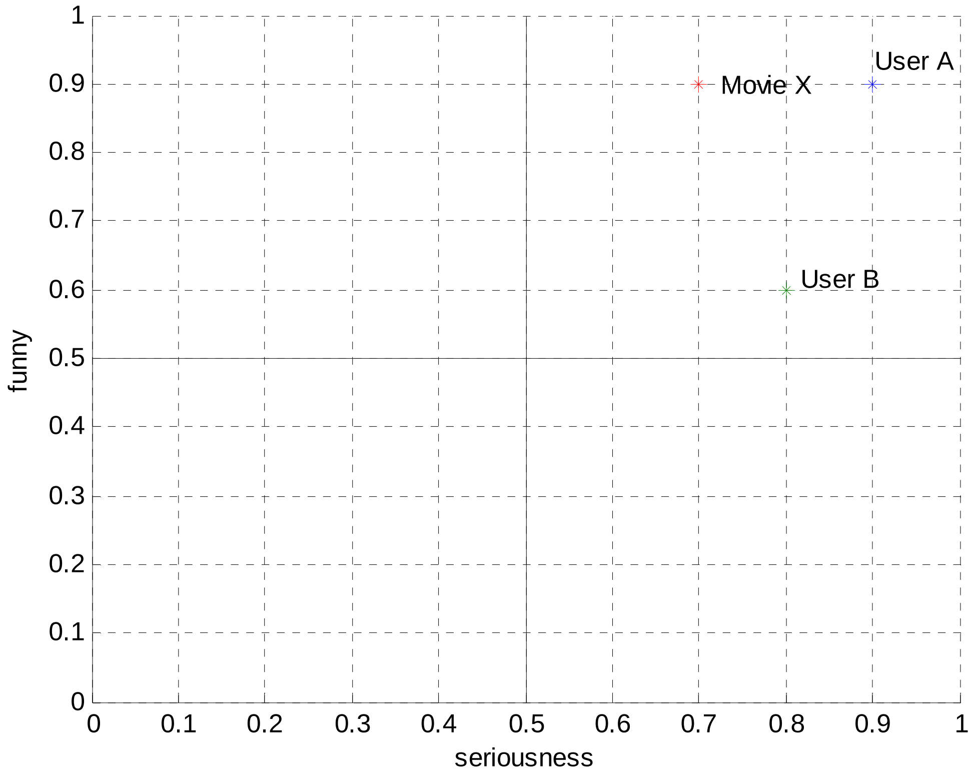


Figure 2.2 - Latent feature space

Netflix is a movie DVD rental service in USA and in 2005 it launched a competition featuring prize of $1000000 for whosoever improves their in-house recommender system, called Cinematch by 10%. The database they provided contains dyadic data [6] of 480189 users and 17770 movies and a rating in the range of {1, 5}. The values that were present were only around 100 million out of a total of 8.5 billion entries. The error was calculated using square mean root error.

2.2.2. Triangulation of sensor-net

In sensor networks where the calculating distances between each sensor is not practically feasible due to limited range of signals that each sensor emits, sensors can be thought to be arranged along the rows and columns of the matrix. This matrix contains only partial information about distances because distance between all sensors is not known. So this is again a matrix completion problem which is possible to recover with very few known entries (Singer, Biswas et al).

2.2.3. Other applications

There has been some good progress in the field of quantum tomography when the authors of [7] has shown that quantum tomography can be dealt with the help of matrix completion. Apart from that matrix completion problems are very common in computer vision as many pixels may be missing due to occulusion or tracking failures in video sequences. Further scene recovery from inferring motion of camera can be casted as a matrix completion problem which is known as structure from motion problem [8] [9]. These are some of many other applications which are constantly growing out of this relatively newer field.

2.3. Assumptions

The matrices above in all the applications are structured in the sense that all are of low rank (we recall that low rank means that all the columns of the matrix are not linearly independent). Thus there is redundancy in the matrix, which we can afford to lose. In particular, in the Netflix dataset matrix, it is commonly believed that only a few factors contribute to likings of person, in sensor net distance matrix, if the sensors are arranged in 2D space the rank of the matrix is two, otherwise if the sensors are arranged in 3D space the rank of the matrix is three. So in this particular application we need very few entries to recover because the rank is very low.

The intuition behind this idea is that in case of a $n \times n$ matrix, we have n^2 entries, however if the matrix is of low rank, we don't have the freedom to choose all vectors in R^{nxn} space, instead only certain vectors are allowed which satisfy the constraint that the matrices are of low rank. This means there is low dimensional space from which matrices are selected.

It turns out that in case of the above $n \times n$ matrix of rank r the space from which we select all vectors is $2nr - r^2$ we call this term as degrees of the freedom [5] of the matrix $n \times n$ with rank r.

Proof

Consider the SVD of the matrix

$$M = U\sum V^T$$

The set of possible singular values is a *convex cone* in R^r. Specifically, it is the cone defined by the following inequalities. This cone is an r-dimensional manifold.

$$0 < s_0,\ s_0 < s_1,\ s_2 < s_3, \ldots, s_{r-1} < s_r$$

The set of possible left singular vectors is an *orthonormal k-frame* in R^n. The space of all such frames is known as a *Stiefel manifold*, which has dimension (2n - r - 1)r/2. Similarly, the set of possible right singular vectors is also a *Stiefel manifold* of the same dimension.

$$M_1 \times M_2 \times M_3 \rightarrow R^{nxn}$$

The singular value decomposition corresponds to a **bijection** from the product of these three manifolds to a subset of the manifold consisting of all n x n matrices of rank r. The existence of such a bijection proves that the two manifolds have the same dimension.

Thus actually if we know only r(2n-r) number out of n^2, we can hope to complete the whole matrix. The next major question is that whether any set of these r(2n-r) number is sufficient to recover the matrix or whether only some combinations work?

2.4. Mathematical model

Let U and V be two sets of cardinality m and n such that:

$$U = \{u_1, u_2, u_3, \ldots . u_m\}$$
$$V = \{v_1, v_2, v_3, \ldots \ldots v_n\}$$

There exist an injection, which defines a matrix such that

$$\gamma : U \times V \rightarrow R$$

$$X(i,j) = o \quad ; o \epsilon R$$

Now consider an orthogonal projection $P_\Omega : R^{nxn} \rightarrow R^{nxn}$ onto the subspace of matrices which vanishes outside of set Ω ($\Omega \subseteq \{m\} X \{n\}$) such that

$$Y_{ij} = \begin{cases} X_{ij} & (i,j \in \Omega) \\ 0 & otherwise \end{cases}$$

We can hope to recover the matrix M given only the partial matrix $P_\Omega(M)$, if only one low rank matrix is consistent with data in P_Ω(M). If only one matrix of rank less or equal to r, fits the data, then we can hope to recover the matrix by solving the optimisation problem as under

$$\begin{array}{cc} \min & rank(X) \\ subject\ to & P_\Omega(X) = P_\Omega(M) \end{array}$$
$$Here\ X\ is\ the\ variable\ matrix\ with\ some\ iniitial\ value\ in\ algorithm$$

However the problem above is NP-hard and grows doubly exponentially in time because of its combinatorial nature [10] so recovery using the optimisation problem above is not practical. So the tightest convex envelope to this problem is nuclear norm minimisation problem which was introduced by fazel in her phd thesis [11]. Futher in a paper in 2003 they proved that it is a semidefinate program [12]

$$\begin{array}{cc} minimize & \|X\|_* \\ subject\ to & P_\Omega(X) = P_\Omega(M) \end{array}$$
$$Here\ X\ is\ the\ variable\ matrix\ with\ some\ initial\ value\ in\ algorithm$$

where $\|X\|_*$ is the nuclear norm of the matrix M defined as below

$$\|X\|_* := \sum_i s_i$$

2.4.1. Which samples of r(2n-r) entries?

In this section we would like to address the question left unanswered whether all combinations of r(2n-r) entries are sufficient to for the rank minimisation and corresponding nuclear norm minimisation problem to converge and give solution. We note the following points in this regard.

