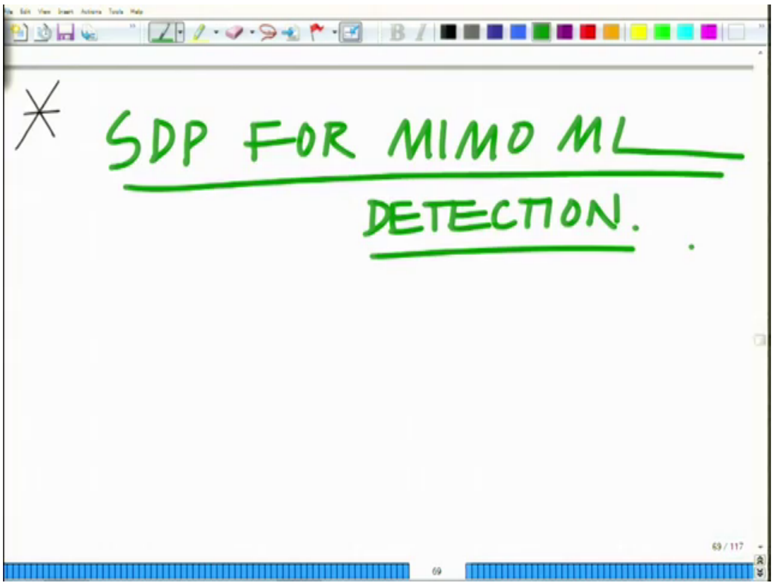
Applied Optimization for Wireless, Machine Learning, Big Data
Prof. Aditya K. Jagannatham
Department of Electrical Engineering
Indian Institute of Technology, Kanpur

Lecture – 76
Application: SDP for MIMO Maximum Likelihood (ML) Detection

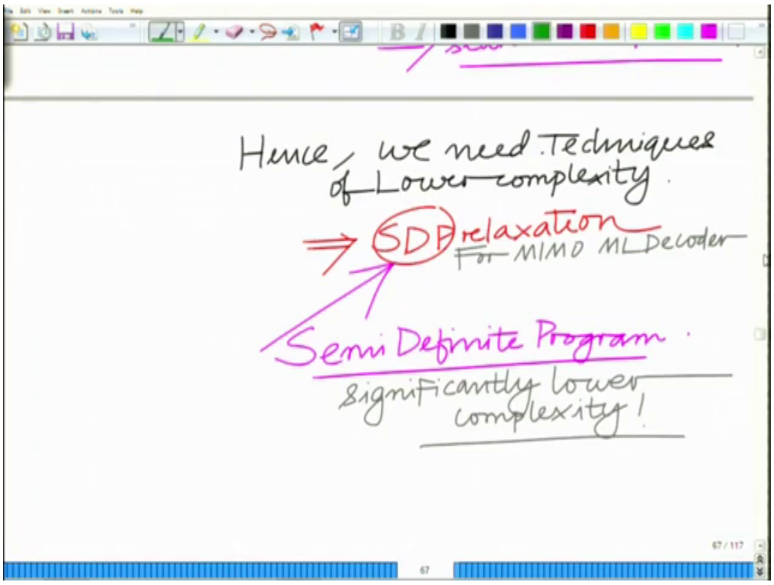
Hello, welcome to another module in this Massive Open Online Course. So, we are looking at same SDP that is Semi Definite Programming and its application in the context of MIMO detection that is how to reduce the complexity of the MIMO detector that there is maximum likelihood detector.

(Refer Slide Time: 00:30)



