Solving Matrix Factorization from Lower Dimensional Projections: Applications in Collaborative Filtering and Magnetic Resonance Imaging

Student Name: Anuj Rajani

IIIT-D-MTech-CS-DE-12-035

Jul 1, 2014

Indraprastha Institute of Information Technology

New Delhi

Submitted in partial fulfilment of the requirements
for the Degree of M.Tech. in Computer Science,
with specialization in Data Engineering

Keywords: Low rank matrix factorization, Least squares, Collaborative Filtering, MRI, Stochastic Gradient Descent, Landweber, IRLS, Lower dimensions, Latent factors

Certificate

This is to certify that the thesis titled "**Solving Matrix Factorization from Lower Dimensional Projections: Applications in Collaborative Filtering and Magnetic Resonance Imaging**" submitted by **Anuj Rajani** for the partial fulfilment of the requirements for the degree of Master of Technology in Computer Science & Engineering (Data Engineering) is a record of the bonafide work carried out by him under my guidance and supervision in the Data Engineering group at Indraprastha Institute of Information Technology, Delhi. This work has not been submitted anywhere else for the reward of any other degree.

Dr. Angshul Majumdar
Indraprastha Institute of Information Technology, New Delhi

Abstract

Low-rank matrix factorization finds applications in large number of problems in signal processing and machine learning. Stochastic gradient descent (SGD) is a standard technique for solving large scale matrix factorization problems. However, SGD can only solve the problem when the matrix is directly observed.

There are several applications where the matrix is not directly observed but is only available via linear projections. In such a case the problem is to estimate the low-rank matrix from its projections. To solve such problems, we have derived an algorithm based on the majorization minimization approach – this effectively decouples the projection from the matrix factorization problem via Landweber iterations. After decoupling, the factorization problem can be solved efficiently using SGD.

The original matrix factorization problem decomposed a matrix into two dense matrices. In recent times, several applications required being factored into a dense and sparse component. We recast such problems into the original low-rank matrix factorization framework via the Iterative Reweighted Least Squares technique.

The proposed algorithm is applied to two problems. The first one is on collaborative filtering where the problem is to predict user's choices on unrated items based on the user's and other user's choices on similar items. Traditionally this problem was to recast into a matrix factorization problem via latent factor modeling. However, we noticed that significantly better recovery is achieved when instead of factoring the matrix into two dense matrices; it is factored into a dense and a sparse matrix. In this thesis we have explained this phenomenon intuitively.

The second problem we look at is a dynamic MRI recovery problem. Here the problem is to recover a sequence of dynamic MRI frames from the observed samples. This turns out to be a low-rank recovery problem owing to the spatial correlations among successive frames. However, upon closer look it turns out that the spatial correlation can be modeled as a smoothly varying function which in turn has a compact support in the Fourier domain. Thus, this too can be recast as a matrix factorization into sparse and dense components. Our approach does not improve the accuracy of recovery but helps achieve the goal in a more efficient fashion.

Acknowledgments

First and Foremost, I would like to sincerely thank my advisor Dr. Angshul Majumdar for his immense continuous support during my M.Tech. Thesis Research. I am very grateful to have got the opportunity to work under him. His guidance helped me throughout the research work and without it; this thesis work would not have been possible.

Besides my Advisor, I would like to thank PhD. Students Hemant Aggarwal and Anupriya Gogna for so many helpful and knowledgeable discussions. Also, I would like to thank all faculty members of IIIT Delhi for providing me such a brilliant platform to enhance my knowledge.

I would also like to thank Dr. Vikram Goyal and Dr. Kalapriya Kannan for their acceptance to be a part of my thesis committee.

Last but not the least a big special thanks to my family for supporting me throughout my life.

Anuj Rajani
Indraprastha Institute of Information Technology, New Delhi

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

The base of this thesis relies on the concept of low rank matrix factorization. A detailed study of low rank matrix factorization can be seen in [1]. Here, we will briefly discuss Matrix factorization and its techniques. Matrices that can be simplified into product of two simpler matrices with lower dimensions offer a very useful insight in the analysis of high dimensional data. Low rank matrix factorization is an extremely popular area of research when dealing with analyzing big data in form of latent features with small dimensions. This concept of matrix factorization has been used for various applications. The ones, discussed in this dissertation are Collaborative Filtering (CF) and Magnetic Resonance Imaging (MRI).

The latent factor model suggests representing the important aspects of data in lower dimensions, hence, requiring less storage requirements along with studying the underlying factors affecting the high dimensional data. Let the tabulated data be a matrix Y of m*n dimensions, which is to be approximated by a product of two simpler lower dimensional matrices, U (dimensions m*k) and V (dimensions k*n), where k (k<n and k<m) is said to be the rank of matrix Y, signifying Y to be a low rank matrix.

$$X=UV$$

Let columns of V be represented by V_j (j=1, 2…….n). The linear combinations of the columns of U with the coefficients of V_j leads to columns of X, signifying that U is the basis of column space of X and V is a basis for row space of X.

1.1 Matrix Factorization techniques

Some of the other matrix factorization techniques are as follows:-

1) LU decomposition-

This algorithm simplifies the matrix, X, as the product of lower (L) and upper triangular (U) matrix and is applicable to square matrices i.e. n*n.

- $X=LU$

2) SVD (Singular value decomposition) –

This algorithm divides the original matrix as product of three matrices and is applicable to rectangular matrices i.e. m*n.

$X=U\Sigma V^T$

where, $X \rightarrow$ original matrix

U and $V \rightarrow$ unitary matrices (V^T denotes transpose of V)

$\Sigma \rightarrow$ non –negative diagonal matrix (these are singular values of the original matrix which are arranged in descending order i.e. $\sigma_1 >= \sigma_2 >= \ldots\ldots\ldots\ldots >= \sigma_{\min\{m,n\}}$)

$$\Sigma = \begin{bmatrix} \sigma_1 & & & & & & \\ & \sigma_2 & & & & & \\ & & \ddots & & & & \mathbf{0} \\ & & & \sigma_r & & & \\ & & & & 0 & & \\ & \mathbf{0} & & & & \ddots & \\ & & & & & & 0 \end{bmatrix}$$

Fig1 Singular values in matrix

When approximating the measurement matrix Y via using low rank matrix factorization, the degree of correctness of the approximation is measured by using a statistical metric known as least squares error.

1.2 Least Squares Problem Statement

A least squares problem is described as follows-

$\min_X F(x)$

where, $F(x) \rightarrow \Sigma_i r_i(x)^2, \; 1 \leq i \leq m$

$r_i(x) \rightarrow$ residual error between the actual and estimated vector i.e. y-Ax

$A \rightarrow$ Restriction operator

So, the problem can be represented as:

$$||r_i(x)||_2^2$$

Where, $||.||_2$ is known as l2-norm, which is defined as $\sqrt{\Sigma_i \, ri(x)^2}$. As it is root of sum of squared differences, mean squared error (MSE) measurement. l_2 norms are typically used when the vector is dense.

As $r_i(x) = y_i - Ax_i$

$$||\, y_i \; - \; Ax_i||_2^2 \text{ Or } ||Y - AX||_F^2 \tag{1}$$

where, $||.||_F$ is known as Frobenius norm (root of sum of squared singular values) and Y, X are represented in matrix forms

Solving Least squares:-

$||y - Ax||_2^2$ = $(y-Ax)^T (y-Ax)$

⇨ $y^Ty - y^T Ax - x^TA^T y + x^TA^TAx$

Taking gradient w.r.t . x:-

⇨ $-A^Ty - A^Ty + 2 A^TAx$ (2)

⇨ Equating (2) to 0

⇨ $(A^TA)x=A^Ty$

⇨ $x=(A^TA)^{-1}A^Ty$

Iterative matrix factorization algorithms [2] are followed for solving such system of linear equations as the inverse of A^TA may not always exist and even if it does, it could get too expensive to calculate it for large matrices i.e. it would take $O(n^3)$ with matrix size being n*n.

When the model is trained on a set, it may start memorizing the values instead of recognizing patterns when applied on test dataset. This problem is known as over-fitting.

In order to avoid over-fitting in the reconstruction, regularization terms are added to (1).

$$||y - Ax||_2^2 \; + \; \lambda \; ||x||_2^2 \tag{3}$$

where, λ is the regularization parameter.

For improving the reconstruction, the penalty added as regularization is updated to l_1 norm from l_2 norm –

$$||y - Ax||_2^2 + \lambda \, ||x||_1 \qquad (4)$$

where, $||.||_1$ is known as l_1-norm, which is defined as the sum of the non-zero values in the vector.

l_1-norm penalty puts a constraint of sparsity on x and also performs better in case of presence of outliers whereas l_2 norm causes a smoothening of the function which always leads to a dense vector which might not be expected in case of sparse recoveries. The description of a matrix factorization algorithm which uses l_1-norm penalty in order to tackle the case of outliers can be found in [3].

1.3 Iterative algorithms used for solving least squares problem

1) Alternating Least Squares (ALS) :

ALS is a low rank matrix factorization algorithm used for solving problem of least squares.

Let the rank be r. Now, the reconstructed matrix can be represented as:-

$$X = UV \qquad (5)$$

Where, $X \rightarrow$ solution matrix of dimensions say m*n

$U \rightarrow$ factorized part of dimensions m*r

$V \rightarrow$ factorized part of dimensions r*n

As the matrix has been factorized into two simpler parts containing the latent factors, the problem of (3) gets transformed as follows:-

$$||Y - A(UV)||_F^2 + \lambda_1 \, ||U||_F^2 + \lambda_2 \, ||V||_F^2 \qquad (6)$$

where, λ_1 and λ_2 are the regularization parameters

As both U and V are unknown in (6), ALS algorithm progresses by fixing one of U or V for computing the other. When U is fixed, ALS computes V by solving least squares problem and vice versa. This way of progressing finally leads to convergence as in every iteration, U and V both are getting updated based on the latest update of the other factorized part.

ALS Algorithm –

Step 1- Initialize: iteration counter k = 0; initial estimate X.

Step 2- Initialize one of the factorized parts, say U, randomly.

Repeat the following steps until convergence -
Step 3- Compute other factorized part V, from X and U solving least squares problem. In MATLAB, it can be done by mldivide operator '\' i.e. V=U\X.

Step 4- Now, using X and new V computed in previous step, update U, again solving least squares problem. In MATLAB, it can be easily done by using mrdivide operator '/' i.e. U=X/V. Go to Step 3.

2) Stochastic Gradient Descent (SGD):

SGD is another popular low rank matrix factorization algorithm which is used for solving least squares problem. SGD is like the gradient descent approach with the addition of stochastic attribute in it. Reconstructed matrix is represented the same way as in (5). It uses matrix factorization to solve the objective function, which remains the same as (6).

For computing the reconstruction, this algorithm approaches by iterating over the entire training set and uses the computed error values (residual between actual and predicted values) for updating the factorized parameters of X i.e. U and V. Following is a broader view of the update procedure:-

- $e_{ij} = y_{ij} – v_j . u_i$ (7)
- $v_j = v_j + \gamma .(e_{ij} .u_i - \lambda . v_j)$ (8)
- $u_i = u_i + \gamma .(e_{ij} .v_j - \lambda . u_i)$ (9)

where, $e \rightarrow$ error between original and reconstructed elements

$y \rightarrow$ original vector

$U \rightarrow$ factorized part of X with dimensions m*r

$V \rightarrow$ factorized part of X with dimensions r*n

$r \rightarrow$ rank of matrix X

$\gamma \rightarrow$ learning factor, which affects step size

$\lambda \rightarrow$ regularized parameter to cut down on overfitting of features

After running a certain number of iterations of the above mentioned updates (7), (8) and (9), the algorithm converges to a minimum. In each iteration it makes a small step in the opposite direction of the gradient, which as specified above, is controlled by the learning rate parameter γ.

3) Landweber Iterations:

This method for solving least squares is usually used for ill posed system of linear equations. It is based on the method of Majorization-Minimization. The objective function (J(x)) to be minimized is the same as (1).

Majorization-Minimization (MM) is a method used in optimization theory, which approaches the problem of least squares by taking a function, $G_k(x) \geq J(x)$, where $G_k(x)$ =J(x) only at $x=x_k$ (i.e. function G(x) majorizes J(x)) and J(x) is the objective function. The chosen function $G_k(x)$ is generally easier to minimize. So, on each iteration, the solution vector is updated based on the version of the vector in the previous iteration, such that $J(x_{k+1}) < J(x_k)$, leading into a sequence of vectors x_k, k=0,1,2… ultimately converging to the desired solution.

Algorithm is as follows:-

MM Algorithm

Initialize: Iteration counter k = 0; initial estimate x_0.