- If a complete row or column remains un-sampled, there is no hope for matrix recovery because you cannot guess what the row may be out of infinite many options.

$$M = \begin{array}{ccc} 1 & 3 & 2 \\ 2 & 6 & 4 \end{array}$$

- In case of matrices which are of the following form

$$M = e_1 e_n^* = \begin{bmatrix} 0 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix}$$

$$M = e_1 x^* = \begin{bmatrix} x_1 & \cdots & x_n \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Figure 2.3 - Matrices hard to recover

In the former case if 1 at the top-right corner remains un-sampled there is no hope and in the latter case even if one entry from row1 remains un-sampled again you can't hope anything positive out of the convex optimisation problem. So the conclusion is that apart from the fact that there are not all combinations that work, there are some matrices which are more favourable than others for recovery. We will formalize this notion using coherence conditions in the subsequent sections.

2.4.1.1. Coupon collector's effect for fundamental recovery

The fundamental limit on matrix recovery is defined by the degrees of freedom. If the entries sampled

$$m < r(2n - r)$$
$$alternatively, \quad m < \Theta(nr)$$

there is no hope for recovery whatsoever is algorithm but we wish to improve this fundamental limit so that when we sample entries no row or column remains un-sampled. For that we have the revised result as follows.

$$m < \Theta(nr\log(n))$$

The extra log factor in the fundamental limit of entries to be sampled is due to coupon collector's effect, which is based on the idea that you much not leave even one row or column of the matrix un-sampled. Coupons collector effect can be stated as '*Given n coupons, how many coupons do you expect you need to draw with replacement before having drawn each* *coupon at least once? The mathematical analysis of the problem reveals that the expected number of trials needed grows as* $\Theta(n\,log(n)$*'.* [13]

2.4.2. Theorems for matrix recovery

2.4.2.1. Candes and Retch, 2008 [5]

Let M be $n_1 \times n_2$ matrix of rank r sampled from the random orthogonal model, using $n = \max(n_1,n_2)$. Suppose we observe m entries of M with locations sampled uniformly at random. Then there exist numerical constants C and c such that if

$$m \geq C\, n^{\frac{5}{4}} r\, log(n) \quad sometimes\, \frac{6}{5}\ instead\ of\, \frac{5}{4}$$

The minimizer to the nuclear norm minimization problem is unique and equal to M with probability atleast $1\text{-}cn^{-3}$ which means semidefinite program recovers all the entries of M with no error.

2.4.2.2. Candes and Tao, 2009 [14]

Let $M \in R^{n1 \times n2}$ be a matrix of rank r sampled from the random orthogonal model, obeying a property called strong incoherence property [14] using $n = \max(n_1,n_2)$. Suppose we observe m entries of M with locations sampled uniformly at random. Then there exist a numerical constant C such that if

$$m \geq C\, \mu^2\, nr\, log(n)^6$$

$$m \geq C\, \mu^4\, n\, log(n)^2 \qquad sometimes\ when\ r = O(1)$$

The minimizer to the nuclear norm minimization problem is unique and equal to M with probability atleast $1\text{-}cn^{-3}$ which means semidefinite program recovers all the entries of M with no error.

2.4.3. Coherence conditions

We observe in section 2.4.1 that there are some matrices which are more favourable than others, that is to say that there exist a property of the matrix involved which makes some of them favourable and some unfavourable in the sense that more number of combinations qualify for successful recovery and hence the probability of recovery is more. We denote such unknown property as

$$m \propto \Phi_X(M)$$

In other words singular vectors should be sufficiently spread in the sense that they must be uncorrelated with the standard basis so that we may require fewer samples for matrix recovery. Both the left and right singular vectors must be uncorrelated with the standard basis.

$$\mu(U) \equiv \frac{n}{r} \max_{1 \le i \le n} \|\boldsymbol{P}_U \boldsymbol{e}_i\|^2.$$

2.5. Matrix completion with noise

A major point of concern of the above analysis is that like the Netflix data matrix, many practical applications may contain some outliers or even filled with noise. If the noise is assumed to be Gaussian we can change the results above slightly to accommodate the extra noise term and deduce the following results. [15]

$$Y_{ij} = M_{ij} + Z_{ij}, \quad (i,j) \in \Omega,$$

$$\mathcal{P}_\Omega(Y) = \mathcal{P}_\Omega(M) + \mathcal{P}_\Omega(Z)$$

$$\begin{array}{ll} \text{minimize} & \|X\|_* \\ \text{subject to} & \|\mathcal{P}_\Omega(X - Y)\|_F \le \delta \end{array}$$

The problem above can be solved by casting into a semi-definite program and can be solved with available SDP solvers.

2.6. Cardinality minimisation

Another problem known as cardinality minimisation, popular in compressed sensing literature needs a small mention here.

$$\begin{array}{l} \min \|x\|_0 \\ s.t\; Ax = b \end{array}$$

This problem has deep relations to matrix completion in the context that it is also NP hard similar to rank minimisation problem in matrix completion. Analogously we replace the cardinality minimisation problem by its corresponding convex envelope which is l_1 norm here. So instead we solve for the following

$$\min \|x\|_1$$
$$s.t\ Ax = b$$

2.7. Algorithms

Because rank minimisation problem is NP hard in general due to combinational nature of the rank function, nuclear norm minimisation problem can be casted as a semi definite program as under and used for matrix completion:

$$\begin{aligned} &\min \|X\|_* \\ &\text{s.t. } \mathscr{A}(X) = b. \end{aligned}$$

If the matrix is contaminated by noise,

$$\begin{aligned} &\min \|X\|_* \\ &\text{s.t. } \|\mathscr{A}(X) - b\|_2 \leq \theta \end{aligned}$$

Or its Lagrangian version, which can be solved by FPC/ FPCA algorithm [16]

$$\min \mu \|X\|_* + \frac{1}{2}\|\mathscr{A}(X) - b\|_2^2,$$

The semi definite program formulation of the equation above is

$$\begin{aligned} &\min_{X, W_1, W_2} \tfrac{1}{2}(\operatorname{Tr}(W_1) + \operatorname{Tr}(W_2)) \\ &\text{s.t. } \begin{bmatrix} W_1 & X \\ X^\top & W_2 \end{bmatrix} \succeq 0 \\ &\mathscr{A}(X) = b, \end{aligned}$$ [10]