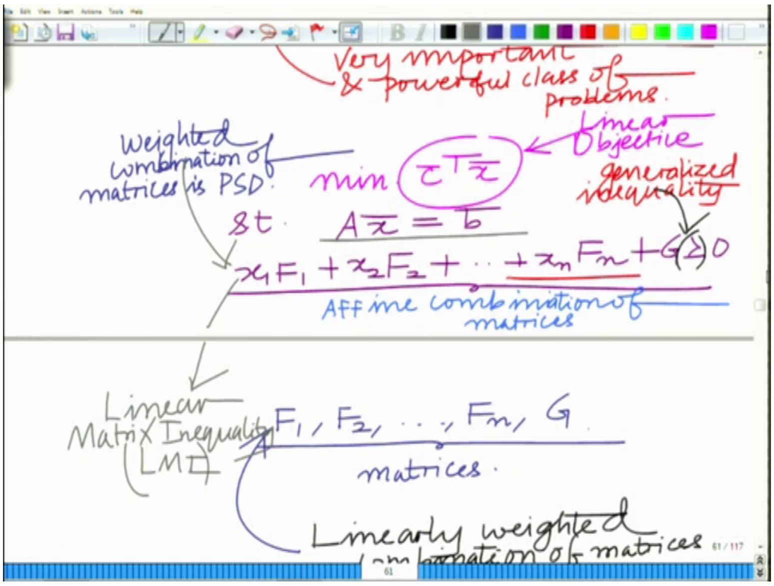
So, we are looking at SDP for MIMO ML detection rather.

(Refer Slide Time: 00:52)



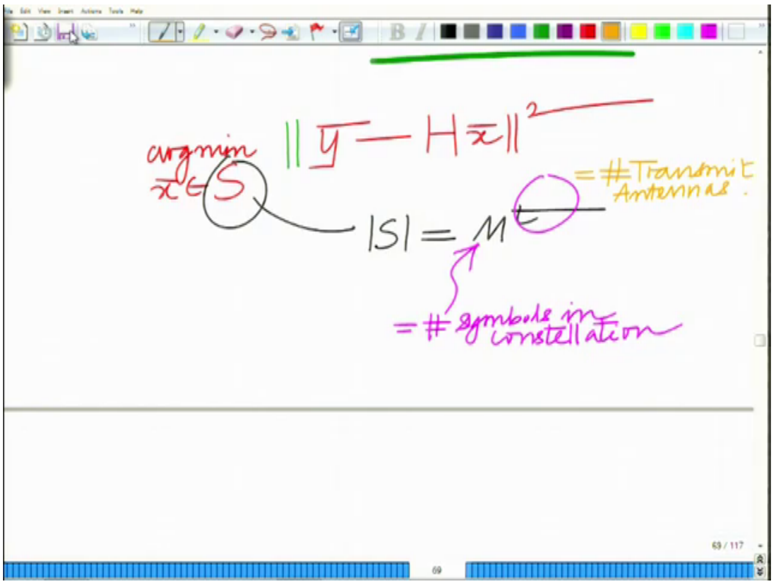
And, SDP employs an interesting constraint if you remember this; this is a positive semi definite constraint.

(Refer Slide Time: 01:01)



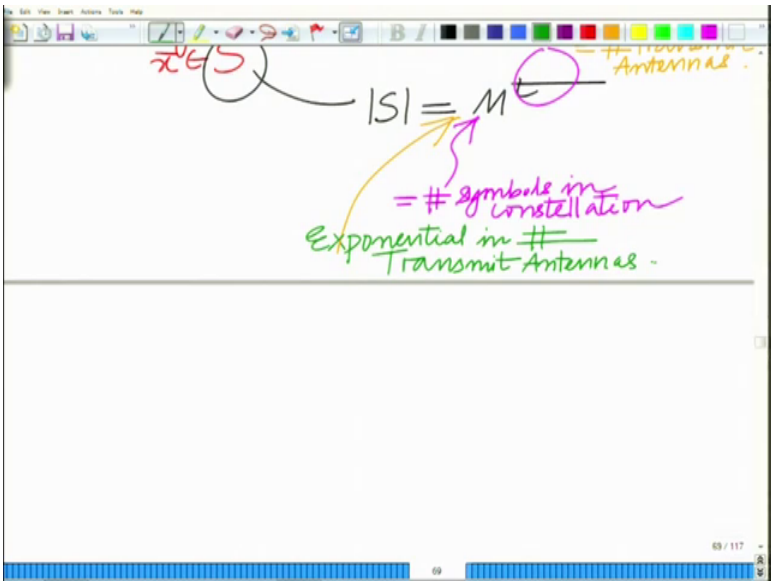
That is that basically this matrix has to be this linear combination of matrices has to be positive. So, positive semi definite this is termed as a linear matrix inequality, LMI, ok. So, SDP is interesting in the sense that it employs a matrix inequality. This is a linear matrix inequality SDP and force has a linear matrix inequality that is what is novel about SDP.

