

**Scalable Data Science**  
**Prof. Anirban Dasgupta**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Gandhinagar**

**Lecture – 15 b**  
**QB Decomposition**

Welcome to the course on Scalable Data Science. I am Anirban, I am from IIT Gandhinagar. Today's lecture is number 15 b on QB Decomposition. Today we will talk about different kinds of dimension reduction that are applicable for large datasets ok.

(Refer Slide Time: 00:30)

**Application of dimension reduction**

Recommendation system

movies

users

$A_{ij}$

$X$

topic

Y

topic

movie

romance, drama

action, movie

Minimize some loss function, e.g.  $\sum_i (A_{ij} - X_{ik} Y_{kj})^2$

Use  $\sum_k X_{ik} Y_{kj}$  to predict missing entry  $(i, j)$

IIT Gandhinagar  
Indian Institute of Technology Gandhinagar

NPTEL ONLINE  
CERTIFICATION COURSES

So, before we go into our definitions what are the some algorithms are dimension reduction that we are looking into, let us motivate the question of dimension reduction from the point of view of machine learning applications right. So, as you probably know that recommendation systems building recommendation systems is a big area of machine learning right. And we have mentioned that if we look at sites like Amazon, Netflix, Flipkart, even Google all of these have I mean generate a huge amount of revenue by being able to recommend the right product to the right person right, the appropriate product to the right person.

So, imagine a particular recommendation system. Let us suppose we have we have data that consists of users how a bunch of users rated a bunch of movies ok. So, this is for instance the data that Amazon prime has collected ok. And each entry in this matrix each

entry in this matrix maybe is a term that is a score that is based on how a particular user may be rated a movie, how much time a particular user spends seeing a movie and so on and so forth. So, we are not gone going to the details of how this particular data, how this particular matrix has been filled up.

The big question is that that this matrix is pretty big right because potentially there are hundreds of thousands of users, and similarly there are hundreds and thousands of movies ok. So, but the matrix is also very sparse because each user probably has not seen more than what, 100, 200 movies even if even though she might be a movie buff right. And so every row potentially has every row which is of size the number of movies which is let us say 200000 movies has only 200 entries potentially right. And now this is the data that we have and we might have some additional data on the user profile and the movies and the other movies, but let us disregard from that from now right.

The question is that based on this user movie data what we have to see is that user for whom we have some data about let us say let us say this particular user and we know that she has liked these movies she has not liked these particular movies and so on and so forth right. And we have some rating on these on these movies my decision. So, what should I suggest this user right, what movie should I just for recommend this user to sort of see right.

And you can potentially imagine that the that the that the better the recommendation system the more the users are likely to come back to the site right, because it takes time to search through big to search the movies of the site. So, if I can actually recommend the user right that this is the movie that that I think that they are going to like best, the user is likely to users enjoyment is likely to increase and she is likely to come back to the site a again and again.

So, this is what we really want right that we want to really sort of try to predict that what is the movie that the user is going to like best, ok? So, the way this is formalized is that we kind of imagine that this particular data, the user movie data has some structure, because otherwise we cannot hope to learn a model right we cannot hope to predict the missing entries from a model ok.

So, what is the kind of structure? So, here is one typical assumption about the structure. We assume that the users have preferences right. Let us say that the let us say that there

are different genres, it say that that one of the genre could be romance, one of the genre could be action, one of the genre could be drama and so on and so forth. And the users suppose have different preferences about these genres right that maybe I like to see drama movies 30 percent of the time I like to see I have so much preference of horror drama movies, have so much preferences of action movies and so much of romance movies ok.

Similarly, the movies can also be characterized in terms of belonging to genre or a mixture of genres ok. So, So, what this kind of implies what this gives an intuition of is that maybe there is a low rank structure in this data right. So, how exactly the low rank structure manifests itself in generating the data is a question about the generative model. And we are not going into the details of it right, but at the end of the generative model the optimization problem for learning the model typically boils down to this.

That we want to decompose the existing user movie data in terms of the product of two low rank matrices right; one that we call the user versus topic right, and the other is the topic versus the movie right. I am writing topic in quotes because I am not guaranteeing in this in this particular module in any such model that these topics are interpretable in the sense that these topics correspond to that the real movie topics that you understand. One of these topics might be a mixture of many of the of the of the human understandable movie genres. For instance right, but nevertheless right that might not be so important this a user interpretability might not be so important from the point of view of being able to predict well right.