Repeat the following steps until a suitable exit criterion is met.
Step 1: Chose $G_k(x)$ such that:

1. $G_k(x) \geq J(x_k), \forall x$
2. $G_k(x) = J(x_k)$ at $x=x_k$

Step 2: Set: $x_{k+1} = \min G_k(x)$
Step 3: Set: k=k+1 and return to step 1.

Graphical representation of MM procedure:-

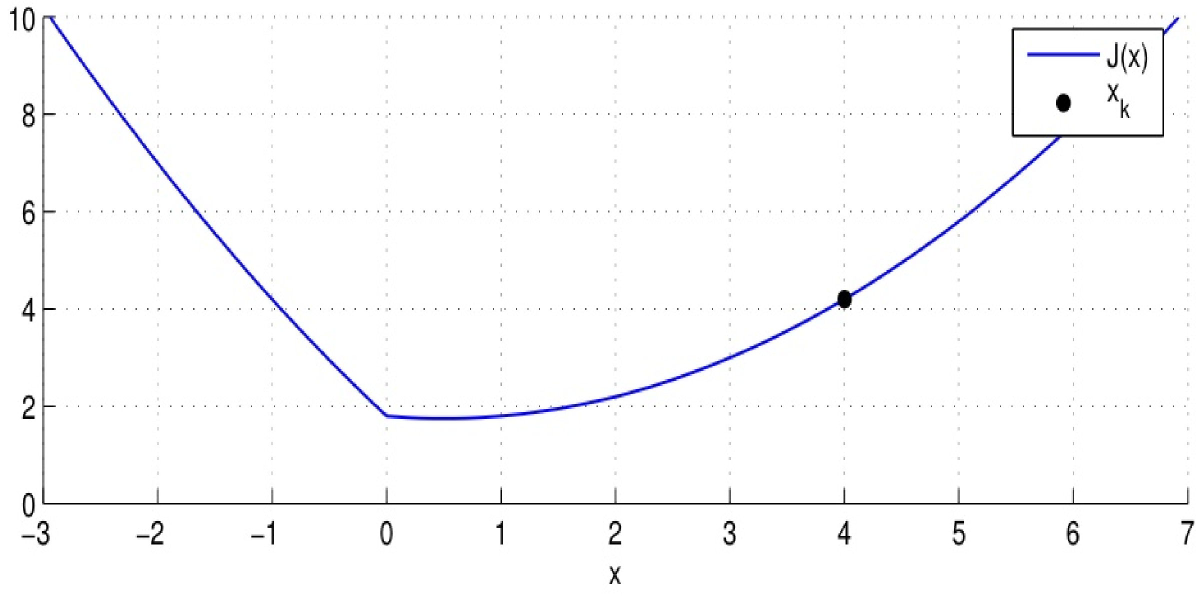


Fig2a) (This figure depicts the objective function J(x) to be minimized with an initial estimate of x_k)

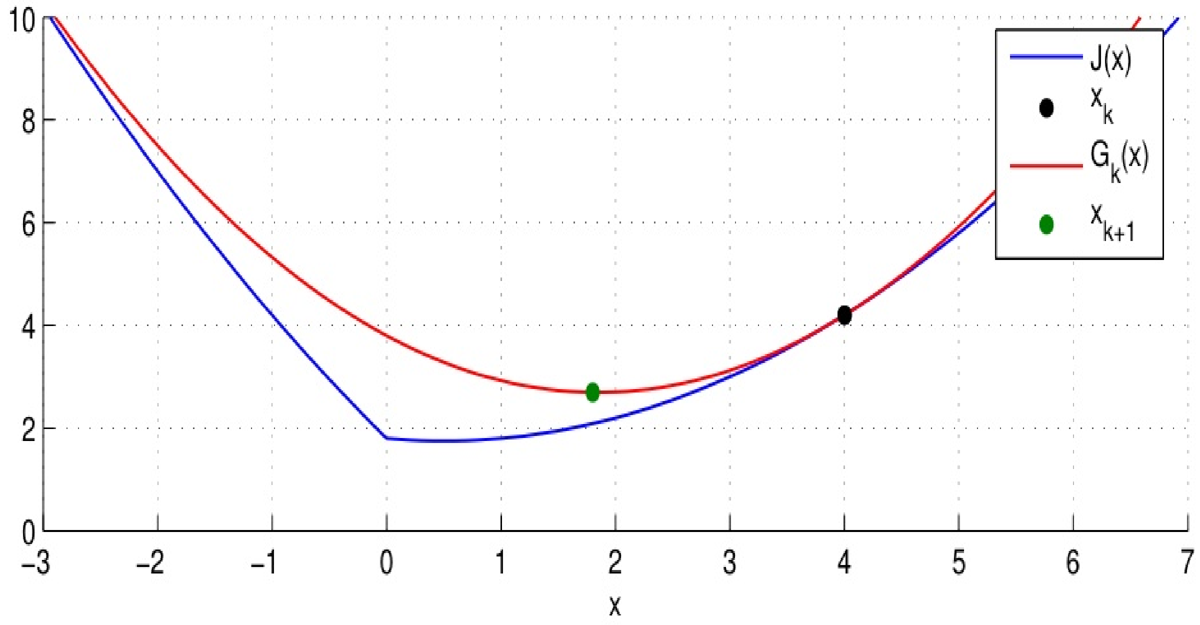


Fig2b) (This figure represents the function $G_k(\mathrm{x}) \geq \mathrm{J}(\mathrm{x_k})$ chosen to be minimized such that $G_k(x)=J(x_k)$ only at $x=x_k$. The value x_{k+1} obtained via minimizing $G_k(x)$ is shown a green dot in the figure.)

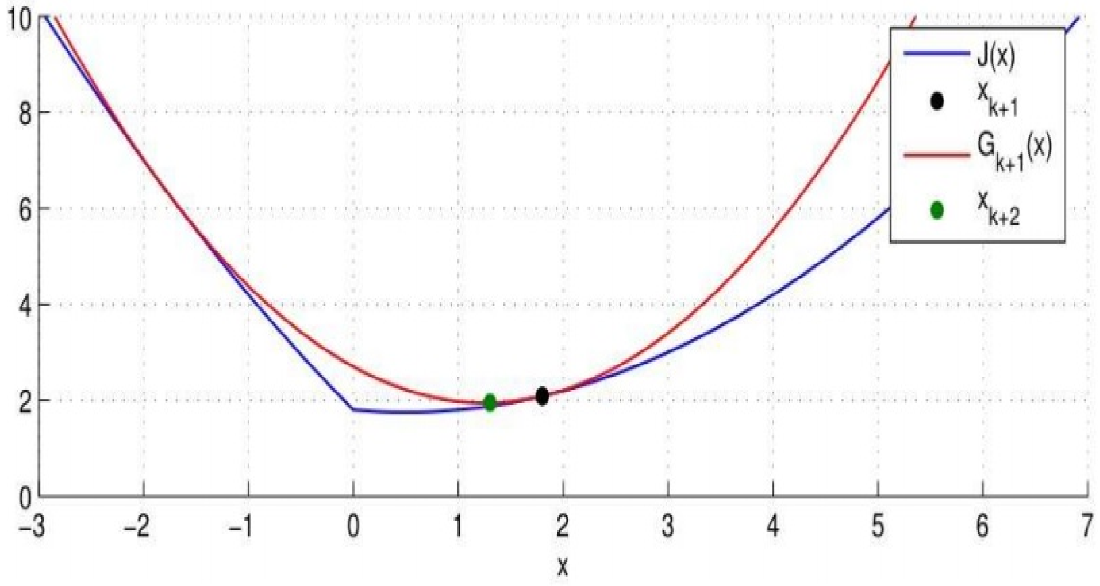


Fig2c) (This figure depicts the next iteration of MM procedure i.e. selection $G_{k+1}(\mathbf{x}) \geq \mathbf{J}(\mathbf{x}_{k+1})$ such that $G_{k+1}(x)=J(x_{k+1})$ only at $x=x_{k+1}$. The minimizer for G_{k+1} i.e. x_{k+2} is displayed as a green dot in the figure.)

Using MM Approach, $G_k(x)$ is selected by adding a non-negative function to J(x) such that the function is 0 at $x=x_k$.

$G_k(x)$ = J(x) + non-negative function

$J(x)= ||y - Ax||_2^2$

Hence, $G_k(x) = ||y - Ax||_2^2 + (x-x_k)^T (\alpha I - A^TA) (x-x_k)$ (10)

⇨ $(y-Ax)^T (y-Ax) + (x-x_k)^T (\alpha I - A^TA) (x-x_k)$

⇨ $y^Ty - 2y^TAx + x^TA^TAx + + (x-x_k)^T (\alpha I - A^TA) (x-x_k)$

⇨ $y^Ty + x_k^T (\alpha I - A^TA) x_k - 2 (y^TA + x_k^T (\alpha I - A^TA)) x + \alpha x^Tx$ (11)

Taking gradient w.r.t. x of (11) and equating it to 0

⇨ $-2A^Ty - 2 (\alpha I - A^TA) x_k + 2 \alpha x = 0$

⇨ $x = x_k + (1/\alpha) (A^T (y-Ax_k))$ (12)

The equation (12) is known as landweber iteration method and it guarantees convergence of solution due to the application of MM procedure in it.

4) Iterative re-weighted Least squares (IRLS):

When dealing with the problem of least squares as defined in (1), some measures are taken in order to avoid over-fitting and to deal with the ill-posedness of the data in order to make the data smoother. The measure taken is regularization. Regularization is defined as a process adding some extra constraint on the solution vector. For just dealing with the problem of over-fitting usually l_2-regularization is used as described in (3). But sometimes, data is distributed a bit irregularly giving rise to outliers and sparsity in the data.

For such problems, l_0 regularization is taken into account, which results in reduced complexity due to its sparseness constraint. l_0 norm is defined as number of non-zero values in the vector and as it is NP hard to compute for, regularization is relaxed to l_p norms. Still, as lp-norms $(0 < p < 1)$ are not convex, they are converted into weighted l2 norms in order to use the same convex optimization methods for l_p as used for l_2-norm minimization.

L_p norm is defined as:-

$$||x||_p = \Sigma_i \, (x_p)^{1/p} \qquad (13)$$

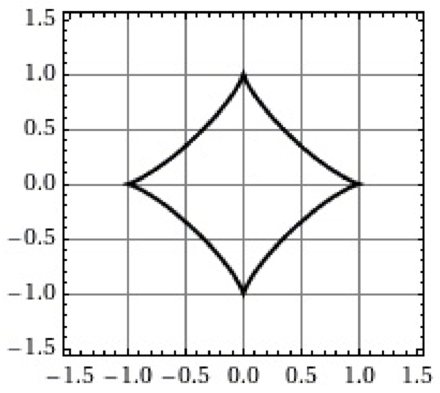


Fig3 l_p-norm (0<p<1) [Non-convex]

Conversion of lp-norm to weighted l2-norm:

$$||x||_p^p = \Sigma_i \, (x_i)^p$$

$$\Rightarrow \quad \Sigma_i \, (x_i)^{p-2} \, (x_i)^2$$

Taking W as a diagonal matrix represented by $(X)^{(p/2)-1}$ (14)

⇨ $\Sigma_i \ (w_i)^2 (x_i)^2$

⇨ $||WX||_F^2$ (15)

IRLS progresses with using this diagonal weight matrix for updation of the matrices/vectors getting the solution vector nearer to the optimum value with each progressing iteration.

Chapter 2

Proposed Method

In this work, we propose an efficient algorithm to solve the problem of least squares where solution vector is not directly observed, using low rank matrix factorization and fractional norm penalty i.e. looking at the sparse property of latent factor matrices.

Our proposed algorithm is a combination of three algorithms- Iterative Re-weighted Least squares, Landweber iterations and Stochastic Gradient Descent, with each algorithm considered as a different phase.

2.1 Phase 1 (IRLS)

The least squares problem depicted in (3) focuses on l_2-norm minimization but as some vectors tend to be sparse, it would be more efficiently recovered if this spasreness is taken into account. In order to achieve this, the problem in (3) is transformed into l_1 minimization problem depicted in (4). l_1-norm enforces the sparse attribute but as it is observed that fractional norms i.e. l_p norms (0<p<1) yield even better results than l_1-norms, l_p norms are used instead. As l_p norms (0<p<1) are not convex, it would be better if it can be transformed into l_2 norm as it would be convex and efficient algorithms can be applied for its minimization. Thus, for such task of representing l_p norm as weighted l_2 norm, Iteratively reweighted least squares is taken into consideration.

IRLS Algorithm:-

Step 1- Initialize the matrix, X, randomly with dimensions m*n.

Repeat the following steps until convergence:

Step 2- Evaluate the weight vector via using formula in (14) i.e. w=$\left(\mathrm{vec}(X)\right)^{\left(\frac{p}{2}\right)-1}$, where p is the norm being calculated.
Step 3- Diagonalize the weight vector into a diagonal matrix, W=diag(w).
Step 4- Update the vector x with x=Wx and the operator A with A=WA. Go to Step 2.