(Refer Slide Time: 01:38)



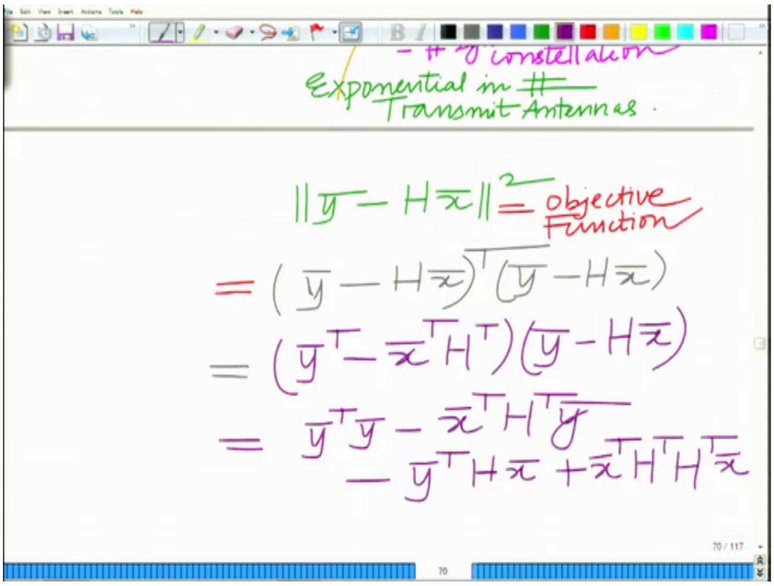
And, we are specifically looking at this MIMO detection problem where we have a received vector y bar minus H is the MIMO channel matrix Hx bar square and we have to perform argumin that is minimized that is find the x bar belonging to this set S which minimizes y bar minus Hx bar square. And, the problem with this is if you look at this set the cardinality of the size of that set is basically M raised to t, where M is the number of symbols in the consideration, and of course, t this is equal to the number of transmit t equals number of transmit antennas.

(Refer Slide Time: 02:46)



And, so, basically this is exponential in the number of transmit antennas which is a very high complexity, ok. So, this is exponential in number of transmit antennas. So, how can SDP help in this context for SDP what we do is let us simplify the cost function.

(Refer Slide Time: 03:12)

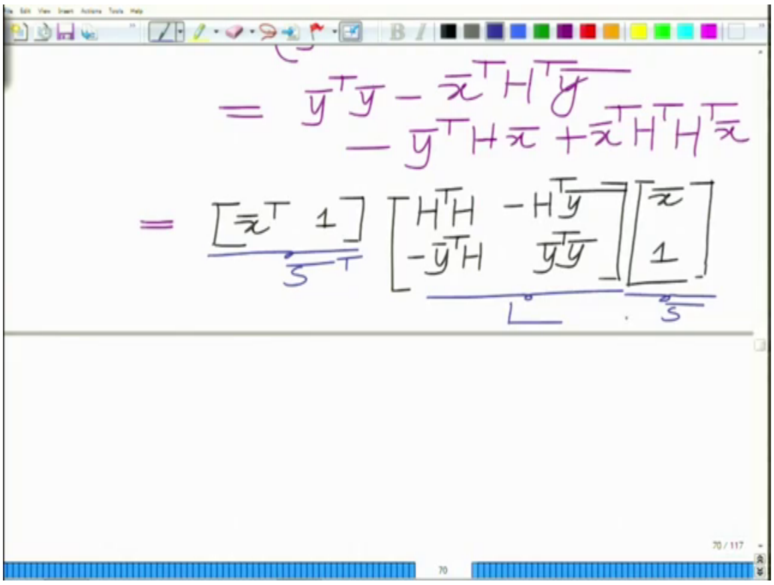


This is y bar minus Hx bar square, this is your objective function or cost that has to be minimized, and I can write this as follows y bar minus Hx bar this is norm square of a vector. So, I can write this as vector transpose times itself, ok.