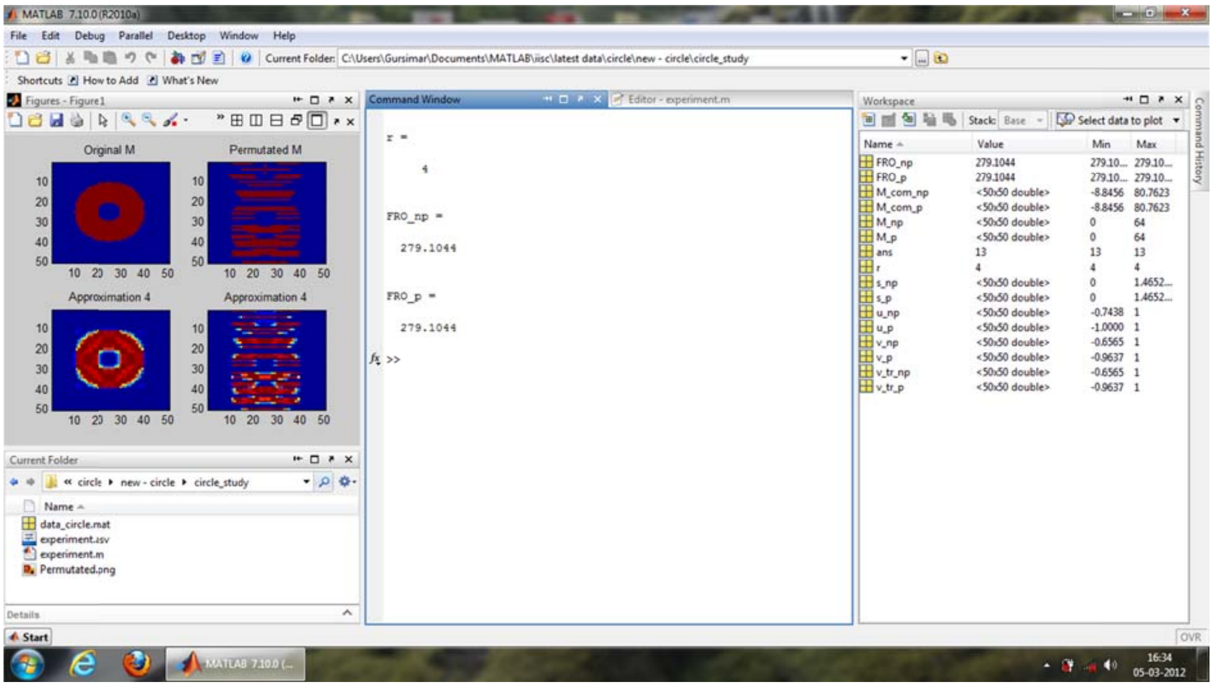
Although we can solve the SDP formulation above easily using various SDP solvers like SeDuMi and SDPT3, the problem is that these solvers aren't computationally efficient working in O(n^3) time. So alternatively there are some other algorithms in literature which are more efficient than SDP solvers but often less wider in range of scope of problems.

Another algorithm by Candes etal is Singular value threshloding which is inspired by the work of linearized Bregman algorithms for compressed sensing. Although SVT is quite efficient for large matrices but it only works for matrices which are sufficiently low rank. However according to [10] FPCA is much more efficient and faster in many domains as compared to SVT. Another algorithm optSpace [17] by another group seems to outperform both of the above with a different perspective on matrix completion [18] as well

.

3. Experiments

The section contains some experiments conducted in MATLAB to verify and study matrix completion problem along with some algorithms existing in literature. A very simple and novel matrix with patterns of 0s and 64s is chosen that can be visualized easily using the *imagesc* function of the matlab. However it tunes out that many of algorithms like FPCA and SVT deisgned for specialized problems doesn't work well with such patterns so in order for comparison we have used the real datasets. However for studying we have used the simple matrix as described in figure in the following section.



```
r =

     4

FRO_np =

  279.1044

FRO_p =

  279.1044

>>
```

Name	Value	Min	Max
FRO_np	279.1044	279.10...	279.10...
FRO_p	279.1044	279.10...	279.10...
M_com_np	<50x50 double>	-8.8456	80.7623
M_com_p	<50x50 double>	-8.8456	80.7623
M_np	<50x50 double>	0	64
M_p	<50x50 double>	0	64
ans	13	13	13
r	4	4	4
s_np	<50x50 double>	0	1.4652...
s_p	<50x50 double>	0	1.4652...
u_np	<50x50 double>	-0.7438	1
u_p	<50x50 double>	-1.0000	1
v_np	<50x50 double>	-0.6565	1
v_p	<50x50 double>	-0.9637	1
v_tr_np	<50x50 double>	-0.6565	1
v_tr_p	<50x50 double>	-0.9637	1

Name
data_circle.mat
experiment.asv
experiment.m
Permutated.png

Figure 3.1 - Matlab view of experiments

All the experiments to study verify and compare matrix completion in the following sections is conducted on a machine with the flowing configuration

- Intel i5 processor with 4 real cores and 2 virtual cores through hyperthreading running at 2.53GHz.
- 8GB RAM with 8 GBps memory-processor bandwidth.

3.1. Matrix completion by factorization

Matrix completion by factorization is not a very good algorithm in the sense that it is a heuristic and proper mathematical analysis is not available very clearly in literature. However this has been applied by Simon Funk successfully in Netflix prize, it also works for a variety of datasets and do so quite fast. This is a variant of simon-funk [19] type heuristic with some modifications in the algorithm.

$$M = UV$$

The user-movie dataset matrix M can be thought to be made up of two latent feature matrices U and V which are sampled from normal distribution for initial value. These matrices are updated in the loop and with each iteration it tries to get closer to original matrix.

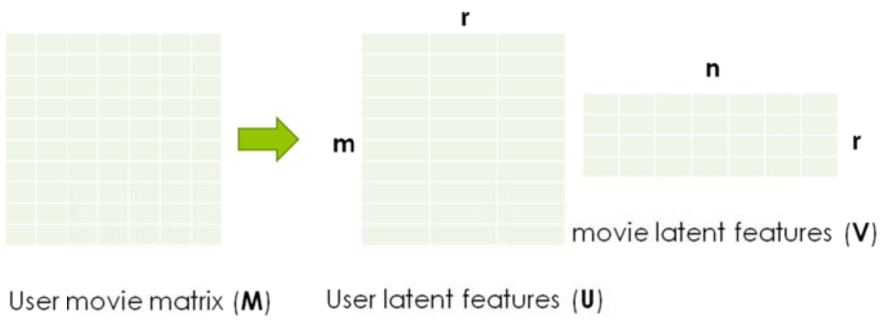


Figure 3.2 – Data matrix as two latent feature matrices

The original matrix M is taken to be a binary matrix containing 0 and 64 as depicted in the figure for simplicity and studying the matrix completion process.

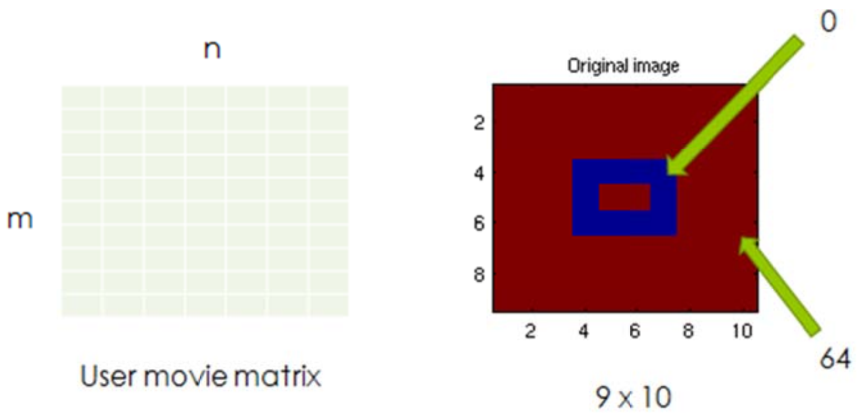


Figure 3.3 - Original matrix M

To study the matrix completion we now randomly hide some entries from the matrix and these missing locations are itself chosen from a normal distribution of internes ranging from 1 to n*m without replacement so that values does not repeat. Hence by following such a procedure we can assure ourselves that the given number of entries is missing without any doubt.