2.2 Phase 2 (Landweber Iterations)

For the regularized least squares problem described in (3), as the matrix is not directly observed i.e. a restriction operator, A, is applied on it to analyze it via linear projections, we have proposed an efficient algorithm which first decouples the projection from the matrix and then solves the transformed low rank matrix factorization problem.

To deal with the linear projection issue, landweber iteration is used. From (11),

$G_k(x) = y^T y + x_k^T (\alpha I - A^T A) x_k - 2 (y^T A + x_k^T (\alpha I - A^T A)) x + \alpha x^T x$

⇨ $\alpha(-2b^T x + x^T x) + c$

where, $b = (1/\alpha)(A^T y + (\alpha I - A^T A) x_k)$

- $b = x_k + (1/\alpha)(A^T (y - Ax))$ (16) [As from (12)]
- $c = y^T y + (\alpha I - A^T A) x_k$

⇨ $\alpha ||b - x||_2^2 - \alpha b^T b + c$ (17)

where, $||b - x||_2^2 = b^T b - 2b^T x + x^T x$

Hence, as the expression "($\alpha b^T b + c$)" does not depend on x, the problem can be thought to be reduced from $||y - Ax||_2^2$ to $||b - x||_2^2$, which as involves direct observation of matrix/vector x, can be efficiently solved via Stochastic gradient descent.

2.3 Phase 3 (SGD)

After Phase 2, the problem gets reduced to:-

$$||b - x||_2^2 \quad (18)$$

which, as per our algorithm is solved using SGD. As already discussed in "Chapter-1 Introduction", Stochastic Gradient descent is an iterative matrix factorization algorithm for solving least squares problem.

SGD Algorithm:-

Step 1- Initialize the factorized matrices U and V randomly with dimensions m*r and n*r respectively, with r being the rank.

Repeat the following steps until convergence:
Step 2- Iterate the entire training set and for each training sample do the following [apply formulae from (7), (8) and (9)]:

a. Calculate the error between actual and predicted values
   - $e_{ij} = y_{ij} – v_j^T . u_i$

b. Iterate k from 1 to r and do (r being assumed low rank of the matrix)-
   - $v_{jk} = v_{jk} + γ .(e_{ij} .u_{ik} - λ . v_{jk})$
   - $u_{ik} = u_{ik} + γ .(e_{ij} .v_{jk} - λ . u_{ik})$

Step 3- Reconstruct the matrix – UV^T. Go to Step 2.

2.4 Combined Summarized Algorithm

Proposed Algorithm:-

Step 1- Initialize the matrices, U and V, randomly with dimensions m*r and n*r, respectively.

Repeat the following steps until convergence:
Step 2- As L_p norm is to be minimized use IRLS for its conversion into weighted l_2-norm for all matrices necessary, as discussed in Phase 1. Calculate weight vector for each matrix, for instance, for U, w= $(abs(vec(U)))^{(p/2)-1}$. W=diag(w). Update the vectors via multiplication with the diagonal weight matrix.

Step 3- Apply regularization on U and V.

Step 4- Evaluate X via $U*V^T$.

Step 5- Apply Landweber iterations on vec(X) as discussed in (16) in Phase 2.

Step 6- Now, apply SGD on the transformed problem of $||B-X||_F^2$ as discussed in Phase 3 thus, updating U and V. Go to Step2.

Chapter 3

Collaborative Filtering

In these modern days, there has been a plethora of information out of which finding appropriate useful information has gotten very challenging due to information overload. Nowadays, a lot of transactions are carried out via online stores such as buying books, movie choices, buying products etc. Earlier, people used to rely on recommendation from other people by social interactions, general surveys etc. But now, due to humongous amount of choices, it is very hard for the customer to select an item of interest. Thus, they rely on automated recommender systems for help. This is the task where collaborative filtering comes into picture. The term was coined by the developers of one of the first recommender systems, Tapestry [4].

Collaborative filtering is a method for predicting user's preferences on items based on his previous preferences and the preferences of other users. It makes the decision in collaboration with other users, hence the term, collaborative filtering. The basic approach of CF is that if two users rate certain no. of items similarly, then they will do so for other items as well [5].

For the information filtering problems in the domain of collaborative filtering, there are different types of formulations used for the data. In this thesis, the formulation used in the dataset for CF problem has the preference indicators as numerical rating triplets of the form (u, i, r) where u is an index specifying the user, i is an index specifying the item and r is the rating value given by the user u to the item i.

Another important aspect of the formulation besides preference indicator is whether the ratings are explicitly given or implicitly formulated based on user's clicks and browsing patterns [6]. A comparison between implicit and explicit ratings can be studied in ClayPool [7]. According to ClayPool, explicit ratings benefit more than the implicit ratings due to overhead involved in computing the implicit ratings from user's browsing history/patterns. In this thesis, we deal with explicit ratings provided in the MovieLens dataset.

The primary task of collaborative filtering is recommendation i.e. it provides users with advice as to what products would be of their interest based on the past preferences of them and other users. Here, the case of collaborative prediction serves as a matrix completion problem i.e. it predicts the missing values in the partially observed user's preferences matrix. For instance, in this thesis for CF application, a movie recommendation dataset is considered, where the input is in form of explicit ratings given by users for the movies they have seen. As typically, users have not seen all the movies present in the dataset, the matrix tends to be partially filled. Hence, the matrix is completed by making predictions for the movies users haven't seen yet based on patterns observed in the data. Although, there are various modeling algorithms for this task but here, the traditional approach of matrix factorization will be used where predictions are made based on the latent factors of the items/movies (director, actor, genre etc.) and users (affinity of users towards the factors of items), thus enabling users to collaborate based on the hidden attributes affecting their preferences instead of relying on some external information.

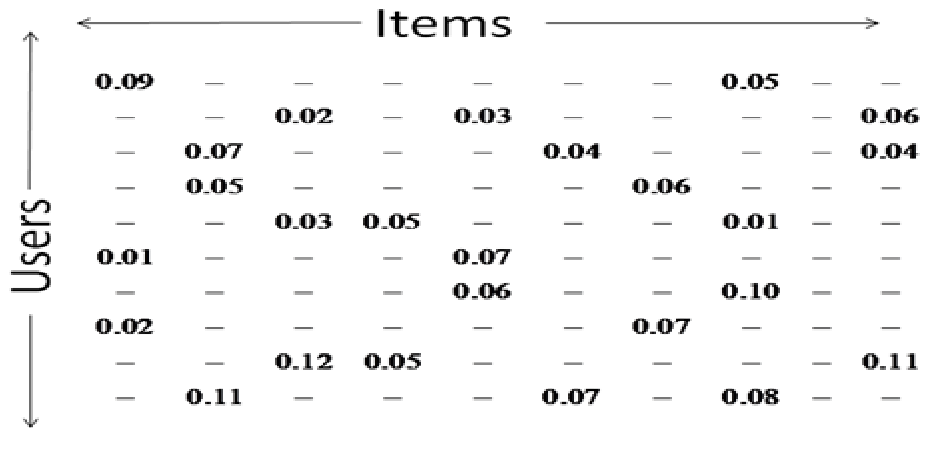
Attributes	Sci-Fi	Action	Comedy
User A	U_{11}	U_{12}	U_{13}

X

Attributes	Movie A
Sci-Fi	I_{11}
Action	I_{21}
Comedy	I_{31}

Predicted rating of user A for movie A = $U_{11} * I_{11} + U_{12} * I_{21} + U_{13} * I_{31}$

There are different applications of CF in this recommendation domain. One of them is being to list the top few recommendations for the user and another being to determine user's preference for each item/movie. Although, the two applications are somewhat related to one another, in this work, we will be explicitly working only with the second application i.e. predicting user's preferences for all un-rated movies. Following is the representation of a matrix completion problem:-


Items
Users

0.09	–	–	–	–	–	–	0.05	–	–
–	–	0.02	–	0.03	–	–	–	–	0.06
–	0.07	–	–	–	0.04	–	–	–	0.04
–	0.05	–	–	–	–	0.06	–	–	–
–	–	0.03	0.05	–	–	–	0.01	–	–
0.01	–	–	–	0.07	–	–	–	–	–
–	–	–	–	0.06	–	–	0.10	–	–
0.02	–	–	–	–	–	0.07	–	–	–
–	–	0.12	0.05	–	–	–	–	–	0.11
–	0.11	–	–	–	0.07	–	0.08	–	–

Fig4 CF matrix completion example

There are various algorithms used for implementing collaborative filtering which can be classified mainly under three types – memory based, model based and hybrid algorithms. These methods are described elaborately in [8]. In this thesis, we will be focusing on model based matrix factorization; however other methods will be discussed briefly for completion.

3.1 CF Methods

1. Memory-Based:

Memory Based CF algorithms rely solely on neighborhood based methods which are used for computing similarity/weight between two users or items. It is assumed that each user has a group of people who have similar interests as him, thus affecting his preferences towards items he hasn't rated yet. In order to find such similar users, the similarity metric used is distance or correlation metric. Based on such similarity metric, active user's preferences can be computed by taking a weighted average of preferences of similar users/items found.

The Neighborhood models can be classified as user-user [9] and item-item [10]. In user-user model, similar users are searched corresponding to the active user whereas in item-item model, similar items are searched corresponding to a given item. Item-item models are found to be more scalable than the user-user models as in a real time system, frequent increase of users in a database is highly probable whereas items tend to increase at certain intervals which aren't relatively as frequent as for users.

For these neighborhood approaches, computation of similarity is very important. There are multiple similarity metrics which can be used for computing similarity. One of the ways is Euclidean distance metric which is computed using formula:-

$$\sqrt{\sum_{i=1}^{n}(\mathrm{xi} - \mathrm{yi})^2}$$

This similarity measure usually works well only when the dataset is dense. Another similarity metric is correlation computed between two user/items via Pearson correlation [11]:-

$$r = \frac{\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{xi} - \overline{x})\,(\mathrm{yi} - \overline{y})}{\sqrt{\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{x_i} - x)^2}\,\sqrt{\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{y_i} - y)^2}}$$

This similarity measure is used in order to handle the bias included in users ratings (some users tend to like everything, some always give binary ratings etc.). When dealing with sparse data, cosine similarity is used as the similarity measure. It is defined as:-

$$\cos(x, y) = \frac{x.y}{||x||*||y||}$$

where, '.' refers to dot product of vectors x and y and ||x|| indicates the length of the vector, which is also termed as norm, defined as $\sqrt{\sum_{i=1}^{n} {x_i}^2}$.

After the computation of similarity, next step is the prediction of preferences, which is the primary task of collaborative filtering. A subset of nearest neighbors is selected based on some defined threshold similarity value and weighted average of their ratings is taken to predict ratings of the active user.

2. Model Based:

Model based CF algorithms employ concepts from known fields such as machine learning and data mining to train the data to identify complicated patterns, thus enabling the system to make efficient predictions on test data. Some machine learning concepts used such as Bayesian models and clustering models are known to solve the limitations experienced in memory based CF algorithms [12, 13]. Classification algorithms are employed for categorical ratings whereas in case of numerical ratings, regression models and SVD methods are used. In this thesis, we will be looking at least squares based regression model; however some other model based algorithms are discussed in brief.

Bayesian Algorithm – It uses the naïve Bayes approach to evaluate for predictions, that relies on computing the probability of a class given all features and declaring the class with highest probability as the prediction output [14].

Clustering – It is a technique of reducing objects to clusters, where clusters are collection of objects that are similar to each other. Some of the similarity metrics used are– Pearson correlation and Minkowski distance. Classification of clustering algorithms is done in three groups- partitioning methods, hierarchical methods and density based methods. There are multiple ways of using these clustering algorithms for making predictions. One of the ways is to first make clusters out of data and then apply memory-based CF approach to make predictions within each cluster [15].

Regression based CF algorithms – Model based Regression techniques prove to be better prediction models as compared to memory based CF algorithms. The basic regression model is defined as:-

$$Y = UV^T + \varepsilon$$

where, $Y \rightarrow$ original measurement matrix with dimensions m*n (m being users and n being items)

$U \rightarrow$ factorized part (specifying users affinity to items latent factors) of matrix X, which is to be reconstructed (dimensions m*r)

$V \rightarrow$ factorized part (specifying items latent factors) of matrix X, which is to be reconstructed (dimensions n*r)

$\varepsilon \rightarrow$ Noise associated with real word measurements

In this work, we have used model based matrix factorization to solve least squares regression problem.