So, I can write this proceed by writing this as y bar minus Hx bar transpose y bar minus Hx bar, this simplification we have seen previously also. So, I will go fast over this y bar minus y bar transpose minus x transpose H transpose into y bar minus Hx bar which is equal to y bar transpose y bar minus x bar transpose H transpose y bar minus y bar transpose Hx bar plus x bar transpose H transpose H transpose x bar, ok.

So, this is basically the cost function simplify cost function. Now, here I am going to do something interesting.

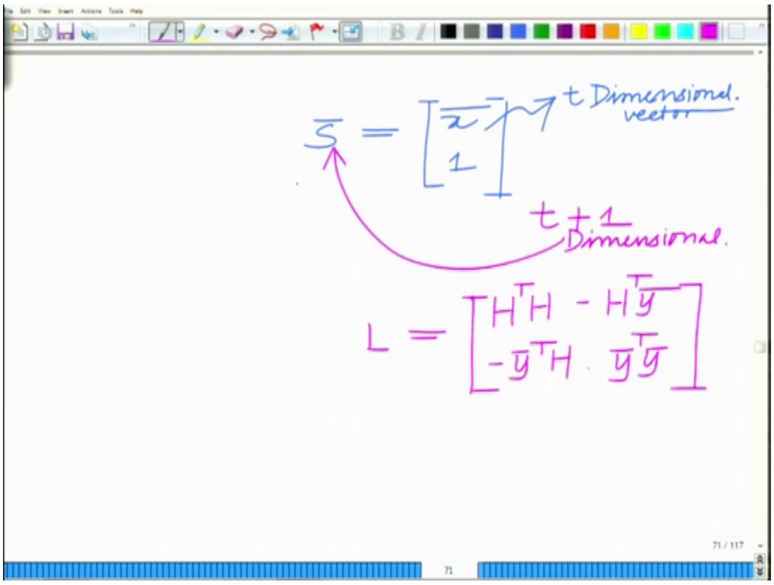
(Refer Slide Time: 04:39)



I am going to write this as follows and convince yourself this can be written as the following thing, ok. This can be written as equivalently as x bar transpose 1. So, I am making a column vector. So, x bar I am stacking it along with this number 1. So, I am making it t plus 1 dimensional column vector, x bar transpose 1 and this matrix I can write this as H transpose H minus H transpose y bar minus y bar transpose H y bar transpose y bar times x bar 1.

And, if I call this t plus 1 dimensional vector, now S bar or rather S bar transpose this I call as S bar this I can call as the matrix L.

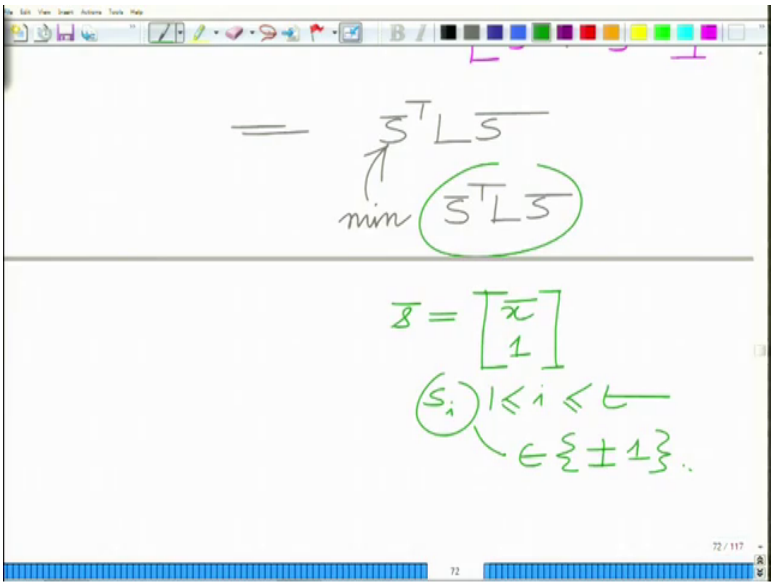
(Refer Slide Time: 05:49)



And therefore, what I have is I have this vector S bar which I am defining as x bar 1 this is a t dimensional and this is. So, this is single element. So, this net vector will be this vector will be t plus 1 dimensional and if you look at this vector this is a matrix L which is given as follows. So, L is the matrix which is given as follows this is H transpose H minus H transpose y bar minus y transpose H and this is y bar transpose y bar, ok.