In terms of the model, what we are really saying now is that lets say that that  $A_{ij}$  right is the is the entry is the value of the or the rating that the  $i$ th user gave on a  $j$ th movie. This now right we are writing it we are trying to model it as a product of two things. We are trying to model it as a distribution of the user  $i$  over the different genres which is given by this by this particular by the  $i$ th row of the matrix  $X$  this is let me call this the matrix  $X$ , let me call this the matrix  $Y$ . So, then the entry  $A_{ij}$  right we try to model it as the product of the  $i$ th row of the matrix  $X$ , and the  $j$ th column of the matrix  $Y$  right. So, the  $j$ th column of the matrix  $Y$  corresponds to in some sense the add a sort of description of the movie in terms of the hidden topics or hidden genres ok.

And now what we are trying to find is that what is the best  $X$  and  $Y$ , what are the best low rank  $X$  and  $Y$  right that explain the data right or rather what are the  $X$  and  $Y$  find out  $X$  and  $Y$  that are constrained to be low rank such that the best explain the data right. And how do we measure best typically it is in terms of norm of a matrix norm right of some you have to define some loss function. One of the popular loss functions is that you take a sum over all the  $i$  mean overall the observed entries right. So, the sum is over all the observed entries. You take the  $A_{ij}$  which is no observed entry and then you take  $X$  the the  $i$  mean recall that  $X_i$  star is the is the row vector which is the  $i$ th row of  $X$  and  $X_i$   $Y_j$  star is a is a is a column vector that is the  $j$ th row of  $Y$  right. So, I am trying to represent  $A_{ij}$  by  $X_i$  star dotted with  $Y_j$  star right. So, this is an inner product.

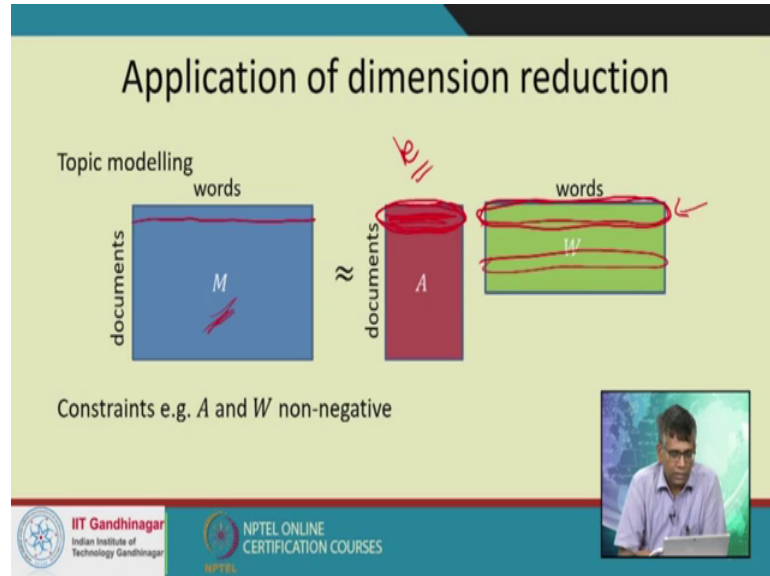
And then we take the  $l_2$  square loss and then we sum it up right. So, this we typically sum it up, we could sum it up over all the observed entries right. Or we could in that case the problem is not really is not really a nice linear algebraic problem, we have to resort to optimization problems. We will sometimes do something simpler in the sense that will forget about the missing entries, we will assume that the missing entries are filled in by some constant or some let us say 0 or even and then will just sum this up over all over all this over all the entries over all the  $i$  and  $j$  in the matrix  $A$  ok.

So, now so now if the if the entry  $A_{ij}$  happens to be missing right, then after we have learned  $X$  and  $Y$ . So, first we so first we take the training set a the training values the set of values that are provided to me in the matrix  $A_{ij}$  and try to find out the  $X_i$  and  $Y_j$  that that sort of minimize this quantity right. And we might even assume that that let us say we might even assume and for that we might need to assume something about the number of topics  $k$  right or we might resort to some kind of regularization. But once you have done that, then in order to predict a missing entry  $A_{ij}$  right, I will take the corresponding dot product of the row  $i$  and the column  $j$  and that will be my prediction for the entry  $ij$  right.

And if I want to take the and if I want to take the highest rated movie, highest potentially rated movie for this user, then for the user I will calculate this for all possible  $j$  and I will find out which is the  $j$  that is the maximum that the user has not seen. So, find out  $A_j$  that the user that is that is that the user has not seen right, find out the movie that the user has not seen and that has the highest predicted value among all these among all such unseen movies right, this is what we would do in the recommendation system. So, as you can see

that at the heart of it comes down to finding the solution of this optimization problem right,  $A_{ij}$  minus the product of two low rank matrices ok.