Usually, Y is a sparse matrix, for which Canny [16] proposed a sparse factor analysis, in which the average value of observed elements of the matrix was used to replace the missing elements in the matrix, post which, regression model was used as initialization for expectation maximization. Another approach was proposed in which similar items were searched for [17], then a collection of simple linear models was created (with parameters for regressions being selected using least squares) and combined to rate preferences of the active user. One other approach used slope one algorithms for making rating predictions [18].

3. Hybrid model:

Hybrid CF algorithms are defined to be combination of collaborative filtering and content based filtering systems to make recommendations. Content based approach provides external information/attributes (age group of users, demographics etc.) for users and items in addition to ratings, to make predictions.

3.2 Challenges Faced in CF

Nowadays, as a lot of users use online portals for shopping and looking for products of their interest, the automated recommender systems are expected to make effective and quality recommendations to advance in this competitive era. But there are various challenges which stand in the way of the recommender systems. Some of them are discussed in [8]. Here, we will briefly give an overview of few of these problems.

The most frequently faced problem in CF is the sparseness of data. The cause of this sparseness is either the laziness of users in explicitly rating the item (movie in our case) or unfamiliarity of the user with the item. Users tend to only the rate the movies which they have seen, that too, usually only the ones they liked. This leads into a partially filled preference matrix, otherwise known as the sparseness problem as most of the items remain unrated. This sparseness further creates many problems in the prediction task, one of them being the cold start problem [19] (also called as new user/item problem). This cold start problem occurs in case of new users/items as initially they don’t have any data corresponding to them. Without user rating at-least one movie or for an item getting rated by at-least one user, it gets very challenging to perform recommendation for the users. Some methods handling this issue are discussed in [20].

Another major problem relates to scaling down of data matrix, as computation on very large data sets involve very high computational costs. As recommender systems need to be efficient in terms of running time in addition to its quality, data needs to be scaled down and this can be achieved via several dimensionality reduction algorithms such as SVD (Singular value decomposition). SVD primarily removes the users and items with less significance by decomposing the original matrix into factorized parts. PCA (Principal Component Analysis) is another method used to reduce dimensions [5]. But the catch in dimensionality reduction is that it may lead to a drop in recommendation quality as the information which is getting pruned for lowering the dimensions may turn out to be important. For computing SVD, there is a method which uses existing users [21]. To increase scalability, it uses a folding projection technique [22] which prevents recalculation of SVD for newly added set of ratings. Similarly, to increase scalability, clustering algorithms use an approach to make recommendations to users by selecting smaller and similar clusters, thus preventing the need to traverse entire dataset.

One other problem is Shilling attacks and it is related to bias included by users while rating products, which results into high rating of one's own products but very low rating for someone else's items. Thus, there is an essential need to introduce algorithms to handle this bias to generate better recommendations.

3.3 Problem Statement in context of Collaborative Filtering

As in this thesis we are focusing on low rank matrix factorization, CF problem of matrix completion can be defined mathematically as minimization of regularized least squares as depicted in (6) :-

$$||Y - A\,(X)||_F^2 + \lambda\,||X||_F^2$$

$$||Y - A\,(UV)\,||_F^2 + \lambda 1\,||U||_F^2 + \lambda 2\,||V||_F^2$$

As discussed in Introduction section, U and V are factorized matrices which, in this case of CF represent user's and item's latent factors that secretly affect the user's ratings. The reconstructed ratings matrix X is expressed as the inner product of U and V.

The earliest study in this area [23] used an approach based on probabilistic modeling for collaborative filtering. Then down the line, latent factor models were introduced which gave rise to matrix factorization problem [24, 25]. Matrix factorization is very fast but due to its non-convex nature, researchers found that instead of solving this model as matrix factorization, it can be transformed into a matrix completion problem [26, 27], which although requires high computation time, is convex in nature.

Here, in our work, our objective is to find the factorized latent factor matrices for users and items. We have proposed that although this user matrix is required to be dense, item matrix is not. This is so because an item is not expected to possess all the defined latent factors, for instance in case of movies, if a movie is an action/thriller, it will usually not possess the traits of romance, comedy, animation etc., thus making it's matrix (item latent factor matrix) sparse whereas in case of users, it is highly likely for a user to possess affinity to almost all the features. For instance, if a user likes sci-fi movies, that only signify that he has greater tendency to watch sci-fi than other genres such as action/thrillers i.e. a user is expected to have a degree of likeliness towards all the latent factors, leading it to be a dense matrix. This approach of focusing on sparseness constraint wasn't followed in the previous studies. Thus, as we will be accounting the sparseness of item latent factors, the problem statement gets transformed to the following:-

$$||\mathrm{vec}(Y) - A\,\mathrm{vec}(UV)||_2^2 + \lambda_1\,||\mathrm{vec}(U)||_2^2 + \lambda_2\,||\mathrm{vec}(V)||_1 \qquad (19)$$

Where, $||.||_1$ → represents the l_1 norm of vector V which is the sum of non-zero values of vector V and it imposes a sparseness constraint on it.

A → is the binary mask that considers only non-zeros values/ratings from X i.e. UV

As per our approach U being dense implies an l_2 norm penalty on it and V being sparse implies an l_1 norm penalty. However, as it is known that l_p norm $(0 < p \leq 1)$ yields even better and sparser results than l_1 norm, we have rather proposed l_p norm penalty on V. Also, as usually Absolute error metric is used to calculate error of the reconstructed ratings matrix w.r.t actual ratings, it is more logical to minimize l_1 norm i.e. $||\mathrm{vec}(Y) – A\ \mathrm{vec}(UV)||_1$ instead of l_2 norm. But it has been observed in earlier studies that l_1 norms don't provide better results, therefore, in this approach, we have rather proposed to minimize intermediate l_q norms $(1 < q \leq 2)$, hence transforming the problem statement defined in (19) to:-

$$||\mathrm{vec}(Y) - A\,\mathrm{vec}(UV)||_q^q + \lambda_1\,||\mathrm{vec}(U)||_2^2 + \lambda_2\,||\mathrm{vec}(V)||_p^p \qquad (20)$$

Here, apart from just reconstruction, we also implant a model to overcome the problems introduced by user and item biases i.e. some items (movies in our case) tending to always get top ratings whereas some others always below average. Also, some users have a tendency to like everything, thus, giving top ratings even to a mediocre movie whereas others just can't be overwhelmed by anything, thus giving average rating even to a top class movie. The following model is used to take care of biases along with interaction between user and item [28].

$$X = \mu + B_u + B_i + UV \qquad (21)$$

where, μ → global mean

B_u → user bias

B_i → item bias

Here, $(\mu + B_u + B_i)$ is defined as baseline and UV is known as interaction.

Global mean is fairly easy to compute i.e.

$$\mu = \frac{\text{total of ratings}}{\text{total entries in matrix}} \tag{22}$$

whereas other parameters B_u and B_i are estimated using Potter's estimation:-

$$b_i = \frac{\sum_{u \in R(i)} r_{ui} - \mu}{\mu_1 + |R(i)|} \qquad b_u = \frac{\sum_{i \in R(u)} (r_{ui} - \mu - b_i)}{\mu_2 + |R(u)|} \tag{23}$$

3.4 Error Metric in CF

The metric used for evaluating the quality of recommendation is Mean Absolute Error (MAE)/ Normalized Mean Absolute Error (NMAE). It is defined as:-

$$\text{MAE} = \frac{\text{Total error}}{\text{Total no.of test data entries}}$$

$$\text{NMAE} = \frac{\text{MAE}}{\max(\text{training Set}) - \min(\text{training Set})}$$

The range of MAE and NMAE is 0 to infinity.

3.5 Summary of Proposed Algorithm in terms of CF

Step 1- Load the dataset and make trainSet (80%) and testSet (20%).

Step 2- Calculate the global mean as defined in (22); user bias and item bias as defined in (23).

Step 3- Calculate the baseline as ($\mu + B_u + B_i$) and interaction Y as (trainSet-baseline).

Repeat this step until convergence-
Step 4- Apply the "proposed algorithm" described in Chapter-2 over this dataset to minimize least squares as specified in (20) i.e. apply the three phases in order i.e. IRLS, Landweber and SGD.

Step 5- Add baseline to reconstructed matrix i.e. ($UV + (\mu + B_u + B_i)$) as defined in (21).

Step 6- Return the NMAE and reconstructed ratings matrix X (after adding baseline).

Chapter 4

Magnetic Resonance Imaging (MRI)

In this thesis, we have applied our proposed algorithm for reconstruction of DCE (Dynamic Contrast Enhanced) MRI sequences via under-sampled data in temporal frame. MRI is a radiology based technique used to analyze the body functions and diagnose health issues if any. As it is a technique used in radiology, it takes images of the body part(s) which is/are being examined via strong magnetic fields and radio waves, to analyze them in detail.

Dynamic MRI serves as a very important technique in various medical exams such as functional imaging. Data for dynamic MRI is collected in k-t space, k being the spatial domain (images available in fourier domain) and t being the temporal frame. The problem statement is to reconstruct images in all times frames in order to give the resultant as a smooth MRI video. This reconstruction is done offline i.e. as a post event of collection of k-space samples for all time frames. Such reconstruction is seen to be as an under-determined linear inverse problem.

The spatial correlation within the frame has been used for reconstructing for static MR images [29, 30]. In our work, we use the fact of spatial-temporal data being able to be represented as linear combination of few temporal basis functions. Recent studies in dynamic MRI reconstruction [31-42] use this spatio-temporal correlation for reconstructing MR images from under sampled data in k-space.

4.1 Dynamic MRI Model

The dynamic MRI problem is depicted as:-

$$Y=RFX+\eta \tag{24}$$

where, $Y \rightarrow$ is the collected k-space data in all temporal frames (y_t is collected image at tth time frame)

$R \rightarrow$ is the sampling operator which under-samples data from k-space

F → Fourier operator mapping image space to k-space

X → is the data of images being contained in all temporal domains (x_t being the image at time frame t)

η → system noise

4.2 Motivation for MRI reconstruction

The motivation for this work is to reconstruct MRI sequence efficiently along with good spatial and temporal resolution. Traditional methods usually require all the samples from k-space for reconstruction of frames via inverse FFT. In this case, generally there is often a tradeoff between spatial and temporal resolution. But fortunately, recent research in dynamic MRI [31-42] has provided a helping hand by showing that reconstruction is possible even by partially sampling k-space from each time frame. Although, there is a slight deterioration in image quality due to under sampling of k-space but that is acceptable given the reduction in no. of measurements required for reconstruction.

4.3 Earlier studies in DMRI

Earlier studies on dynamic MRI reconstruction focused only on sparsity of image in transform domain [31-37] but later it was noticed that the entire sequence could be represented using only a few temporal basis functions, leading into a formulation of low rank matrix. This approach was used in [38, 39] for dynamic MRI reconstruction. The works combining transform domain sparsity and low rank attribute of signal [40-42] yield better results than the sparsity approach used in [31-39].

4.4 Mathematical representation of problem Statement

In this thesis, we have used the same approach as used in [38, 39] i.e. of modeling dynamic MRI problem as a matrix which can be reconstructed using only a few temporal basis functions, but have proposed an efficient algorithm to implement this approach. The proposed approach is as defined in Chapter 2 i.e. usage of least squares optimization via low rank matrix factorization and regularized by convex l_1-norm implying the sparsity constraint. The mathematical problem statement taking into account the low rank and sparsity constraints, can be formulated as:-

$$\min_X ||vec(X)||_1 \text{ subject to } ||Y - MX||_F^2 \leq \varepsilon \qquad (25)$$

Where, $||.||_1$ denotes l_1 norm, $||.||_F$ denotes Frobenius norm and ε is the data fidelity term.

Here, l_1 norm was chosen because it was the closest convex replacement for l_0 norm. Transforming (25) into its unconstrained version equivalent, problem statement becomes:-

$$\min_{X} ||vec(Y) - M\ vec(X)||_2^2 + \lambda\ ||vec(X)||_1 \tag{26}$$

The data model as described in (24) represents the dynamic MRI problem where the data is stacked in columns as Y= $[y_1|y_2|\ldots|y_T]$, X= $[x_1|x_2|\ldots|x_T]$, $\eta = [\eta_1|\ \eta_2|\ldots|\ \eta_T]$ and T is the time frames.

The spatio-temporal data to be reconstructed, X, can be represented in a matrix form, with rows corresponding to the spatial part and columns corresponding to the time samples, as follows:-

$$X=\begin{bmatrix} (x_1,t_1) & \cdots & (x_1,t_T) \\ \vdots & \ddots & \vdots \\ (x_m,t_1) & \cdots & (x_m,t_T) \end{bmatrix}$$

Now, as this signal can be modeled as a linear combination of a few temporal basis functions, X can be assumed to be a low rank matrix (say, with dimensions m*n). Applying low rank matrix factorization over it, X can be expressed as, X=UV, as explained in Chapter-1 (theory about matrix factorization) with U representing the coefficient matrix having dimensions m*r and V representing temporal basis functions with dimensions r*n. So, after matrix factorization the problem in (26) can be depicted as:-

$$||\mathrm{vec}(Y) - M\ \mathrm{vec}(UV)||_2^2 + \lambda_1\ ||\mathrm{vec}(U)||_1 + \lambda_2\ ||\mathrm{vec}(V)||_2^2 \tag{27}$$

where, λ_1 and λ_2 are regularization parameters.