So, therefore, we have this matrix L and therefore, now I can write this basically if you look at this, this is S bar transpose L into S bar.

(Refer Slide Time: 07:08)



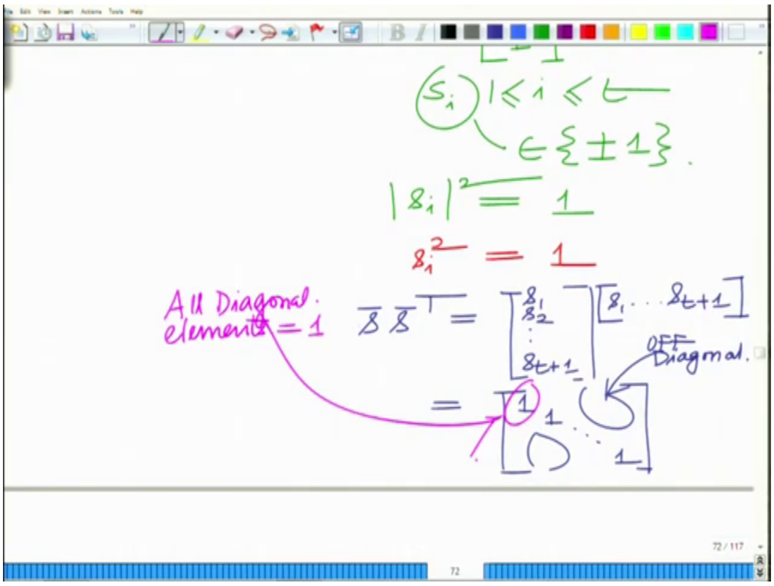
So, we have written this as S bar transpose L into S bar. So, I have recast this as weighted norm minimization problem that is we have written this in the form of S bar transpose L times S bar where L can be thought of as a weighting matrix, ok. And, now I can equivalently minimize this minimize. So, I can minimize this S bar transpose L S bar, ok. So, I can minimize S bar transpose L S bar.

However, the only problem here is that of course, we have constraints. Now, what are the constraints, now remember if you look at S bar still we have a mined the problem we have just recasted the problem has not changed we still have if you look at s bar which is x bar comma 1. So, if you look at S bar s-o, if you look at the ith element ok. So, your for 1 less than equal to i-th, so, if you look at the i-th element that is S i for 1 less than equal to i less than equal to t that still has to belong to the constellation.

So, let us say this is BPSK constellation. So, this has to be plus or minus 1, ok. So, these elements that is the first t elements are still the symbols that is x 1, x 2, up to x t. So, remember these still have to belong to the constellation that is each of this can only take two values that is plus or minus 1, which is where the problem is arising in the first place, alright because you have this large number of factors over which you have to search.

Now, I am going to manipulate this cost function in a step by step.

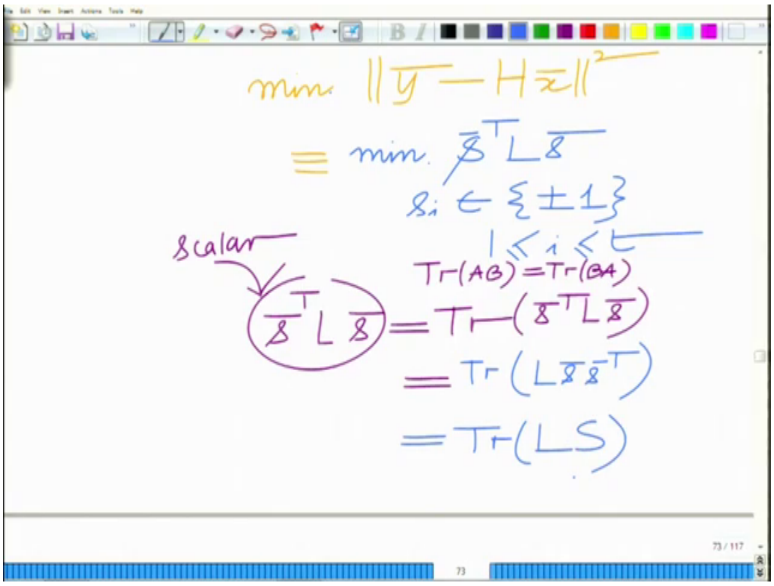
(Refer Slide Time: 08:59)