(Refer Slide Time: 10:09)



A similar application can be seen in modelling right. In topic modelling for instance we have a set of documents, and each of the documents is being represented as a vector over the set of words ok. And now what we want to say is that we want to understand what are the underlying topics ok. So, again what we do is that we say that that suppose there are  $k$  hidden topics, they may or may not be interpretable right.

And the way we find out what are these, what are the best key topics is by trying to decompose the document word matrix  $M$  in terms of the product as a product of two low rank matrices right. One of them is the document versus topics; the other is the topics versus words ok. And what and what this kind of means is that each document this kind of also posits the I mean also a series of the generative model here is something like that each document right is I mean chooses each word is being distributed each topic has been distributed as a as a distribution over words here.

So, this is a topic you can imagine that this is a distribution over words. Assume that this  $W$  is suitably normalized and so on. And now each document first chooses a distribution of a topics that this document says that maybe I am going to be a 80 percent at politics and 20 percent about sports. And then it sort of mixes the corresponding topics right according to these weights according to the weights out here right, and then and then it

generates the words and then it samples the words. So, again I am not going to the details of the generative model, but the basic point that I want to stress, I want to put forward is that that learning the topic distribution modelling the topic distribution in a in from a from a document word corpus essentially boils down to doing a low rank often boils down to doing a low rank factorization of the matrix  $M$ , which is the document word matrix right.

And depending on the particular application that whether we are doing a recommendation system or whether we are doing a topic modelling the we might want to put in different kind of constraints on the on the matrices. We might want to change the last function in one case it might be it might be the  $\ell_2$  I mean  $\ell_2$  square the sum of squares; in the other case it might have to do something with some other divergence right. In the topic modelling case for instance we typically put in non-negative, a non-negative constraint on the matrices  $u$  and  $w$  right. So, this is typically falls on a non-negative matrix factorization right.

(Refer Slide Time: 12:37)

**Singular Value Decomposition**

- $A = U\Sigma V^t$
- Optimality properties e.g.
  - $A_k = U_k \Sigma_k V_k^t = \operatorname{argmin}_{X: \operatorname{rank}(X) \leq k} \|A - X\|_F$
  - $A_k = U_k \Sigma_k V_k^t = \operatorname{argmin}_{X: \operatorname{rank}(X) \leq k} \|A - X\|_2$

Issues: high complexity, negative  $U, V$  etc. *missing values*

Yet often forms the initialization for other matrix factorizations

*Handwritten notes on slide:* Three boxes containing  $U$ ,  $\Sigma$ , and  $V^t$  are drawn. A large red box contains the formula  $\sqrt{\sum_{i,j} (A-X)_{ij}^2}$ .

**IIT Gandhinagar**  
Indian Institute of Technology Gandhinagar

**NPTEL ONLINE**  
CERTIFICATION COURSES

So, one of that one of the matrix factorizations, so then so then where do we whether the I mean where do we come in right where does the scalable data sense coming. Now, the question is that how do we once we have come up with such a model of the data, how do we do this matrix factorizations, because typically the documents the matrix that will be given to you either the either the user movie matrix, the document word matrix will be

pretty large right. And it is it is very hard and doing matrix factorization is going to be a non-trivial question ok.

So, one of the one of the starting points right in trying to do such matrix factorization is actually the singular value decomposition that we have already seen right. And the reason it sort of and why it does not satisfy a lot of the constraints that I mentioned before for instance it has it does not satisfy non-negativity constraints it cannot deal with missing values and so on and so forth. I mean it does not certain very nice properties right. So, missing values let me write down missing values here. It cannot deal with missing values ok. But it has a very nice property.

For instance, as you can see as we have mentioned that I mean once I write down the on the matrix  $A$  as a product of as a product of three matrices  $U$ ,  $\Sigma$  and  $V^T$  right. If I really want to get the optimal low rank approximation to the matrix  $A$  right in I mean for instance in terms of the Frobenius norm that is if I want to solve this particular question right that I want to get the matrix  $X$  such that  $X$  is a rank at most  $k$  right and the Frobenius norm that is the sum of sum of  $\sum_{ij} (A_{ij} - X_{ij})^2$  and the summation over  $ij$  square root this if you recall is the Frobenius norm that.

And if you want to minimize this norm, if you want to minimize this error under the constraint that the  $X$  is of rank  $k$  rank at most  $k$ , then the optimal such matrix  $X$  is obtained can be obtained by doing the from; by looking at the singular value decomposition. Essentially, what you do is that you find out the singular value decomposition right you keep only the top  $k$  singular values and then you keep the corresponding singular vectors on the left and the right hand side that gives you the matrix  $A_k$ .

And interestingly enough this also if you change this norm to be something like the two norm right, which is the which is the spectral norm of the matrix of the matrix  $A - X$  the same matrix  $A_k$  also serves as the optimal as the as a optimal solution of that ok. So, because of this because of such nice reasons, singular value decomposition often starts as a starting point of an act as a starting point for such for more complicated optimization problems ok, for more complicated matrix factorization problems.