Code

```
k = randperm(a(1)*a(2));
k = k (1:l);

M_missing = M;
M_missing(k)=32;
```

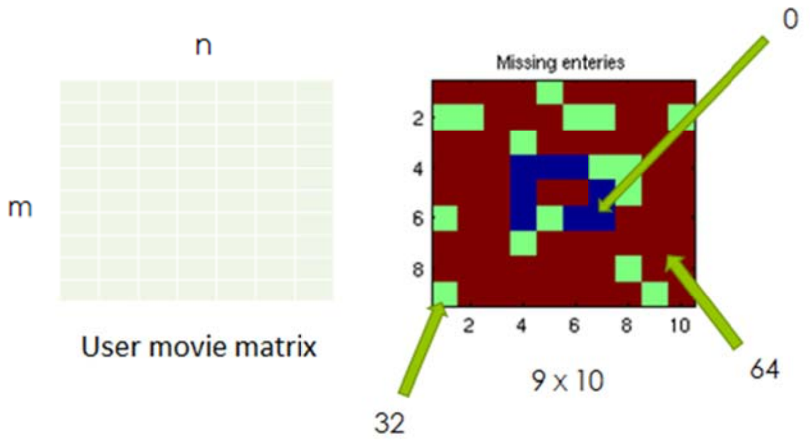


Figure 3.4 - Matrix with missing enteries M_missing

The values shown in blue are 0 and that shown in red are 64 as in the figure above, however the entries in green are missing and their information will not be used in the algorithm which predicting the original matrix. We have used a value of 32 just for illustration purposes and to depict it clearly, The value assigned to missing numbers does not matter as it is not to be used in the algorithm which computing the original matrix. The assumption that it is of low rank holds good here. The goal of our algorithm is that we wish to recover the original matrix from this matrix with missing enters.

3.1.1. Mathematical model

The goal of the algorithm is to minimize the frobinius norm between the two matrices X and M.

$$\min \left\| X_{ij} - M_{ij} \right\|_F$$

$$s.t\ (i,j) \in \Omega$$

Or equivalently

$$\min \mathrm{e} = \sum \left\| x_{ij} - m_{ij} \right\|^2$$

We update matrix x so that our error term e gets minimised after each iteration, a very simple algorithm for this is introduced by Simon Funk here [19]. The algorithm here is gradient descent, so calculate the delta term by differentiating the error term as under

Decompose the matrix into two latent factors as described above

$$R \approx SV = \hat{R}$$

$$\hat{e}_{MF}(u_i, m_j) = \underline{s}_i^T \underline{v}_j = \sum_{k=1}^{f} s_{ik} v_{kj}$$

This is the error with respect to each entry between the two matrices.

$$\varepsilon_{ij}^2 = (r_{ij} - \hat{r}_{ij})^2 = \left(r_{ij} - \sum_{k=1}^{f} s_{ik} v_{kj} \right)^2$$

We differentiate this error with respect to corresponding row and column (latent features here) to get update rules for gradient descent and add a learning rate α.

$$s'_{ik} = s_{ik} + \alpha \frac{\partial \varepsilon_{ij}^2}{\partial s_{ik}} = s_{ik} + 2\alpha \varepsilon_{ij} v_{kj}$$

$$v'_{kj} = v_{kj} + \alpha \frac{\partial \varepsilon_{ij}^2}{\partial v_{kj}} = v_{kj} + 2\alpha \varepsilon_{ij} s_{ik}$$

3.1.2. Algorithm

With the values of U and V we calculate the values of M' after each iteration and compare it with M. The algorithm updates the rows and columns of U and V so that the norm of the residual is minimised. We found that such a direction is given by steepest descent algorithm.

```
• Initialize the matrices U and V or S and R.
• For k = 1 to max_iter
    o Loop
        ▪ Loop for all $(i, j) \in \Omega$
            • Compute error
            • Update features
    o Recompute and print error term
• end
```

Here is a snippet of MATLAB implementation of the algorithm above.

```
while(FRO_NORM_DIFF_MISSING>fro_th)
    FRO_NORM_DIFF_MISSING = 0;  %resets the value
    for i = 1:a(1)
        for j = 1:a(2)
          %pause
          if M_missing(i,j) ~= 32 % assume 32 are missing enteries
          so we can't calculate errors
          % make corrections
          err = M_missing(i,j) - (u_approx_missing(i,:) *
          v_approx_missing(:,j));
          c = u_approx_missing(i,:);
          u_approx_missing(i,:) = u_approx_missing(i,:) +
          (err*lrate).* v_approx_missing(:,j)';
          v_approx_missing(:,j) = v_approx_missing(:,j) +
          (err*lrate).* c';
          FRO_NORM_DIFF_MISSING = FRO_NORM_DIFF_MISSING + err*err;
        end
     end
   end
   k=k+1; % this will calculate iterations

   M_approx_missing =u_approx_missing * v_approx_missing;
   image (M_approx_missing);
end

FRO (f,l) = FRO_NORM_DIFF_MISSING;
RMSE_missing(f,l) = sum(sum((M_approx_missing-M).^2  ))/l;
```

3.1.3. Fundamental limits for recovery

How the matrix X recovers from M is also dependent on rank r of U and V matrices which is user defined. This is different from rank R of the matrix M. In the subsequent sections we have conducted experiments for r>R because our interest was to investigate whether some optimal r exist for which matrix recovery is more accurate or efficient. If r<R it is obvious that we can't recover the matrix exactly, so it is of less practical interest.

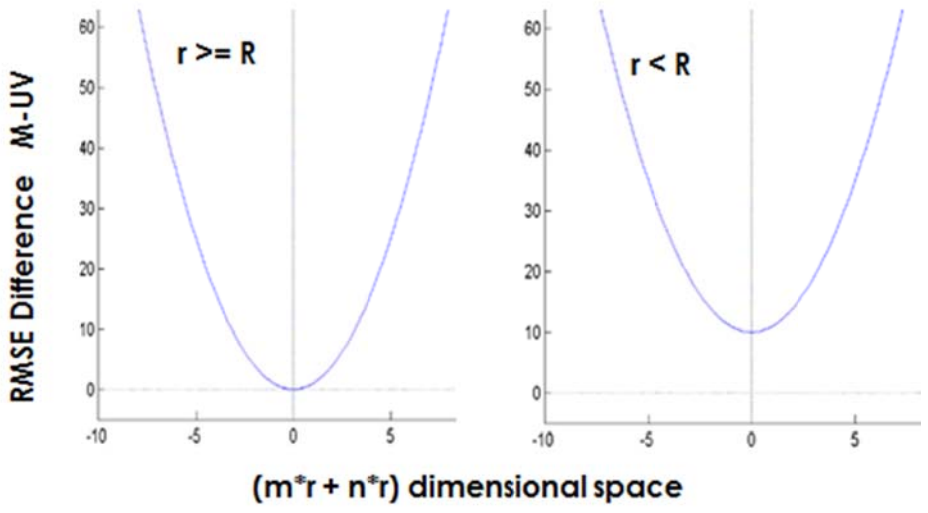


Figure 3.5 - RMSE difference curves with r and R

3.1.4. Results

3.1.4.1. Rectangle matrix M

This section contains numerical simulations performed on the matrix M of Figure 4.3. During initialization we have to randomly sample matrices U and V and the corresponding, M', obtained by multiplying them is displayed below.

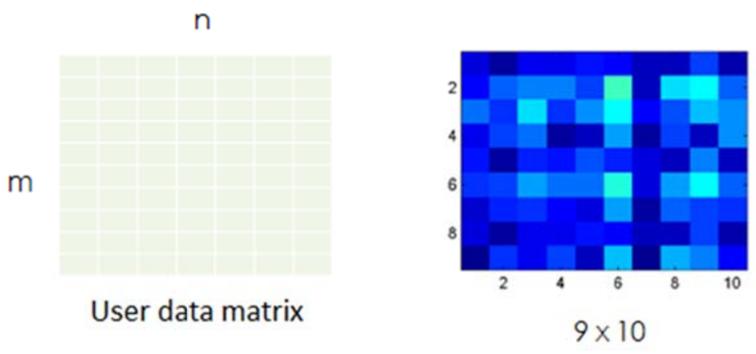


Figure 3.6 - Matrix M' at the start of algorithm

Following which the algorithm in section 4.1.2 is expected to converge at global minimum. The following illustration, developed in MATLAB shows that the optimisation function is convex and how it converges to global minimum.