First note that each magnitude s i square in fact, for the t plus 1 element is also 1. So, all magnitude s i square equals unity or in fact, in this case simply considering real symbols we can note that simply s i square equals unity,. So, if you look at s bar s bar transpose which is basically s 1, s 2 up to s t plus 1 into it is transpose s 1 up to s t plus 1; if you look at the diagonal elements of this so, this is not a diagonal matrix, but if you look at the diagonal elements all the diagonal elements will be there is there are also off-diagonal. These are the off diagonal.

But, if you look at the diagonal entries all the diagonal entries are equal to 1 because each s i square is 1. Therefore, all the diagonal elements of this s bar s bar transpose are 1, are unit.

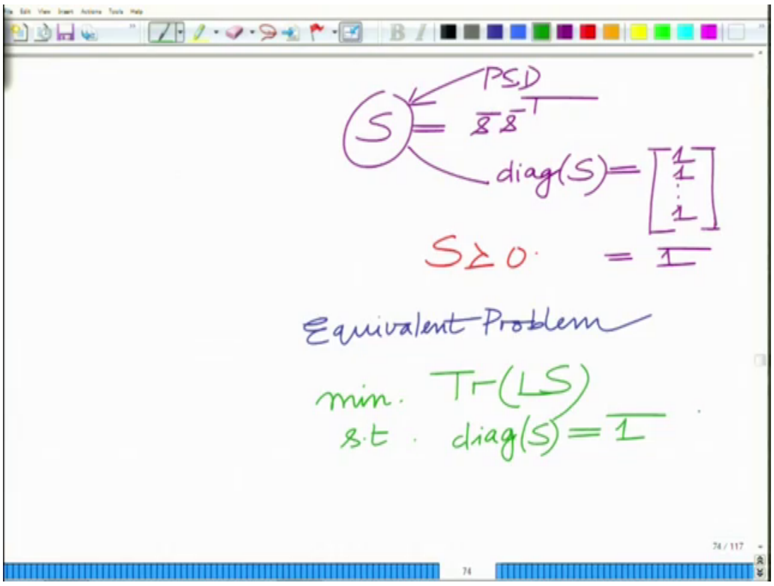
(Refer Slide Time: 10:47)



Now, what I am going to do is basically now you realize the first thing as we have shown these two are equivalent. So, minimize norm y bar minus Hx bar square can be equivalently written as minimize s bar transpose L s bar, each s i belongs to still this symbol of plus or minus 1, assuming BPSK binary phase shift key and for of course, 1 less than equal to i less than equal to t ah. Now, what I am going to do here is now I am going to employ this thing. So, now, first step – 1, observe that S bar transpose L s bar this is a scalar quantity this is a number.

So, I can write this as trace because if you take the trace of a number it remember trace is the some of the diagonal elements for a square matrix. So, a single number is a special case of square matrix. So, the trace will yield the number itself. So, this I can write this as trace s bar transpose L s bar. Now, using the property trace AB equals trace BA. So, this becomes trace of L s bar s bar transpose which I can now write as trace L types S.