(Refer Slide Time: 15:28)

### Alternate look at SVD

A

≈

U<sub>k</sub>

X

$A_k = U_k X$   
 $= U_k U_k^T A$

$$X = \Sigma_k V_k^t = U_k^t A$$

Can find a similar decomposition, but more efficiently?  
 Should retain (approximately) the optimality properties

Optimal Frob error =  $\|A - U_k(U_k^t A)\|_F^2 = \sum_{j>k} \sigma_j^2$

So, let us try to look at try to do SVD in a I mean another problem, but the problem behind SVD is that it is a it also has very high complexity.

(Refer Slide Time: 15:36)

### Singular Value Decomposition

- $A = U\Sigma V^t$
- Optimality properties e.g.

$A_k = U_k \Sigma_k V_k^t = \underset{X: \text{rank}(X) \leq k}{\text{argmin}} \|A - X\|_F$   
 $A_k = U_k \Sigma_k V_k^t = \underset{X: \text{rank}(X) \leq k}{\text{argmin}} \|A - X\|_2$

Issues: high complexity, negative U, V etc. *missing values*

Yet often forms the initialization for other matrix factorizations

Right in the sense that it sort of takes time order  $nd$  mean of  $nd$  right, which is like cubic right mean of  $n$ ,  $n$  and  $d$ , which is almost cubic in  $n$  ok. And the question is that can we do faster right because this is not I mean given the size over data this we have to mention is not scalable. So, for that let us to get an alternate look at SVD, right.



So, one way of looking at SVD is that let us look at only the only the matrix  $U$  right. So, the matrix  $U$  right or for instance if I take the first  $k$  singular vectors of  $U$ , and call it  $U_k$  that forms a basis I mean  $U$  itself forms the basis of the column space of  $A$  and therefore, and  $U_k$  forms the basis of the column space of  $A_k$  which is the optimal rank approximation right. So, therefore, one way of looking at the matrix  $A_k$  is that it can be written as  $U_k$  times  $X$  right where  $X$  really you can write it either a  $\Sigma_k V_k^T$  or you can equivalently write it as  $X = U_k^T A$ . Basically  $X$  is then the projection of  $A$  onto the top  $k$  singular vectors  $U_k$ . So, then so then I mean  $A_k$  equivalently can be written as  $U_k U_k^T A$ .

And this if you recall, it is actually projection matrix ok. So, the question now is that can we find such a similar decomposition, but more efficiently right and we wish that it should be written approximately at least optimality properties that the singular value decomposition has right. So, just to sort of refresh your memory, the optimal Frobenius norm that is the optimal Frobenius error. And let us look at the Frobenius error for now is the one that is achieved by  $A_k$  that is given by  $\|A - A_k\|_F$  which is a  $\|U_k U_k^T A - A\|_F$  which is given by the sum of the singular values from  $k+1$  to whatever the rank of the matrix is right from  $k+1$  to  $d$  or  $1$  to  $r$  lets if  $r$  is the rank of the matrix, how much the singular values square ok.

(Refer Slide Time: 17:46)

**QB decomposition**

$n \times d$   $A \approx \begin{matrix} n \times (k+p) \\ Q \end{matrix} \begin{matrix} (k+p) \times d \\ B \end{matrix}$

$B = Q^T A$   
 $QB = Q Q^T A$

Can we find orthogonal  $Q \in \mathbb{R}^{n \times (k+p)}$  and  $B$  such that

$\|A - QB\|_F \approx \|A - A_k\|_F$  and/or  $\|A - QB\|_2 \approx \|A - A_k\|_2$

Also want  $k + p = O(k)$

So, what will do, so what will try to say is that we call I mean we will try to come up with a decomposition that is typically known as the QB decomposition right. And this decomposition will look like look as follows. That given a matrix  $A$  right. We will try to come up with a matrix  $Q$  lets say  $A$  is  $n$  by  $d$ , we will try to come up with a matrix  $Q$ , and  $Q$  will be  $n$  by  $k$  plus  $p$  and a matrix  $B$  and  $B$  will be  $k$  plus  $p$  by  $d$  right such that the error by obtained by the error I mean then this.

So, first of all if you it should be easy to see that  $QB$  is a low rank approximation of the matrix  $A$  right, because the rank of  $QB$  is not more than  $k$  plus  $p$ . And what we are going to say is that the error a the Frobenius error  $A$  minus  $QB$  is not much more than the optimal error  $A$  minus  $A_k$  right its of that it is it is it is bounded by that right. All we want to say that maybe the spectral error of  $A$  minus  $QB$  is not much more than the spectral error  $A$  minus  $A_k$ . You remember that these two are the optimal spectral and the Frobenius errors right.