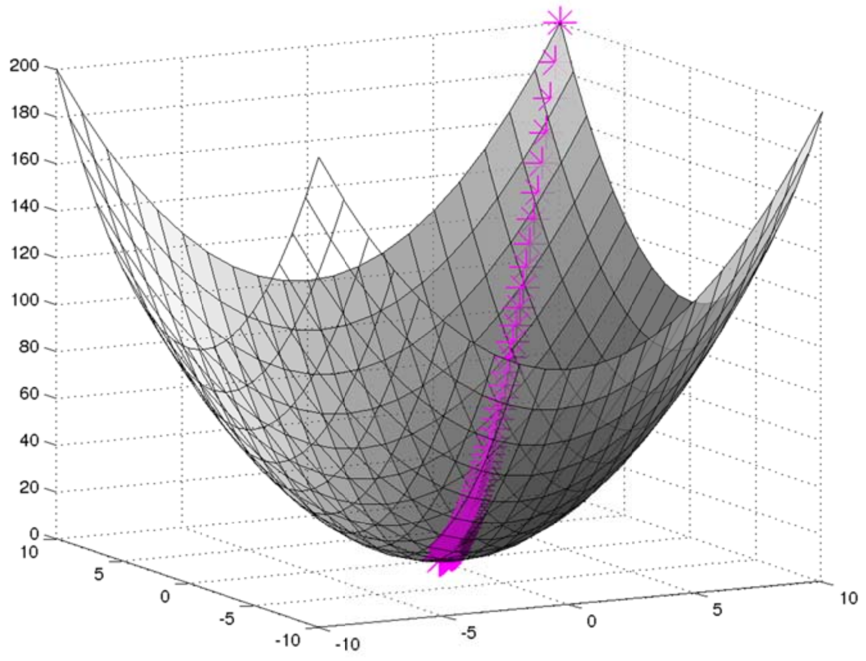


Figure 3.7 - Gradient descent in two dimentions

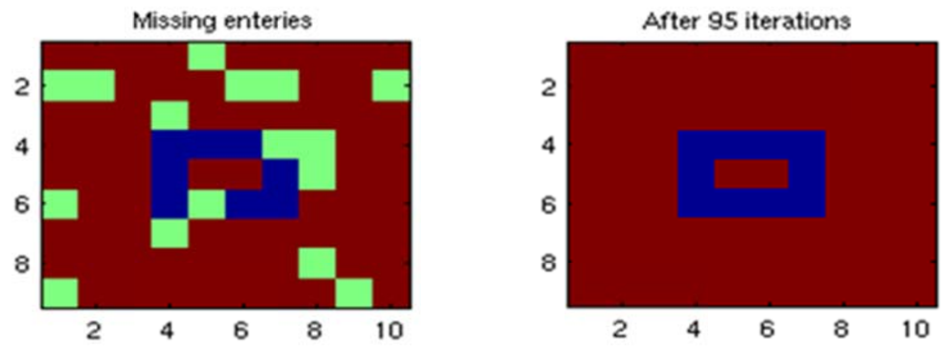


Figure 3.8 - Matrix obtained after completion of algorithm

3.1.4.1.1. Monti carlo analysis

These curves are obtained by monti-carlo simulations ie by running the same program multiple times and averaging the results. Results comply with theory, if percentage sparseness increases, the error RMSE difference between the two matrices increases. Thus matrix recovery becomes harder.

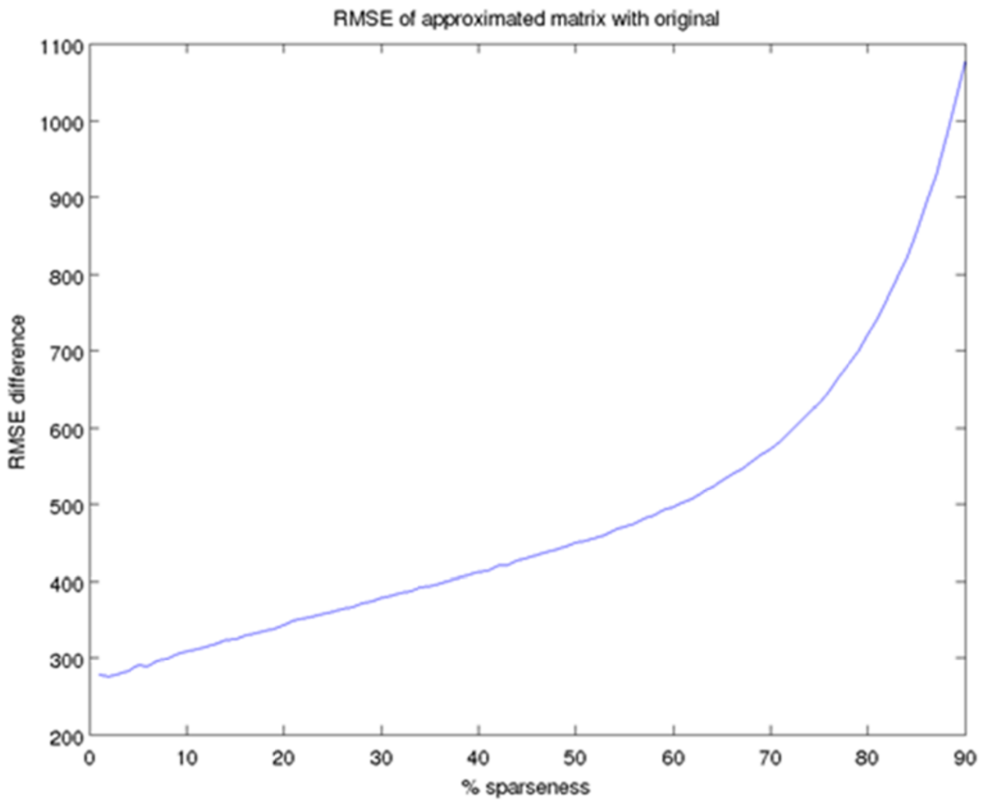
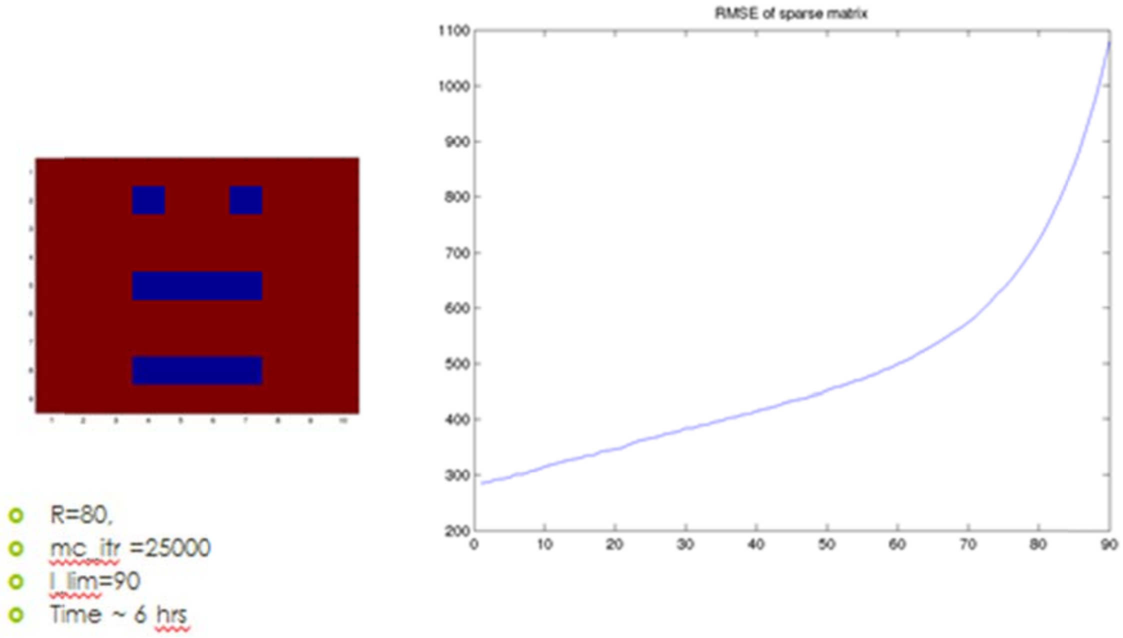


Figure 3.9 - RMSE curve with % sparseness for rectangular matrix

An experiment conducted with **permuted matrix** shows that, even if we permute the matrix we get the same RMSE curve.