Here l_1 is applied on U because it was noticed that the signal X is sparse in its learned basis.

M=RF, is a sampling operator with F being a 2D Fourier operator, which is applied on vectorized MR image. The sparse representation of signal is possible due to the smoothness of DCE MRI signal over time. To solve (27) we have applied the three phase algorithm explained in chapter 2 "The proposed method".

4.5 Error Metric in MRI reconstruction

The metric used for evaluating the quality of reconstructed MR image is Normalized Mean Squared Error (NMSE). It is defined as:-

$$\text{NMSE} = \frac{||\text{Recontsrcuted} \quad -\text{Actual}\ ||_2}{||\text{Actual}\ ||_2} \tag{28}$$

4.6 Proposed algorithm in terms of DMRI

Step 1- Load the dataset 2DDCE1.

Step 2- Create Fourier operator F as FFT 2D matrix of size of spatial part of the data. In MATLAB, it is created via using Sparco operator opFFT2d().

Step 3- Apply sampling scheme on each time frame which selects only 50% of the data in k-space and use “random axis aligned lines” as the pattern for selecting that data from all points in k-space.

Step 4- Create restriction Operator R so as to consider only the sampled locations in k-space in each time frame.

Step 5- Create concatenated operator M from concatenation of R and F.

Step 6- Take only sampled data from all time frames in vector “y” by applying restriction operator M on complete data vector “x”.

Repeat this step until convergence-
Step 7- Apply the “proposed algorithm”(IRLS, Landweber, SGD) described in Chapter-2 over this dataset using observed vector as y and applying l_1 norm over the “U” matrix i.e. the coefficient matrix.

Step 8- Calculate NMSE (using (28)) between reconstructed and actual data by converting both into vectors.

Step 9- Obtain the corresponding reconstructed images of DCE-MRI for the corresponding time frames. In MATLAB, it can be obtained via using function imshow().

Chapter 5

Experimental Results

In this section we will be describing the dataset used for the two applications – CF and MRI. Also, the results evaluated from the proposed algorithm will be discussed here.

5.1 Collaborative Filtering

In this thesis, we have worked with two MovieLens datasets with different sizes, one being 100K and the other being 1M. The data in MovieLens was collected via MovieLens project and is distributed at University of Minnesota by GroupLens research. The datasets consist of movie ratings provided by users. All data taken in account will be in the form of explicit ratings ranging from 1 to 5.

100K dataset contains 943 users, 1682 movies and 1M dataset contains 6040 users and 3952 movies. As per the experimental protocol used by Breese, Heckerman, and Kadie [12], these datasets are divided into a training set and a test set by random distribution with 80% being trainSet and 20% testSet. A 5-fold cross validation is performed.

The model will be trained using the trainSet which being 80% of total data is sparse and the problem is depicted as a matrix completion problem which requires prediction of the ratings of users in the test set, which is further solved using least squares approach.

In these trainSet and testSet, data is represented in form a 2D matrix with rows representing the users and columns the movies. The trainSet can be further split for getting a validation set but in this experiment, validation set is not accounted.

The error metric for measuring the quality of prediction is NMAE (Normalized Mean Absolute Error) as defined in Section 3.4. If $\mathrm{NMAE} \leq 1$, it means that the algorithm has performed better than randomly predicted ratings, otherwise the algorithm is considered worse than the random prediction.

It was observed that for these datasets there are very less amount of features/factors which affected the user's ratings. Hence, for experimental analysis we assumed the rank (latent factors) to be around 20 or 40.

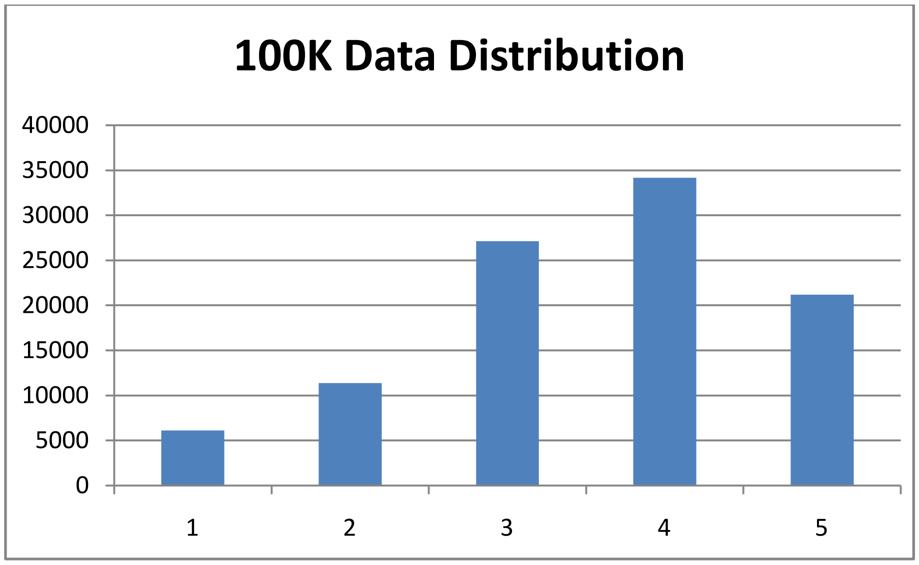


Fig5 a) Ratings Distribution in 100K data (X–axis depicts ratings and Y–axis its frequency)

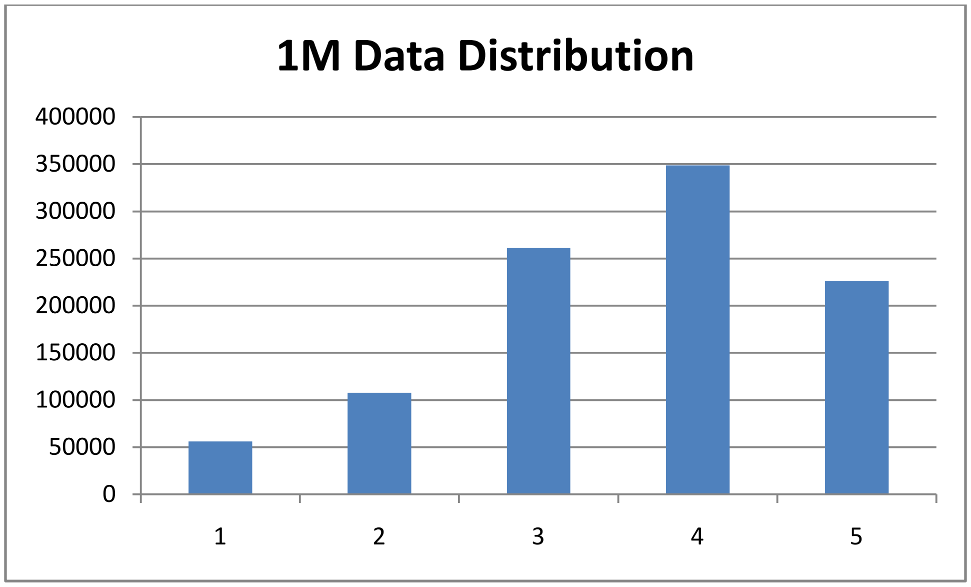


Fig5 b) Ratings Distribution in 1M data (X–axis depicts ratings and Y–axis its frequency)

For landweber iterations, value of α is taken to be 1.05 (slightly more than max. Eigen value of A^*A^T, A being the restriction matrix, which in this CF dataset behaved like a sampling operator for non-zero values within the data).

While calculating item and user bias in (23), the value of μ_1 is used as 10 and μ_2 as 25 as discussed in [43].

The error expression norm and regularization penalty norm as discussed in Section 3.3 and as per (20), was experimented with and after some experimentation was set as p= 0.4 and q=1.8. The values of regularization parameters i.e. λ_1 and λ_2 can be chosen from 10^{-9} to 1 but after a series of simulations, it was manually tuned to $\lambda_1=10^{-4}$ and $\lambda_2=10^{-5}$.

Following are the results on taking the above mentioned values for parameters and ranks as 20 and 40:-

5.1.1 Results for 100K dataset-

Norm / Rank	Rank 20		Rank 40	
	NMAE	Time (sec)	NMAE	Time (sec)
q=1.8, p=0.4	0.1531	1114.369	0.1525	270.674
q=2, p=1	0.1704	717.437	0.1715	871.976
q=2, p=2	0.1726	2721.3261	0.1817	3810.3295

Table1 100K dataset results

5.1.2 Results for 1M dataset-

Norm / Rank	Rank 20		Rank 40	
	NMAE	Time (sec)	NMAE	Time (sec)
q=1.8, p=0.4	0.1473	866.84	0.1557	619.33
q=2, p=1	0.1702	3724.429	0.1744	4491.154
q=2, p=2	0.1735	5459.836	0.1749	10692.96

Table2 1M dataset results

5.1.3 Observation

Results of our proposed algorithm and approach on both datasets imply that using $l_q \; and \; l_p$ norms with $0 < p \leq 1 \text{ and } 1 < q \leq 2$ creates significant improvement in results as compared to l_2 error norm, l_1 penalty norm (q=2, p=1) and the baseline method (q=2, p=2).

5.1.4 Result graphs

- Comparison of our approach of l_q-error and l_p-penalty norm with l_2-error and l_1-penalty norm (x axis- iterations, y axis- NMAE) - **Shows our approach to perform better**