And furthermore we also want that this that the that the that the size of  $Q$  right is not much more than  $k$  right. And when I mean I am writing it can to be  $k$  plus  $p$  for obvious reasons that will come out later right, but basically I want this to be of the order of  $k$  right. And we are looking at  $Q$  to be an orthogonal matrix right. So, think of  $Q$  as a replacement for a  $U_k$  right, which is the top  $k$  singular vectors of the matrix.

And once you have found out such a  $Q$  it should be fairly obvious that the optimal  $B$  to be to be used is really  $Q^T A$  right. So, therefore, this  $QB$  decomposition of the matrix, we are also writing it as  $Q Q^T A$  right, which is essentially the projection of  $A$  onto the space spanned by the columns of  $Q$ . Instead of  $U_k$ , we are instead of projecting it on the space spanned by the columns of  $U_k$ ; we are projecting it on the space column spanned by the columns of  $Q$  right. And we want that errors to be not to blow up from what it was before.

(Refer Slide Time: 19:56)

**CX decomposition**

$A \approx CX$

Can we find  $C$ , a subset of columns of  $A$ , such that

$$\|A - CX\|_F \approx \|A - A_k\|_F$$

and  $|C| = O(k)$ ? It is not clear that this even exists!! Useful in ML applications when actual columns, and not their linear combinations, are needed.

IIT Gandhinagar  
Indian Institute of Technology Gandhinagar

NPTEL ONLINE  
CERTIFICATION COURSES

A different decomposition and one that we will look at later is known as the CX decomposition right. Here what will ask is that instead of finding out an orthogonal matrix Q can I actually find out a set of column C right. So, So, these columns, so these so these columns that I that I use in C should actually come from the columns of A right and still can I approximate the Frobenius the optimal Frobenius error while making while choosing only of the order of k columns. And it is not even obvious that this exists as of now right. It is not it is not even obvious you can choose C columns from A and still retain the optimal Frobenius norm

However, we will show that in sort of coming lectures, we will show that it exists; and furthermore it is actually pretty useful this is actually pretty useful decomposition because sometimes the columns of C right they have special meaning right. For instance remember that when we had the when we had the document I mean let us say that the user movie matrix right. Now, now the columns of C are actually movies right. And now if I am finding out I mean a genre decomposition in terms of in terms of by actually choosing movies then these movies are potentially good representations of each other genres. So, the interpretability of the matrix factorization comes back into the picture we that would actually lost when trying to do the singular value decomposition ok.

(Refer Slide Time: 21:19)




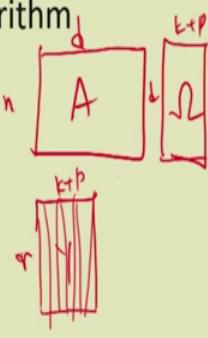
QB: prototype algorithm

Find  $\Omega \in \mathbb{R}^{d \times (k+p)}$  a random matrix

$Y = A \Omega$

Find  $Q$  = orthogonal basis for  $\text{col}(Y)$

Return  $(Q, Q^t A) \rightarrow B$



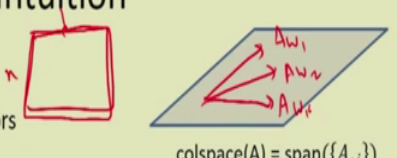
So, we will come to this later. So, here is a prototype algorithm for doing the QB decomposition that is fairly intuitive, but very interesting. So, what we say is that that we have a  $n$  by  $d$  matrix  $A$  right. Let us come with a come up with a  $d$  by  $k$  plus  $p$  matrix  $\Omega$  right. And this is a random matrix. So, how are we generating a random matrix, we will come to this later, but this is a random matrix. So, first we look at the matrix  $Y$  which is a multiplied multiplication of  $A$  times  $\Omega$ , so that means,  $Y$  is of size  $n$  by  $k$  plus  $p$  right which is a pretty thin matrix right.

So, now, doing matrix factorization of  $Y$ , it is actually easier right, because it is a small because it is a smaller matrix. So, what will do is that we will try to get the orthogonal basis for the column space of  $Y$  right. We will try to get the base orthogonal basis for the column space of  $Y$  and we call that  $Q$  right. And then we return the pair  $Q$  and  $Q$  transpose  $A$  right. So,  $Q$  is my sort of this  $Q$  is my candidate for the up for the close to optimal basis for a rank  $A$  decomposition of  $A$ , and therefore,  $Q$  transpose  $A$  is a projection of  $A$  onto that basis right. So, this is the target, this is my  $B$  the  $Q$  transpose  $A$  is my  $B$  ok.