- R=80,
- mc_itr =25000
- l_lim=90
- Time ~ 6 hrs

Figure 3.10 - RMSE curve of rectangle permuted matrix

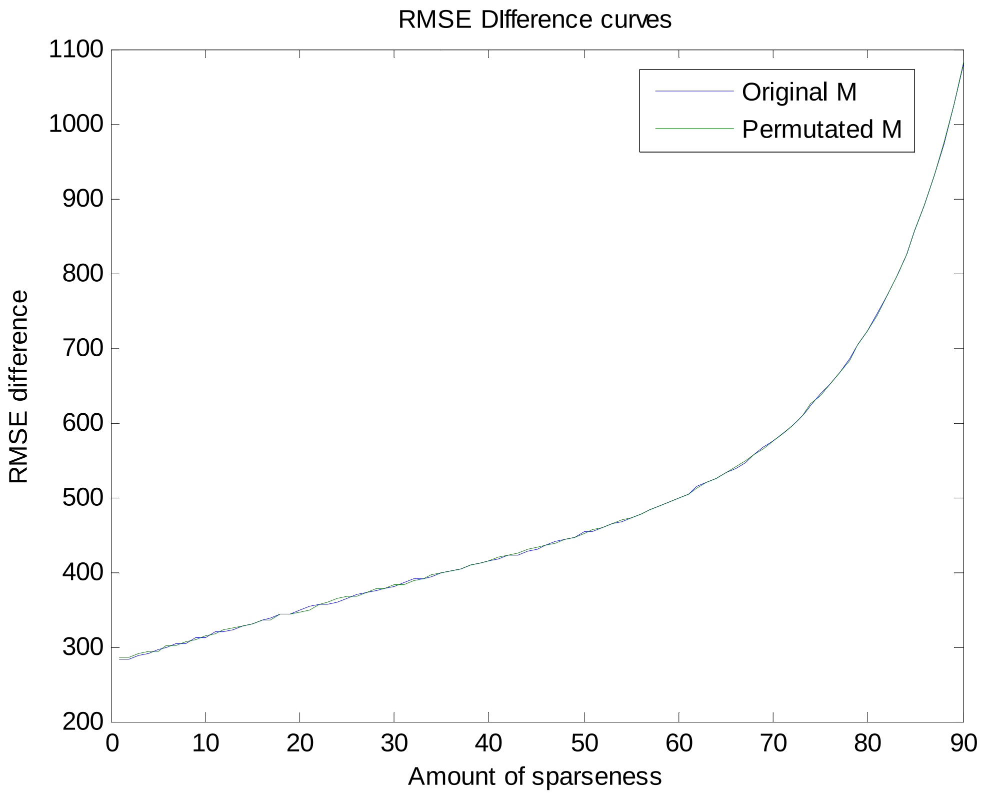


Figure 3.11 - Closer look on RMSE curves

3.1.4.1.2. RMSE curves with r>R

RMSE curves taken with different upper bound on rank of U and V matrix r (>R) indicates that lower rank facilitates better recovery with this algorithm. However we know due to theory that if r<R error will increase it again as there is a fundamental barrier to matrix recovery. So there should be some optimal rank for which matrix recovery is most accurate, however we observer in subsequent section that we have to spent more computational power to get that result. So time factor increases.

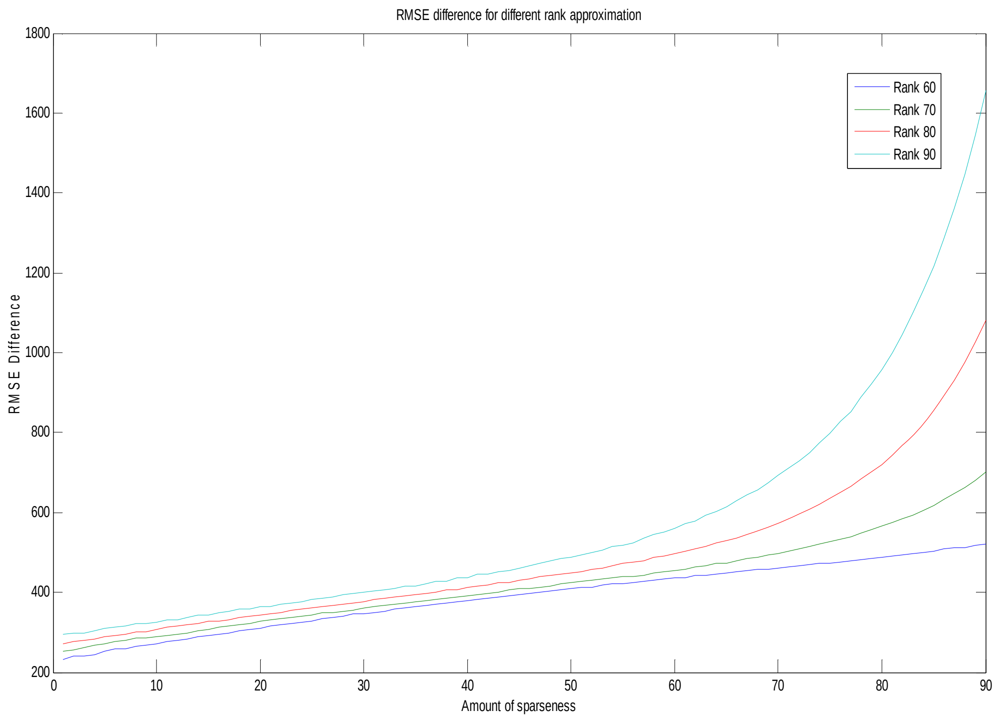


Figure 3.12 - RMSE curves r>R

3.1.4.2. Circle

Generation of circle matrix

```
% To construct image
imageSizeX = 25;
imageSizeY = 25;
[columnsInImage rowsInImage] = meshgrid(1:imageSizeX, 1:imageSizeY);
% Next create the circle in the image.
centerX = 12;
centerY = 12;
radius = 5;
width = 5;
M = 64.*( (rowsInImage - centerY).^2 + (columnsInImage - centerX).^2 <=
radius.^2 & (rowsInImage - centerY).^2 + (columnsInImage - centerX).^2
>=(radius-width).^2);
```

To save time we obtain curves with an interval of 50, as matrix recovery becomes hard when we have middle rank matrices with large size. Another thing worth noting is that we have same pattern of curves as in case of rectangular matrix.