(Refer Slide Time: 12:30)

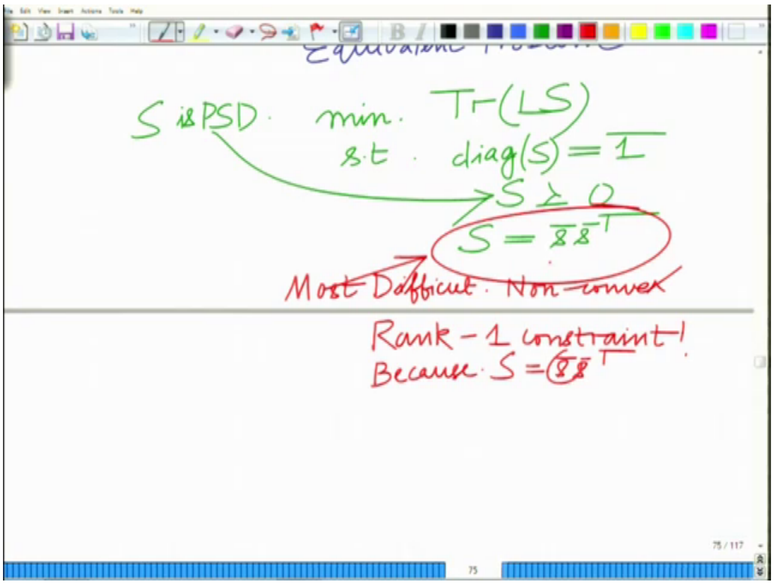


Here what is S? S equals the matrix s bar s bar transpose which is of course, positive semi definite because I can express this as s bar s bar transpose and diagonal of S equals basically you have all the elements are unity. So, this has to be this vector 1 bar and further we have seen that S is positive semi definite S is greater than equal to 2.

So, now, I can equivalently formulate this as. So, equivalent problem will become minimize equivalent problem is minimize s bar transpose L s bar subject to the constraint or. In fact, not minimized s bar transpose L s bar I apologize this is minimize trace of LS subject to the constraint that S is a positive semi definite matrix that is S is greater than or equal to 0 or first let us write the diagonal.

So, diagonal of S equal to unity or not unity that is each element of the diagonal of S is 1. So, diagonal of S is 1 bar the vector of 1’s.

(Refer Slide Time: 14:00)

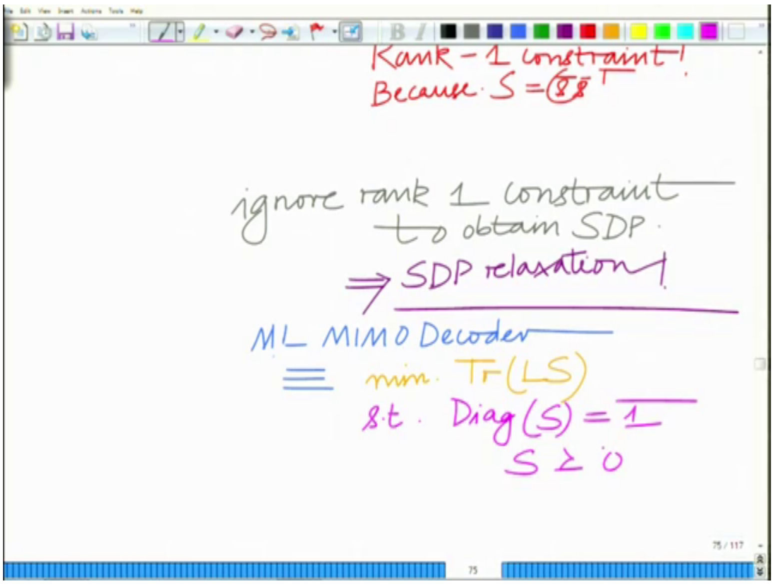


S is a positive semi definite matrix, ok. So, S is positive definite semi definite further S can be written as s bar s bar transpose. Now, of all the constraints this is the most difficult most difficult non-convex constraint because it is non-convex. This is known as a rank-1 constraint. This is known as a rank-1 constraint because S equal to s bar is that a single vector expressed as a vector s bar s bar transpose. So, whenever we have a vector matrix of this form which is vector types it is transpose then it will basically rank-1 constraint and this is a non convex constraint.

So, ensuring this is very difficult of constraining this optimization problem using this rank 1 constraint is very difficult. So, what we do is naturally what we do when we encounter something difficult or difficult to impose we simplify in this case we simply ignore this. This is known as an SDP relaxation, ok. So, we relax it.