(Refer Slide Time: 22:45)


**Intuition**

Assume  $\text{rank}(A) = k$   
 $\omega_1 \dots \omega_k \in \mathbb{R}^d$  be random vectors



$A\omega_1, A\omega_2, \dots, A\omega_k$  are in general position in column space of  $A$

$\text{colspace}(A) = \text{span}(\{A_{\cdot i}\})$



**IIT Gandhinagar**  
Indian Institute of Technology Gandhinagar

**NPTEL ONLINE**  
CERTIFICATION COURSES

19

So, why does this work, what is an intuition? So, to get the intuition of this let us look at a setting, when let us say  $A$  is of size  $n$  by  $d$  so  $A$  is of size  $n$  by  $d$ , but suppose  $A$  is of size  $n$  by  $d$ , but suppose  $A$  has rank  $k$  right. So, the rank of  $A$  is  $k$  which is less than both  $n$  and  $d$  right. So, imagine let me call this, this, this shaded area the column space of  $A$ , which is the space of the span of the columns of  $A$ . So, suppose  $\omega_1$  to  $\omega_k$  are random vectors right in  $d$  dimension right. So, because it is a random vectors in  $d$  dimension chosen in I mean according to whatever reasonable randomness, reasonable I mean  $d$  dimensions randomness we will come to that later.

The vectors they are likely to be in general position right that none of them will be a linear combination of the other ones. What we can further saying is that vectors  $A\omega_1$  to  $A\omega_k$  right that is if we multiply each of these  $\omega$ s by  $A$  these are also in general position first of all right by using randomness with very high likelihood with very high probability that they are in general position. Furthermore we also know that they are in the whole line the column space of  $A$  right. So, the so the vectors  $A\omega_1$ ,  $A\omega_2$  and up to a  $\omega_k$  or lie in the column space of  $A$ .

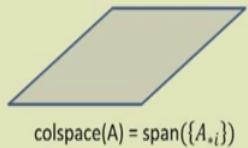
(Refer Slide Time: 24:07)

### Intuition

Assume  $\text{rank}(A) = k$   
 $\omega_1 \dots \omega_k \in \mathbb{R}^d$  be random vectors



$A\omega_1, A\omega_2, \dots, A\omega_k$  are in general position in column space of  $A$

$Q =$  orthogonal basis of  $[A\omega_1 \dots A\omega_k] = A\Omega$  is also basis for  $\text{colspace}(A)$



$\text{colspace}(A) = \text{span}(\{A_i\})$

$Q =$  orthogonal basis of  $[A\omega_1 \dots A\omega_k] = A\Omega$  is also basis for  $\text{colspace}(A)$

So, therefore, if I find out a basis [noise if I find out the orthogonal basis of  $A\omega_1$  to  $A\omega_k$  right that is also a basis for the column space of  $A$  right; and I am calling that  $Q$ . So, any orthogonal basis of  $A\omega_1$  to  $A\omega_k$  is also a basis for the column space of  $A$  ok. So, then in this case it clearly works right that is if the in the rank of  $A$  is  $k$ , then what then the algorithm that I was proposing clearly works.

(Refer Slide Time: 24:32)

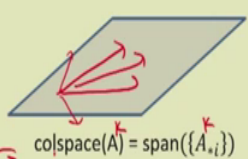
### Intuition

$A = A_k + E, |E|$  small  
 $\omega_1 \dots \omega_k \in \mathbb{R}^d$  be random vectors



$A\omega_i = A_k\omega_i + E\omega_i$

The perturbation  $E$  can take a vector out of the column-space  $A$

But, if we take  $k + p$  vectors, the chance that  $\{A\omega_i\}$  does not span  $\{A_i\}$  is small



$\text{colspace}(A) = \text{span}(\{A_i\})$

What happens at the rank of  $A$  is not  $k$  right. So, imagine that the rank of  $A$  is I mean is let us say I mean either the mean of  $n$  and  $d$ , but suppose  $A$  has the following structure

that  $A$  is a rank  $k$  matrix plus some  $E$  right and think of  $E$  as a small perturbation. So,  $A$  is full rank, but let us say that the norm of  $E$  whatever suitable norm and what norm is appropriate will I mean actually in order to see that you have to do the entire proof which you are not doing right now we are just giving the intuition. So, let us say  $A$  is the full rank matrix, but there is a strong low rank structure this is what you mean.