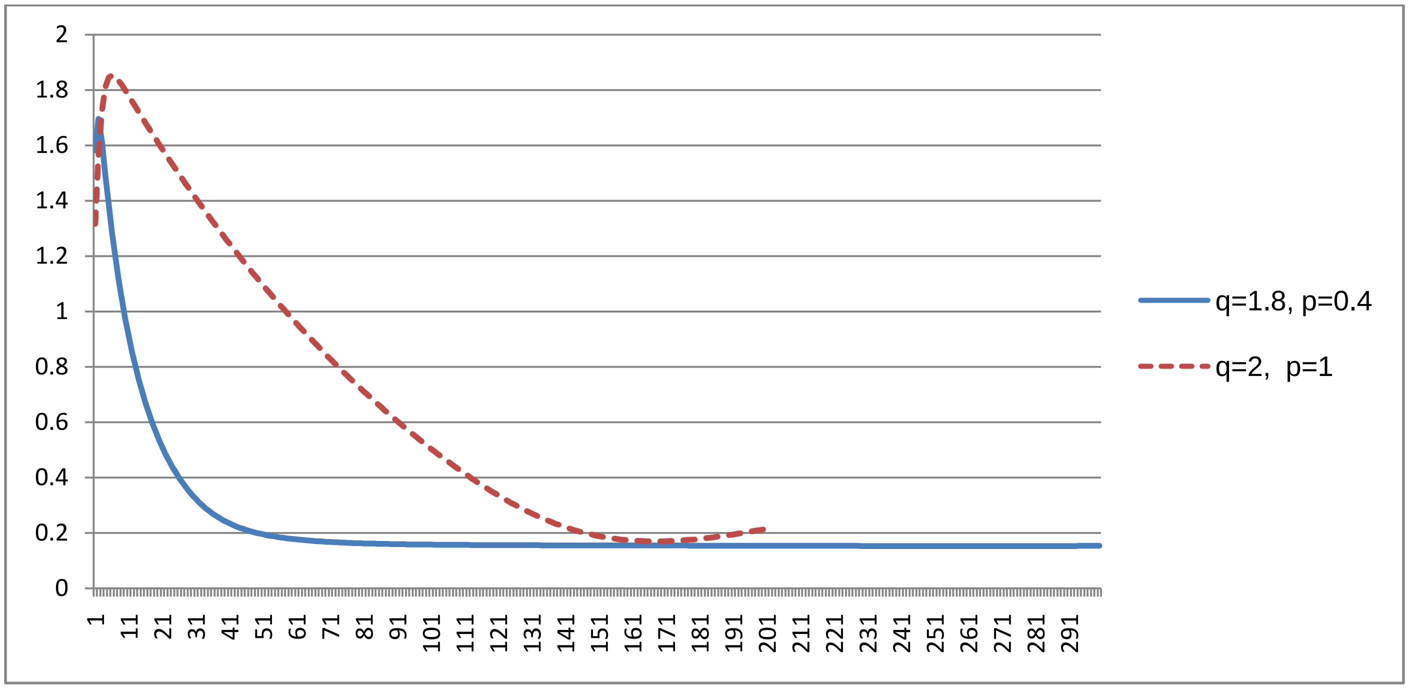


Fig6a) 100K dataset on rank 20

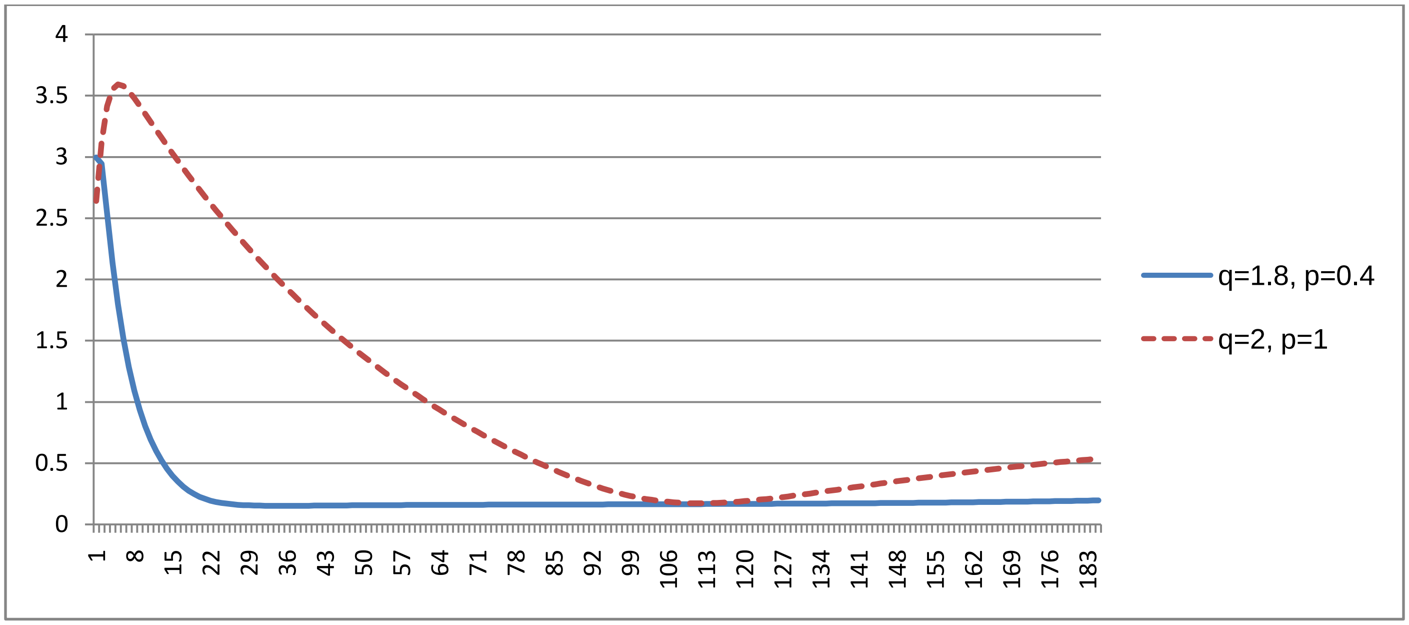


Fig6b) 100K dataset on rank 40

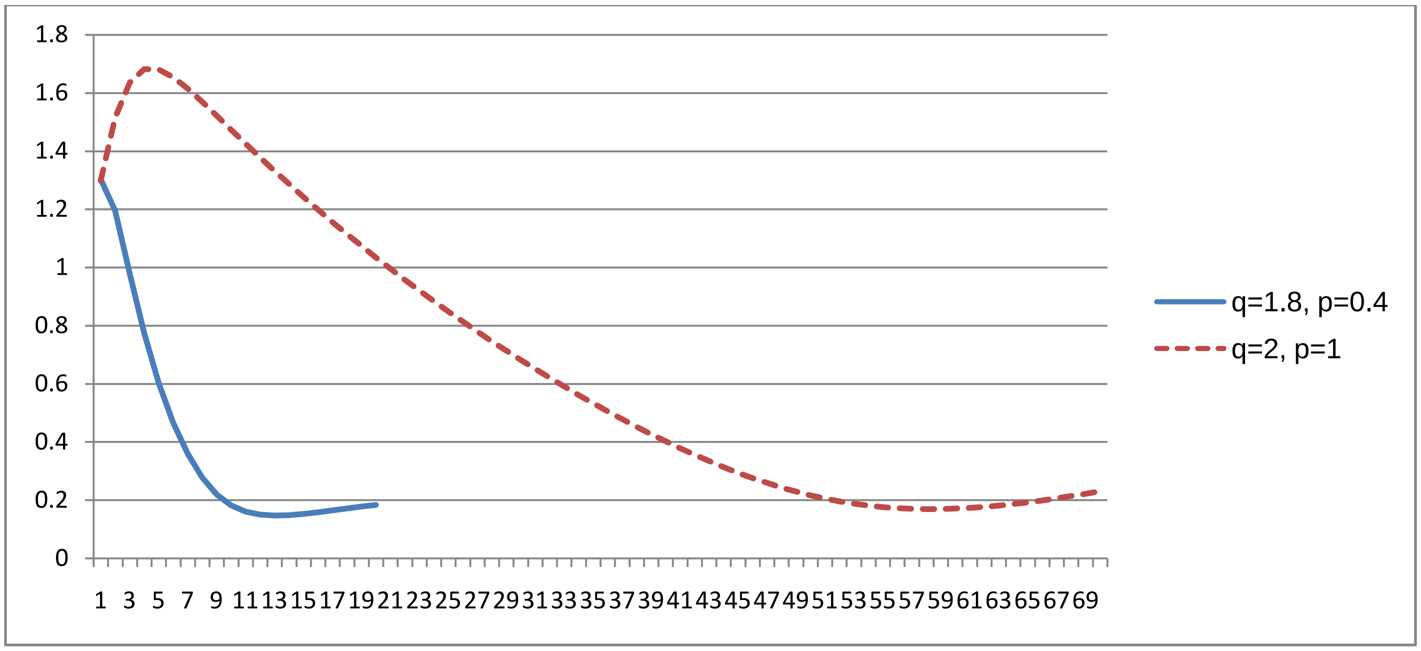


Fig6c) 1M dataset on rank 20

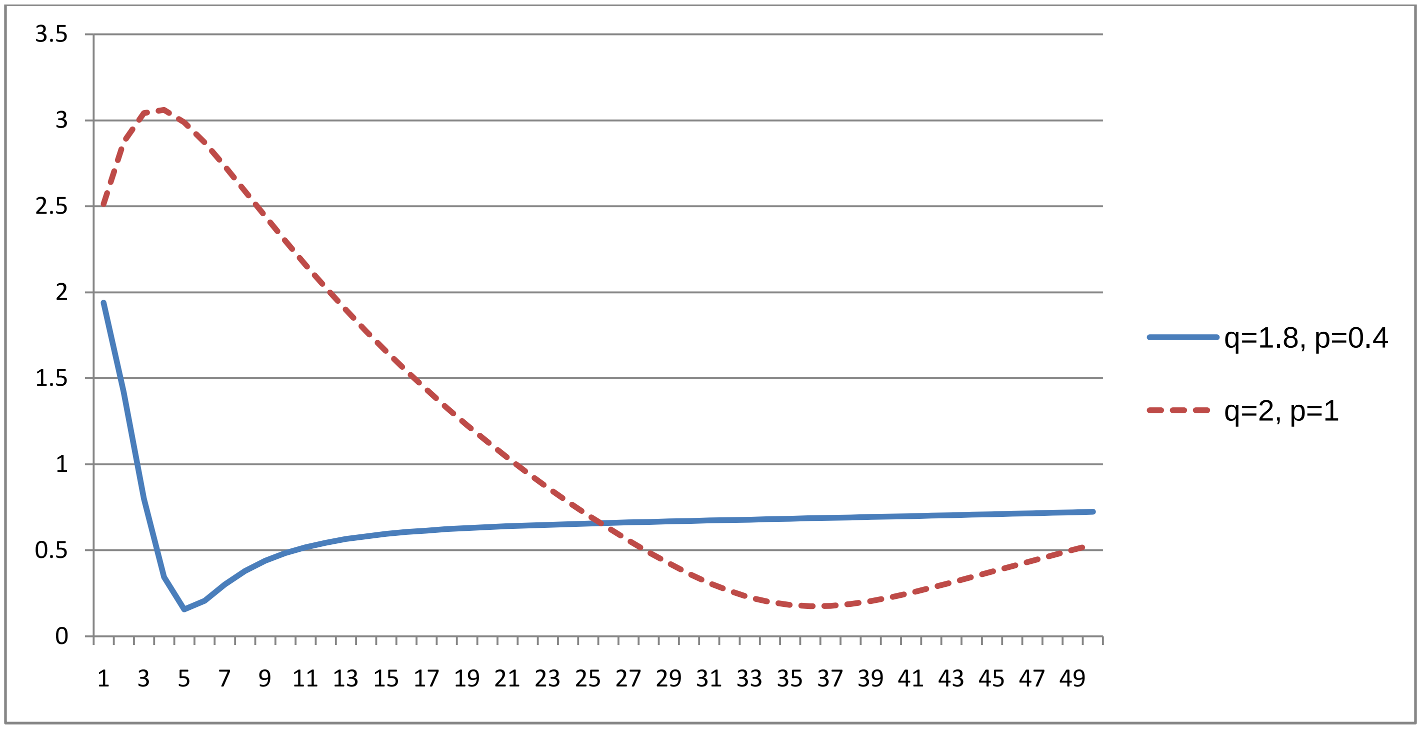


Fig6d) 1M dataset on rank 40

- Comparison of convergence behavior of proposed algorithm on different ranks, with q=1.8 and p=0.4 (**Rank 20 performs better** because it gradually decreases whereas rank 40 drastically decreases then starts to increase, leading into recovery of noise not actual ratings)