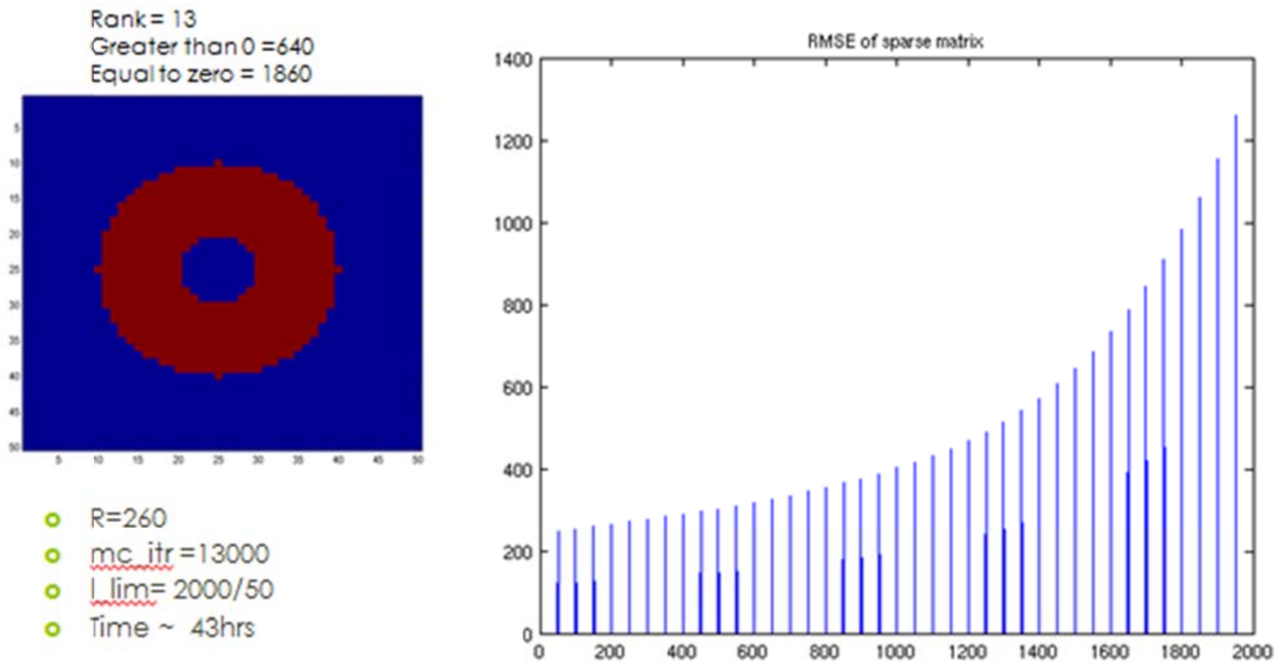


Figure 3.13 - RMSE curve of circle

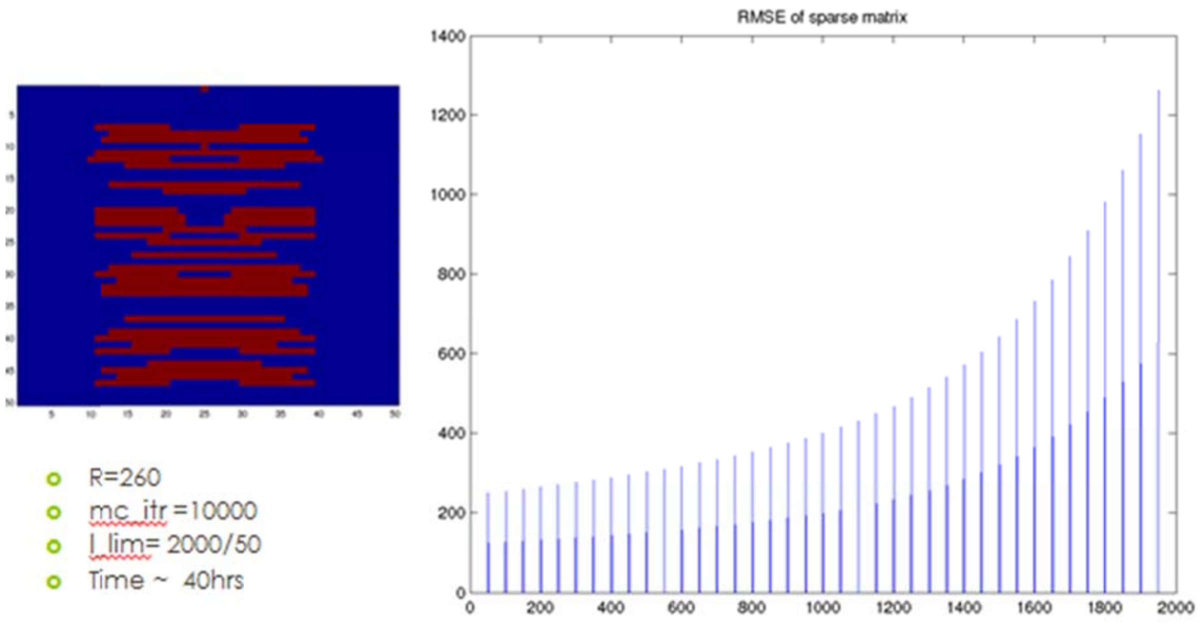


Figure 3.14 - RMSE curve of permuted circle

3.1.5. Testing

3.1.5.1. Random number generator

This figure indicates a result for simulation obtained by taking a large number of sampling and then averaging before plotting them in a histogram (line here). A approximately horizontal line indicates that distribution from which we are sampling is almost uniform.

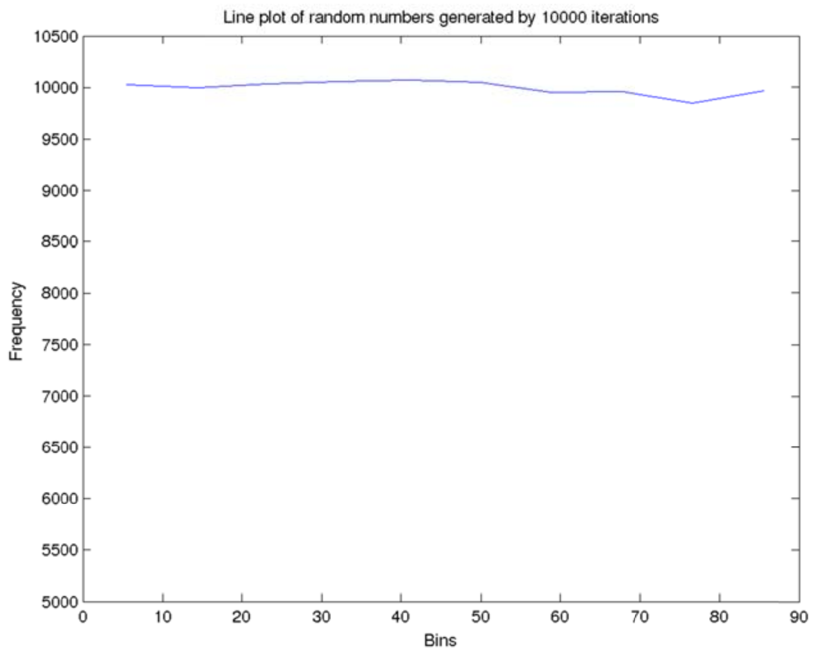


Figure 3.15 - Random number generator distribution testing

3.1.5.2. Convergence time of the algorithm vs rank.

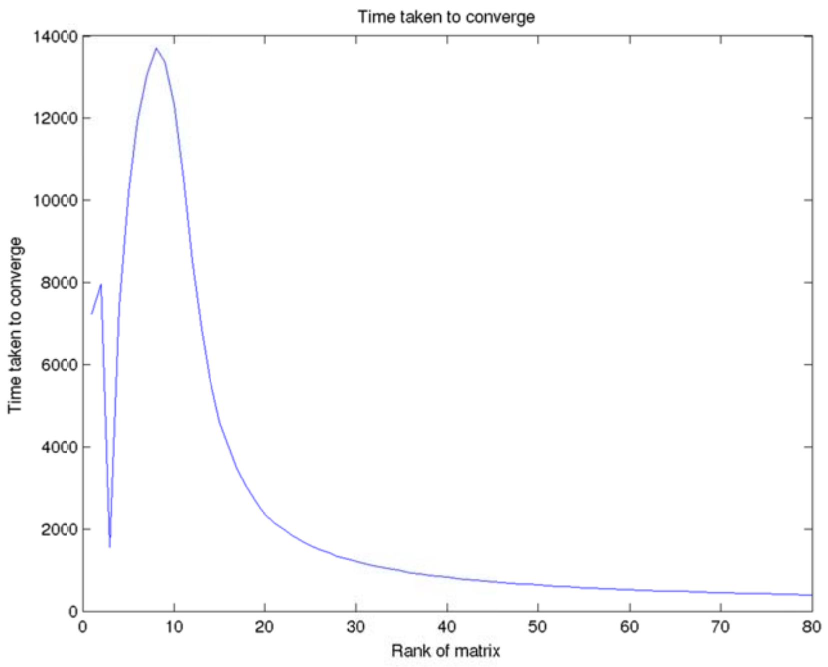


Figure 3.16 - Time vs r curve for RMSE

3.1.5.3. *Working of single iteration*

Single iteration testing while debugging indicates expected pattern. For a matrix with no missing entries we have RMSE of almost 0 while for the matix with 20% sparseness we have spikes as expected. The reason of spikes is that because not all combinations of entries will result in same accuracy of matrix recovery as explained in section 3.