So, in this case also we take the vectors  $\omega_1$  to  $\omega_k$ , and we multiply this by  $A$  right. So, multiplying the vectors  $\omega_1$  to  $\omega_k$  by  $A$  really it is the same as multiplying taking their products with respect to  $A$ . So, I look at  $A \omega_i$  plus a perturbation  $E \omega_i$  right. So, what can happen now right. So, now, so now, suppose this is the column space of the instead of being a column space of  $A$ , this is the column space of the  $A + E$ . So, what is happening is that  $A \omega_i$  lies in this I mean again for  $A$  each of the each of the vectors  $A \omega_i$ , we can again still argue that they are in general position in a column space of  $A$ . However, this perturbation can take some of these vectors a little bit outside right.

And therefore, it is possible that by taking I mean therefore it is possible that if I look at the I mean if I look at the basis of  $A \omega_1$  to  $A \omega_k$  right maybe it does not completely capture the column the basis for  $A$  right, because of this perturbation  $E$  right. But what we can also show is that I mean if I take a little bit more right there instead of  $k$  if I take  $k + p$  vectors right, then the then this chance the then the chance that the vectors  $A \omega_i$  does not span the column space of  $A$  right is very small right. And this can be made and this can be been mathematical except that it is going to take its take little bit of non trivial mathematics using matrix perturbation theory and so on.

But we can say that, but this is really the intuitive claim that what you can claim is that if the perturbation  $E$  is small with respect to the with the norm of  $A$  right then the chance that then with very good chance then there is a very good chance that if you take  $k + p$  right, the span of the vectors  $A \omega_1$  and until  $A \omega_{k+p}$  spans the column space of  $A$ .

(Refer Slide Time: 27:27)

The slide is titled "Random matrix" in a large, bold, black font. Below the title, the text reads: " $\Omega$  = any JL matrix e.g.  $N(0,1)$  or  $\pm 1$  or FJLT or Sparse JL". The term " $N(0,1)$ " is circled in red, and "FJLT" is underlined in red. Below this, it says "Theoretical bounds are the same, however these differ in numerical stability issues". At the bottom of the slide, it states "Most practical implementations use  $N(0,1)$ ". The slide footer includes the IIT Gandhinagar logo and the text "NPTEL ONLINE CERTIFICATION COURSES".

So and that will and that will again come back to my and then my previous argument again holds right. So, what is this random matrix? I mean talking about a random matrix right,  $\Omega$ . So, it turns out that most random matrix a lot of random matrix is work. Essentially any matrix  $\Omega$  that satisfies FJLT property, for instance if you fill it up by  $N(0, 1)$ , or plus minus 1 or FJLT or even sparse JL. Actually for sparse JL you have to be a little bit careful in terms of the in terms of the I mean the sparsity as well as the size of  $\Omega$ , but basically I mean  $N(0, 1)$ , plus minus 1, FJLT, it really works right.

The theoretical bounds are really the same for most of them more or less the same up to log factors and so on. However, these differ in terms of the numerical stability issues which means that in practice, the practical performance is differs significantly right. And I am really for all for most practical implementations, we use  $\Omega$  as of now, we use  $\Omega$  where every entry is coming from  $N(0, 1)$  right. This is not been enough research on the on the other kinds of and the projection matrices in terms of empirical results ok.



(Refer Slide Time: 28:39)

Theoretical Guarantee

$\|A - A_k\|_F$

Claim:  $E[\|A - QB\|_F] \leq \left(1 + \frac{k}{p-1}\right) \|A - A_k\|_F$

Bounds can also be obtained on the L2 norm error, few different variants of the bound

Qualitative take-away: Need to choose  $p$  to be a small constant, in practice  $\approx 5$

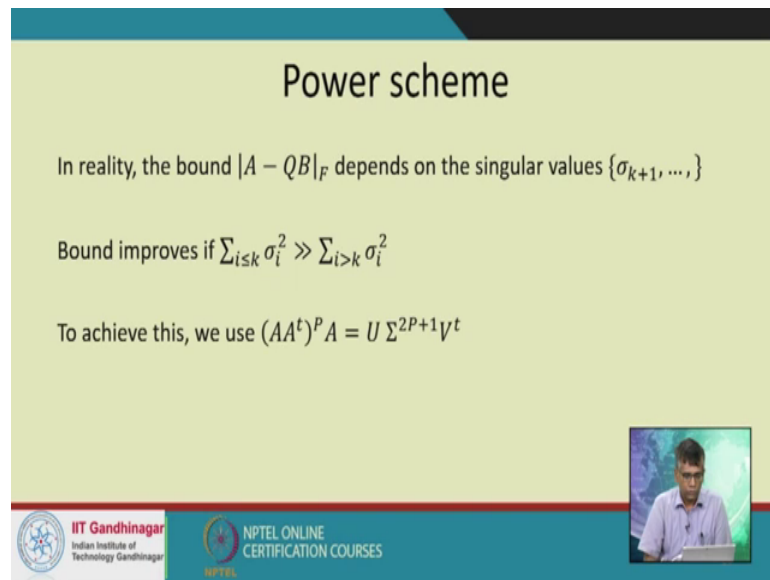
IIT Gandhinagar  
Indian Institute of Technology Gandhinagar