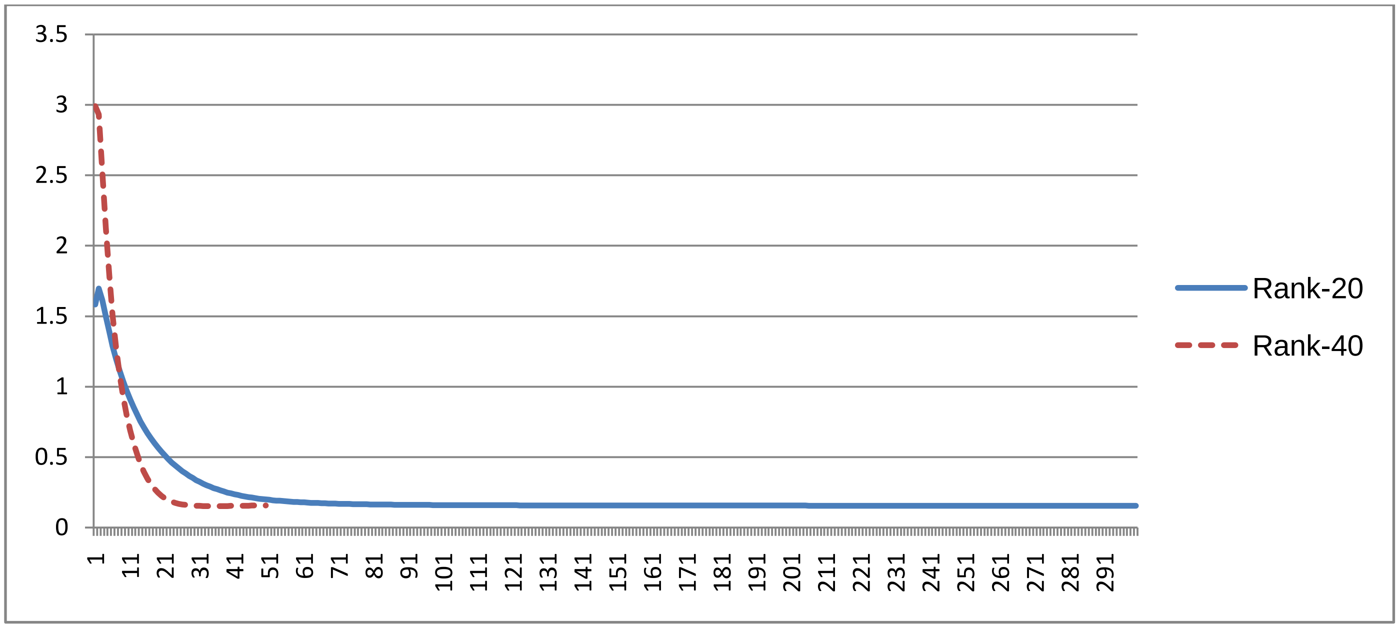


Fig7a) 100K dataset convergence on ranks 20 and 40

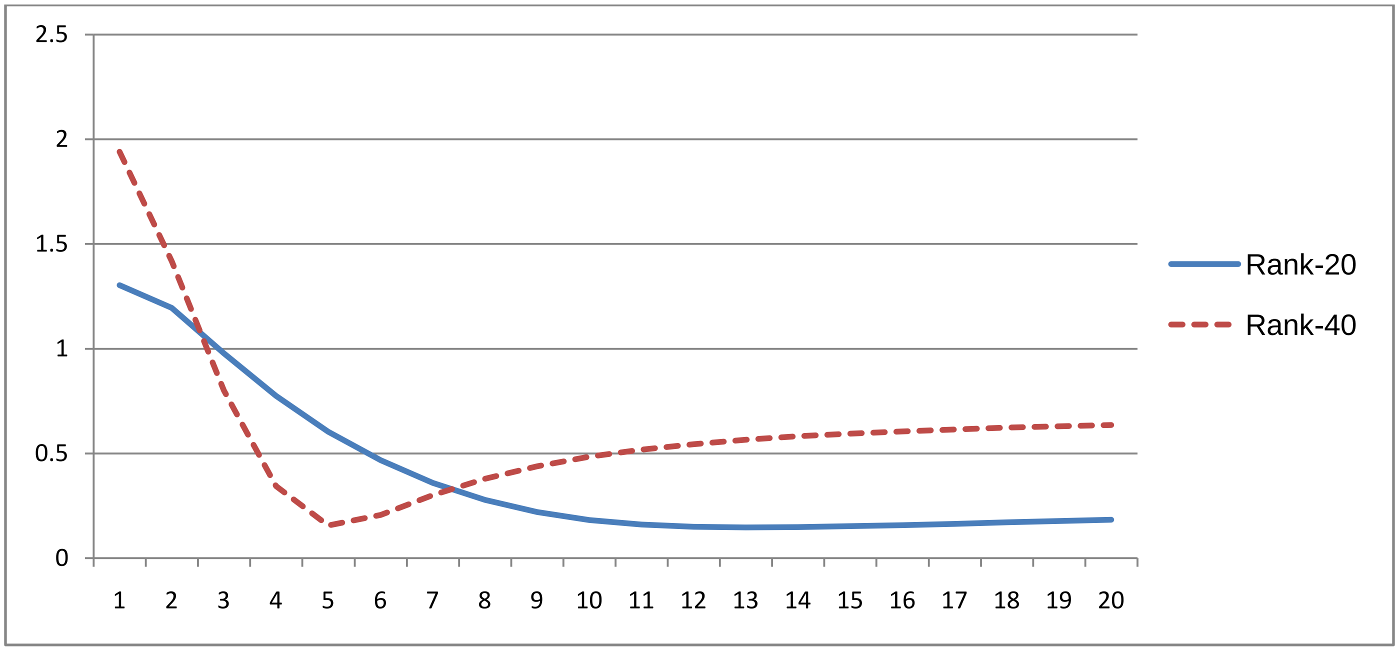


Fig7b) 1M dataset convergence on ranks 20 and 40

- Comparison of convergence behavior of proposed algorithm on different regularization parameters with rank 20 and q=1.8, p=0.4 (**λ_1 =10^{-4}, λ_2= 10^{-5} provides better and faster convergence**)

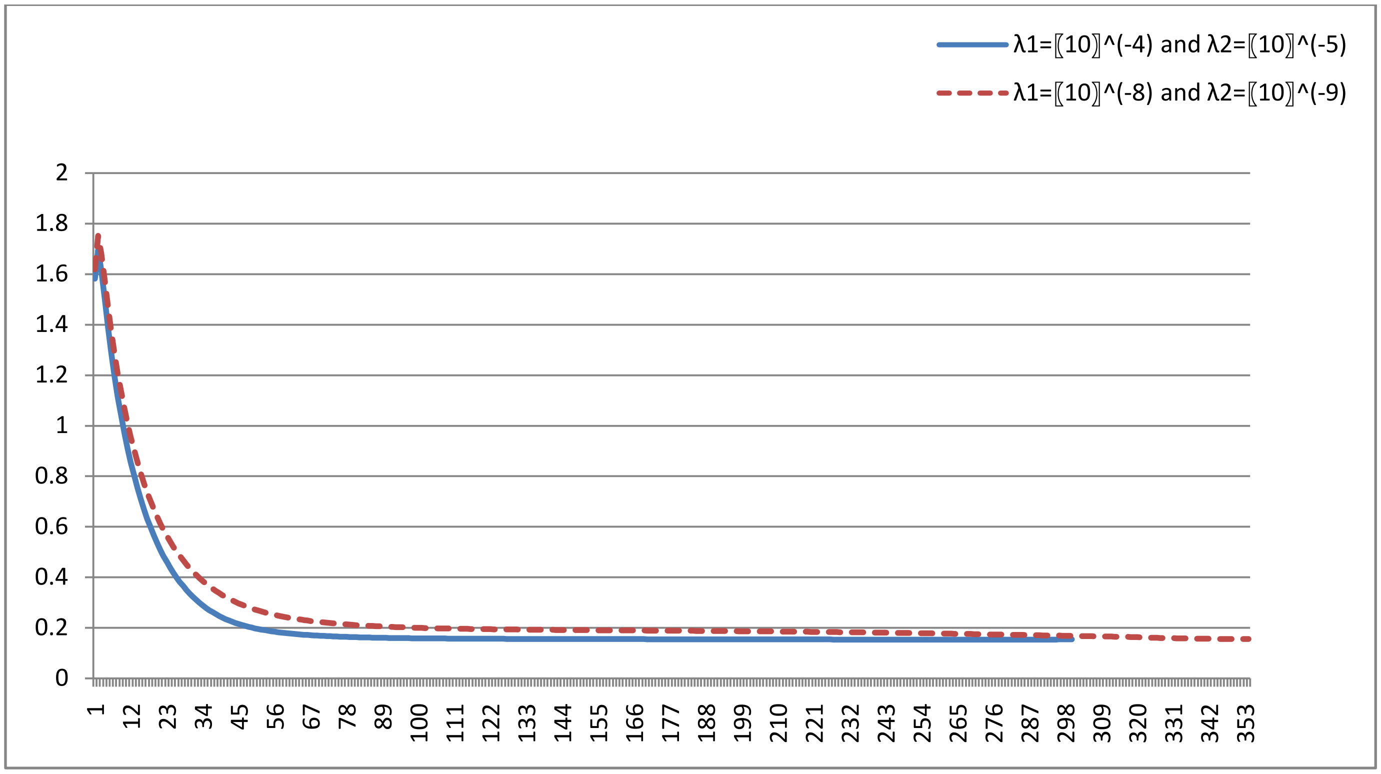


Fig8 100K dataset convergence on different regularization parameters

5.2 DCE-MRI

Here we work on the Dynamic Contrast Enhanced MRI dataset on female diabetic/severe non-obese and immune deficient mice bearing a tumor.

DCE MRI enables the analysis of blood vessels generated by tumor or local-inflammation using an injection of contrast agent. It represents the concentration of contrast agent going in and out of the tissue.

The data was acquired on a 7T/30 cm bore MRI scanner at Bruker, Germany. It has a matrix/image size of 128*64 pixels and 1200 temporal frames with 2.24 seconds of time difference between two frames. The dataset 2D DCE1 was taken from mouse bearing HCT-116 (human colorectal carcinoma) tumor.

As 1200 frames is a lot of data to process, in our experiments, one-sixth of it i.e. 200 frames are considered. In this data, the signal is assumed to be sparse in terms of the learned basis, hence, making possible for the signal to be reconstructed from fewer measurements.

For solving l_1-minimization here (depicting the sparseness constraint), we use our proposed algorithm which is a combination of IRLS, Landweber and SGD. The value of parameter α, for landweber iteration, is fixed to 20 which adhere to its constraint of $\propto \geq \max(eig(A * A^T)$. and $\max(eig(A * A^T)$ is evaluated using lansvd in this experiment.

All methods require the specification of the regularization parameters λ_1 and λ_2 as they are responsible for taking into account the relative importance of the sparsity. But unfortunately, these parameters need to be tuned experimentally. The tuning mechanisms for determining values of these parameters are vaguely described in [40-42], so they are not very clear. Hence, we manually tune them via experimentation. After a series of simulations, these parameters for this dataset are set to $\lambda_1 = 10^{-4}$ and $\lambda_2 = 10^{-5}$. Taking sparsity into account and looking at (27), l1-minimization is to be solved for in this experiment.

The error metric used is NMSE (Normalized Mean square error) as discussed in (28).

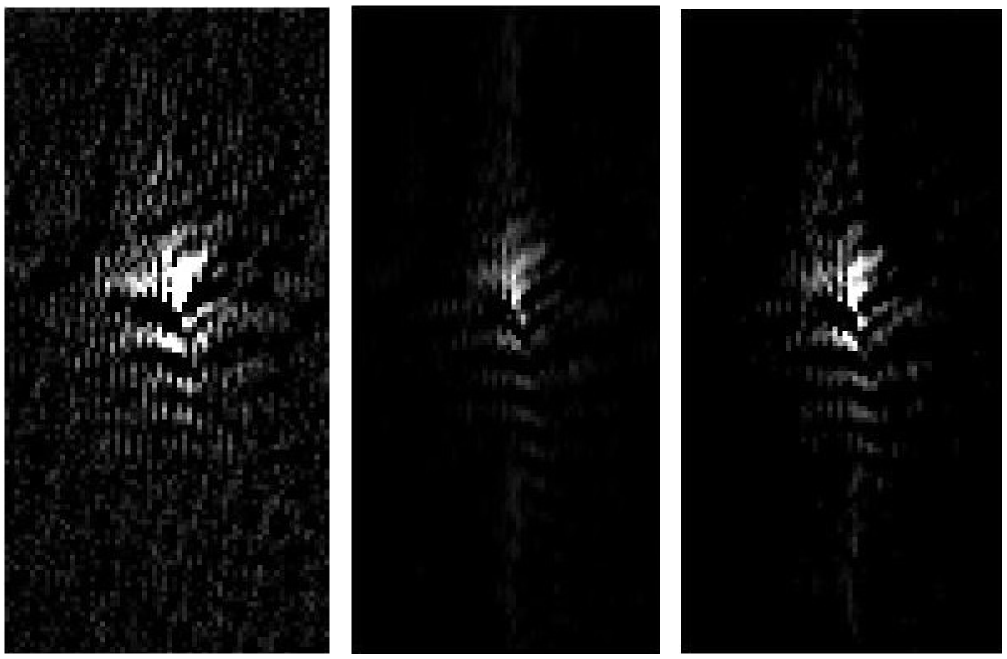
5.2.1 Results for DCE-MRI

	NMSE	Time(seconds)
Sparse l_1 penalty	0.3060	1110.824
Dense l_2 penalty	0.3083	583.188

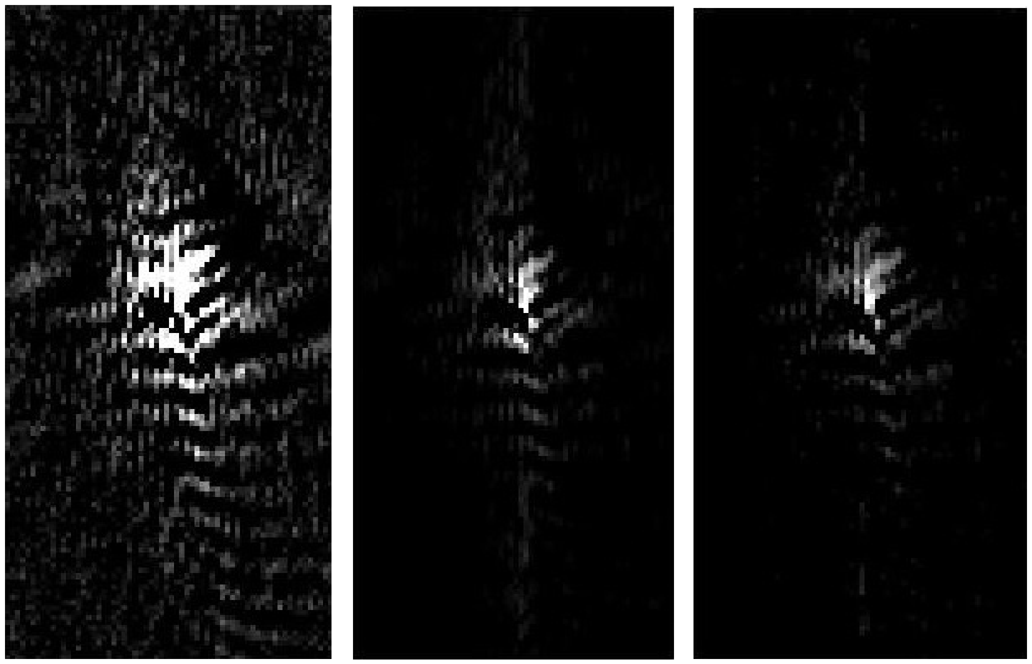
Table 3 DCE-MRI reconstruction Results

These results are observed with rank of the matrix as 20. They signify that although, there is no significant enhancement in the accuracy of MR image reconstruction, this approach of considering the sparsity provides a good alternative of reconstruction with same accuracy using fewer measurements.