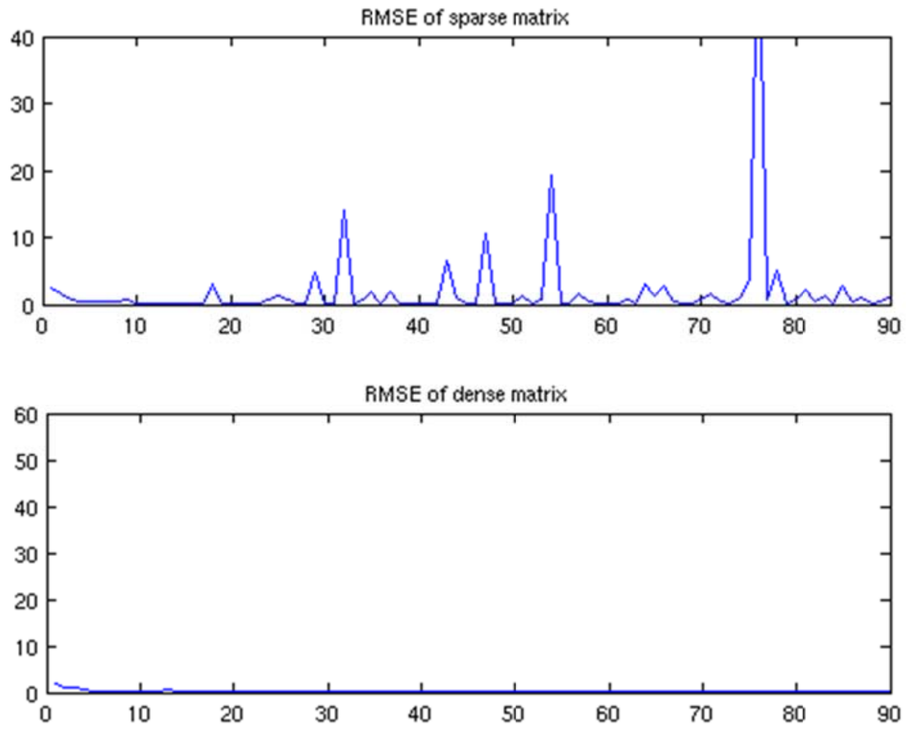


Figure 3.17 - Single iteration testing

4. Conclusion

4.1. Conclusion of experiments

In the earlier experiments we explore how well a matrix can be recovered if the rank is low (rectangle) or medium rank (circle). By observing the visual patterns, which are chosen intentionally to easily compare and realize the output and performance of the algorithm. In the subsequent experiments monti-carlo simulations are performed to invent any possible patterns of recoverability if we limit the rank of the factored matrices. As evident from figure 4.5 we can’t recover matrix what-so-ever the algorithm if we add a fundamental barrier in our algorithm by choosing the case r<R. This was of little practical interest, so the case of r>R has been investigated in the experiments. Figure 4.12 suggests that with less rank we have better scope for accuracy as indicated by low RMSE difference. Thus this gives us some evidence that optimal rank exist. However Figure 4.16 indicates that more time is required to get more accuracy, so there is a trade-off in the algorithm which makes recovery hard with high accuracy. For high sparseness, we observe that assumption laid by Candes etal does not hold, so recovery is not possible very accurately as indicated by high RMSE difference in curves.

4.2. Scope for future work

Regarding the experiments conducted, as indicated in section 4.1.3 we have dealt with case of r>R while less importance was paid to case of r<R primarily due to lack of interest and time. However, it will be interesting to explore the direction and test whether theoretical prediction holds or not. Further we have some unusual pattern of time in figure 4.16 which an interesting reader can explore.

Furthermore the field of matrix completion is exploding with a lot of researchers coming out with innovative idea both in theory, algorithms and applications. Ma etal has complied a list of reference and applications at [20]. The theory of matrix completion has progressed way beyond to include ideas like robust principal component analysis [21] which is an extension to matrix recovery from noise. We have some new advancements in the field of image processing, background extraction etc using TILT [22]. Many new advanced applications collected by Ma etal can be found here [23].

Bibliography

[1] K. Baker, “Singular Value Decomposition Tutorial,” 2005.

[2] J. Shlens, “A Tutorial on Principal Component Analysis,” 2003.

[3] G. Y. Carl Eckart, “The approximation of one matrix by another of lower rank,” *Phychometrika,* 1936.

[4] “Netflix,” [Online]. Available: www.netflix.org.

[5] E. J. Candes and B. Recth, “Exact Matrix completion via Convex optimisation,” 2008.

[6] A. Menon and C. Elkan, “Predicting labels for dyadic data,” 2010.

[7] D. Gross, “Quantum state tomography via compressed sensing,” 2009.

[8] P. Chen and D. Suter, “Recovering the missing components in a large noisy low rank matrix,” 2004.

[9] C. Tomasi and T. Kanade, “Shape and motion from image streams under orthography: a factorisation method,” 1992.

[10] S. Ma, D. Goldfarb and L. Chen, “Fixed point and Bregman iterative methods for matrix rank minimisation,” 2009.

[11] M. Fazel, “Matric rank minimisation with applications PHD thesis,” *Stanford UNiversity,* 2002.

[12] M. Fazel, H. Hindi and S. Boyd, “Log-det heuristic for matrix rank minimisation with applications to hankel and euclidean distance matrices,” 2003.

[13] “Coupon collector's problem,” [Online]. Available: http://en.wikipedia.org/wiki/Coupon_collector%27s_problem.

[14] E. Candes and T. Tao, “The power of convex relaxation: Near optimal matrix completion,” 2009.

[15] E. Candes and Y. Plan, “Matrix Completion with Noise,” 2009.

[16] E. Hale, W. Yin and Y. Zhang, “A Fixed point continuation Method for l1-regularized minimisation with applications to compressed sensing,” 2007.

[17] R. Keshavan and S. Oh, “OPTSPACE : A gradient descent algorithm on the grassman

manifold for matrix completion," 2009.

[18] R. Keshavan, A. Montanari and S. Oh, "Matrix completion from a few entries," 2009.

[19] S. Funk, "Netflix update: Try this at home," 2006. [Online]. Available: http://sifter.org/~simon/journal/20061211.html.

[20] M. e. al, "Low-Rank Matrix Recovery and Completion via Convex Optimization," http://perception.csl.uiuc.edu/matrix-rank/home.html. [Online].

[21] E. Candes, L. Ma and L. Wright, "Robust Principal Component Analysis?," *Journal of ACM 58(1),* 2009.

[22] Z. Zhang, A. Ganesh, X. Liang and Y. Ma, "TILT: Transform Invariant Low-rank Textures,," 2010.

[23] M. etal, "APPLICATIONS: matrix completion," [Online]. Available: http://perception.csl.uiuc.edu/matrix-rank/applications.html.