NPTEL ONLINE  
CERTIFICATION COURSES

So, what is the theoretical guarantee, again we are not going to go into the details of the guarantee, but here is essentially the claim that we can say. What we can say that if I look at the Frobenius norm bound of  $A$  minus  $QB$  right. Remember that this Frobenius norm bound is at least  $A$  minus  $A_k$  Frobenius I mean  $A$  minus  $A_k$  Frobenius. And we can show that it is not much more than this  $A$  minus  $A_k$  Frobenius that is it is at most  $1$  plus  $k$  by  $p$  minus  $1$   $A$  minus  $A_k$  Frobenius.

And we can so show similar one for the L2 norm error. There are a few different variants of the bound that you can see up look up from the references other at the end of the lecture. The qualitative takeaway that you should take from this from that you should learn from this particular from this particular bound is that is that we need  $p$  to be a small constant right. If I take  $p$  to be a small constant, in practice about  $5$  or  $10$  is enough, then you get a very good approximation to that to the Frobenius norm bound right. And although this a stated this in terms of the expectation we can easily apply Markov's inequalities and other things right.

(Refer Slide Time: 29:43)



**Power scheme**


In reality, the bound  $|A - QB|_F$  depends on the singular values  $\{\sigma_{k+1}, \dots\}$

Bound improves if  $\sum_{i \leq k} \sigma_i^2 \gg \sum_{i > k} \sigma_i^2$

To achieve this, we use  $(AA^t)^P A = U \Sigma^{2P+1} V^t$

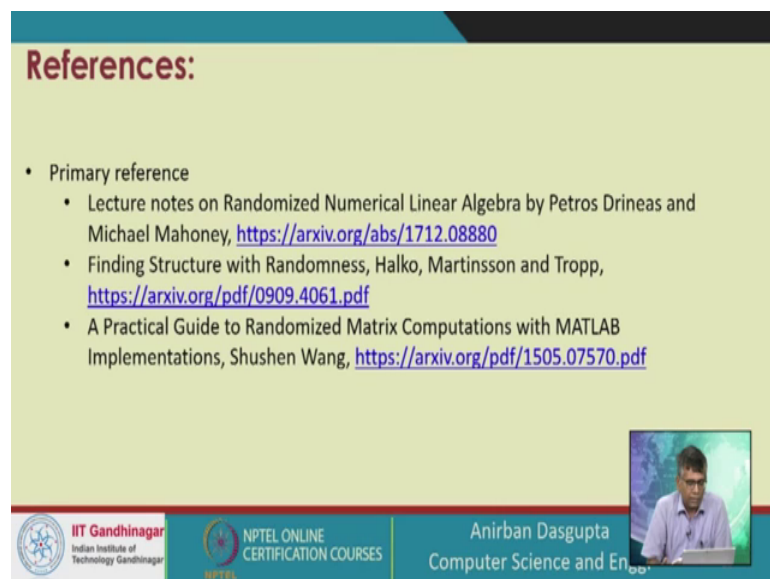
IIT Gandhinagar  
Indian Institute of Technology Gandhinagar

NPTEL ONLINE  
CERTIFICATION COURSES



So, so in fact, there is I mean we can we can modify this algorithm in order to in order to sort of give a better bound. So, let me talk about the modified algorithm in the next slide so just in the next lecture.

(Refer Slide Time: 29:55)




**References:**

- Primary reference
  - Lecture notes on Randomized Numerical Linear Algebra by Petros Drineas and Michael Mahoney, <https://arxiv.org/abs/1712.08880>
  - Finding Structure with Randomness, Halko, Martinsson and Tropp, <https://arxiv.org/pdf/0909.4061.pdf>
  - A Practical Guide to Randomized Matrix Computations with MATLAB Implementations, Shushen Wang, <https://arxiv.org/pdf/1505.07570.pdf>

IIT Gandhinagar  
Indian Institute of Technology Gandhinagar

NPTEL ONLINE  
CERTIFICATION COURSES

Anirban Dasgupta  
Computer Science and Eng



Just to give some references most of this lecture was from these lecture notes from randomized numerical linear algebra by Petros and Mahoney. There is also very nice survey finding structure with randomness by Halko Martinsson and Tropp. And if you are really looking for practical guides with MATLAB code, there is this very nice article

by a Shushen Wang that actually provides you MATLAB code for thought of the matrix factorization and.

Thank you.