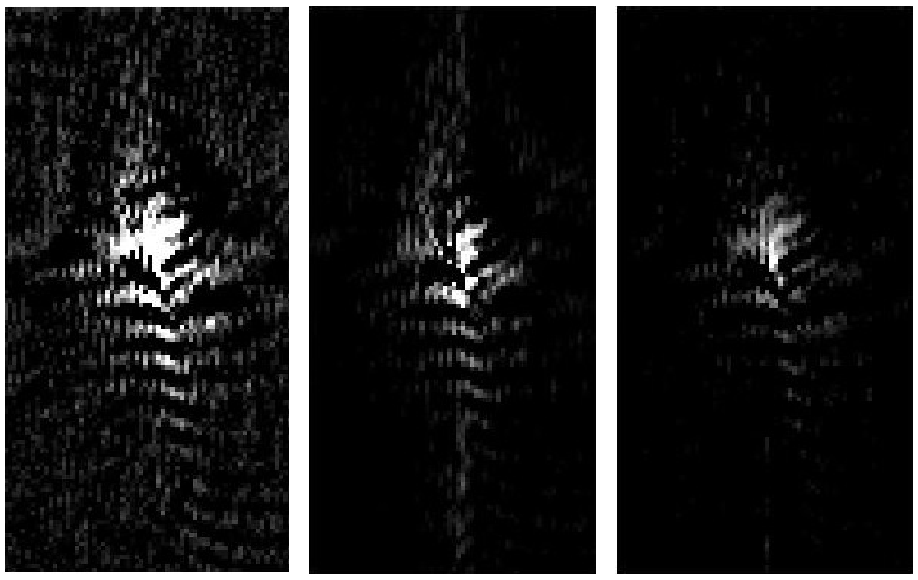
5.2.2 Reconstructed MR images



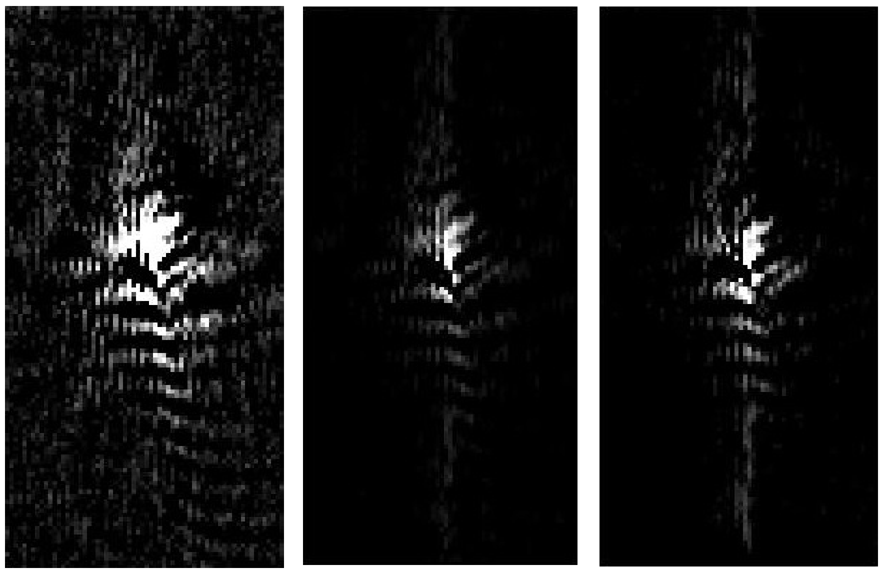
2nd time frame from dataset



75th time frame from dataset



150th time frame from dataset



195th time frame from dataset

Fig9 Reconstructed Images of 2D-DCE1 dataset from Left to Right: Original Image, l2-penalty reconstructed image, l1-penalty reconstructed image

Visual evaluations show that l1-penalty reconstruction proves out to be as better as l2-penalty reconstruction even with fewer measurements.

Chapter 6

Conclusion

This thesis presents an efficient three phase algorithm (IRLS, Landweber, Stochastic gradient descent) to reconstruct solution matrices from lower dimensional projections via matrix factorization. It adheres to low rank matrix completion problem and focuses on the sparsity attribute of the factorized part of the matrix i.e. the latent factor matrix, which help generate better results compared to when dense factorization is considered.

Analytically, for both CF and MRI it explains the reason why l_1-norm or l_p-norm penalties would result in better results as compared to standard approach of l_2-norm penalty.

Also, empirically, after using the proposed algorithm and our proposed approach of enforcing l_p-norm penalty on item latent factor matrix in CF, it leads into significant improvement in results compared to l_2-norm, as computed via NMAE (Normalized mean absolute error). In case of MRI, although, enforcing l_1-norm penalty on coefficient matrix doesn't lead into better results compared to l_2-norm penalty, as computed via NMSE (Normalized Mean squared error), it still is able to give approximately equivalent reconstruction, which shows a better alternative as equivalent results are obtained using less number of measurements which is time effective in context of data collection.

Bibliography

[1] Srebro, Nathan. Learning with matrix factorizations. Diss. Massachusetts Institute of Technology, 2004.

[2] Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009): 30-37.

[3] Eriksson, Anders, and Anton Van Den Hengel. "Efficient computation of robust low-rank matrix approximations in the presence of missing data using the L 1 norm." Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, 2010.

[4] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry", Communications of ACM, vol. 35, no. 12, pp. 61-70, 1992.

[5] K. GoldBerg, T. Roeder, D.gupta, and C.Perkins, "Eigen-taste: a constant time collaborative filtering algorithm", Information retrieval, vol.4, no. 2, pp 133-151, 2001.

[6] Miller, Bradley N., Joseph A. Konstan, and John Riedl. "PocketLens: Toward a personal recommender system." ACM Transactions on Information Systems (TOIS) 22.3 (2004): 437-476.

[7] Claypool, Mark, et al. "Implicit interest indicators." Proceedings of the 6th international conference on Intelligent user interfaces. ACM, 2001.

[8] Su, Xiaoyuan, and Taghi M. Khoshgoftaar. "A survey of collaborative filtering techniques." Advances in artificial intelligence 2009 (2009): 4.

[9] Herlocker, Jonathan L., et al. "An algorithmic framework for performing collaborative filtering." Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 1999.

[10] Sarwar, Badrul, et al. "Item-based collaborative filtering recommendation algorithms." Proceedings of the 10th international conference on World Wide Web. ACM, 2001.

[11] Resnick, Paul, et al. "GroupLens: an open architecture for collaborative filtering of netnews." Proceedings of the 1994 ACM conference on Computer supported cooperative work. ACM, 1994.

[12] Breese, John S., David Heckerman, and Carl Kadie. "Empirical analysis of predictive algorithms for collaborative filtering." Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc., 1998.

[13] Basu, Chumki, Haym Hirsh, and William Cohen. "Recommendation as classification: Using social and content-based information in recommendation." AAAI/IAAI. 1998.

[14] Miyahara, Koji, and Michael J. Pazzani. "Improvement of Collaborative Filtering with the Simple Bayesian Classifier 1." (2002).

[15] O'Connor, Mark, and Jon Herlocker. "Clustering items for collaborative filtering." Proceedings of the ACM SIGIR workshop on recommender systems. Vol. 128. UC Berkeley, 1999.

[16] J. Canny, "Collaborative filtering with privacy via factor analysis," in Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 238–245, Tampere, Finland, August 2002.

[17] Vucetic, Slobodan, and Zoran Obradovic. "Collaborative filtering using a regression-based approach." Knowledge and Information Systems 7.1 (2005): 1-22.

[18] Lemire, Daniel, and Anna Maclachlan. "Slope One Predictors for Online Rating-Based Collaborative Filtering." SDM. Vol. 5. 2005.

[19] Adomavicius, Gediminas, and Alexander Tuzhilin. "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions." Knowledge and Data Engineering, IEEE Transactions on 17.6 (2005): 734-749.

[20] Schein, Andrew I., et al. "Methods and metrics for cold-start recommendations." Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2002.

[21] Sarwar, Badrul, et al. "Incremental singular value decomposition algorithms for highly scalable recommender systems." Fifth International Conference on Computer and Information Science. 2002.

[22] Berry, Michael W., Susan T. Dumais, and Gavin W. O'Brien. "Using linear algebra for intelligent information retrieval." SIAM review 37.4 (1995): 573-595.

[23] T. Hofmann, "Latent Semantic Models for Collaborative Filtering", ACM Transactions onInformation Systems 22 (2004), 89–115.

[24] Zhang, Sheng, et al. "Using singular value decomposition approximation for collaborative filtering." E-Commerce Technology, 2005. CEC 2005. Seventh IEEE International Conference on. IEEE, 2005.

[25] Paterek, Arkadiusz. "Improving regularized singular value decomposition for collaborative filtering." Proceedings of KDD cup and workshop. Vol. 2007. 2007.

[26] Candès, Emmanuel J., and Terence Tao. "The power of convex relaxation: Near-optimal matrix completion." Information Theory, IEEE Transactions on 56.5 (2010): 2053-2080.

[27] Candes, Emmanuel J., and Yaniv Plan. "Matrix completion with noise." Proceedings of the IEEE 98.6 (2010): 925-936.

[28] Koren, Yehuda, and Robert Bell. "Advances in collaborative filtering." Recommender Systems Handbook. Springer US, 2011. 145-186.

[29] Lustig, Michael, David Donoho, and John M. Pauly. "Sparse MRI: The application of compressed sensing for rapid MR imaging." Magnetic resonance in medicine 58.6 (2007): 1182-1195.

[30] Trzasko, Joshua, and Armando Manduca. "Highly undersampled magnetic resonance image reconstruction via homotopic-minimization." Medical imaging, iEEE Transactions on 28.1 (2009): 106-121.

[31] Gamper, Urs, Peter Boesiger, and Sebastian Kozerke. "Compressed sensing in dynamic MRI." Magnetic Resonance in Medicine 59.2 (2008): 365-373.

[32] Lustig, Michael, et al. "kt SPARSE: High frame rate dynamic MRI exploiting spatio-temporal sparsity." Proceedings of the 13th Annual Meeting of ISMRM, Seattle. Vol. 2420. 2006.

[33] Jung H, Ye JC, Kim EY. Improved k-t BLAST and k-t SENSE using FOCUSS. Phys Med Biol 2007;52:3201-26.

[34] Jung, H., Park, J., Yoo, J., & Ye, J. C. (2010). Radial k-t FOCUSS for high-resolution cardiac cine MRI. Magnetic Resonance in Medicine, 63(1), 68-78.

[35] Jung, H., Sung, K., Nayak, K. S., Kim, E. Y., & Ye, J. C. (2009). k-t FOCUSS: A general compressed sensing framework for high resolution dynamic MRI. Magnetic Resonance in Medicine, 61(1), 103-116.

[36] Chen, Liyong, Matthias C. Schabel, and Edward VR DiBella. "Reconstruction of dynamic contrast enhanced magnetic resonance imaging of the breast with temporal constraints." Magnetic resonance imaging 28.5 (2010): 637-645.

[37] Adluru, Ganesh, et al. "Acquisition and reconstruction of undersampled radial data for myocardial perfusion magnetic resonance imaging." Journal of Magnetic Resonance Imaging 29.2 (2009): 466-473.

[38] Zhao, Bo, et al. "Low rank matrix recovery for real-time cardiac MRI." Biomedical Imaging: From Nano to Macro, 2010 IEEE International Symposium on. IEEE, 2010.

[39] Haldar, Justin P., and Zhi-Pei Liang. "Low-rank approximations for dynamic imaging." Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on. IEEE, 2011.

[40] Lingala, Sajan Goud, et al. "Accelerated dynamic MRI exploiting sparsity and low-rank structure: kt SLR." Medical Imaging, IEEE Transactions on 30.5 (2011): 1042-1054.

[41] Lingala, Sajan Goud, et al. "Accelerated first pass cardiac perfusion MRI using improved k−t SLR." Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on. IEEE, 2011.

[42] Zhao, Bo, et al. "Image Reconstruction From Highly Undersampled-Space Data With Joint Partial Separability and Sparsity Constraints." Medical Imaging, IEEE Transactions on 31.9 (2012): 1809-1820.

[43] Koren, Yehuda, and Robert Bell. "Advances in collaborative filtering." Recommender Systems Handbook. Springer US, 2011. 145-186.