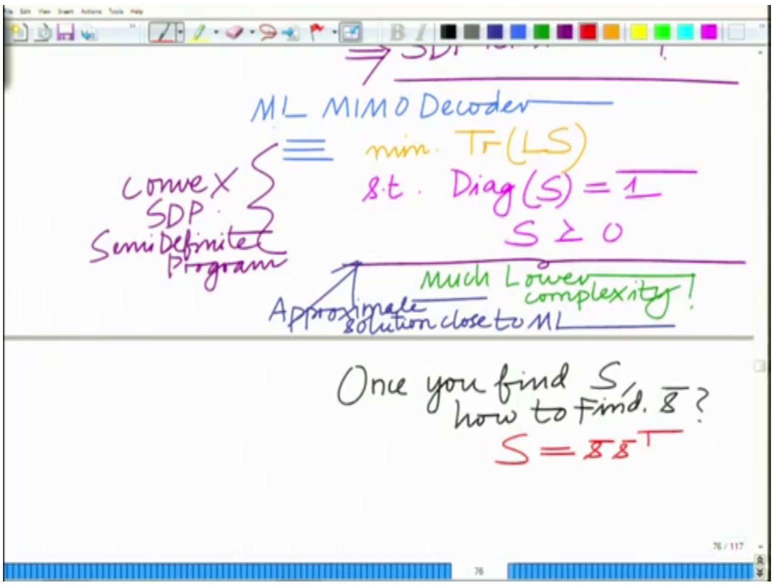
So, this rank-1 constraint makes it non-convex. So, it makes it non SDP. So, we relax it as an SDP that is ignore this rank-1 constraint.

(Refer Slide Time: 15:31)



So, what we do ignore the rank-1 constraint. So, we ignore the rank-1 constraint obtain SDP this is term as this is termed as SDP relaxation. This is termed as SDP relaxation,. So, what we have now is an interesting problem that is we can equivalently so, our ML decoder can be equivalently written as ML MIMO decoder for that matter can be equivalently written as minimize trace of LS subject to the fact that diagonal S equals the matrix of 1's S is positive semi definite and that is it, and this yield approximate solution.

(Refer Slide Time: 17:02)

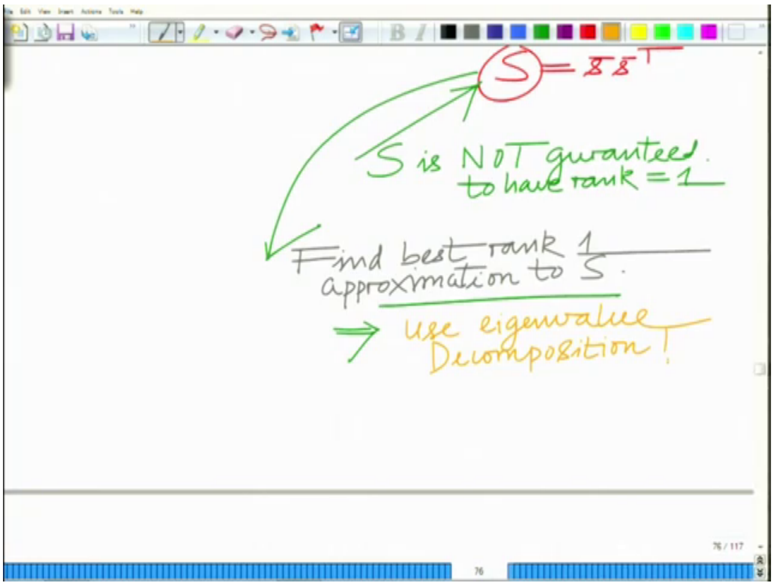


So, this is significantly lower complexity. So, now, from exponential complexity we have come to we are come to this is a convex optimization problem,. Now, this problem is convex. In fact, this is a SDP it has a linear matrix in equal.

So, this is a SDP convex, this is a SDP. Now, while the original problem is exponential this problem has much lower complexity. This has much lower complexity and therefore, it is very amenable to implement this in practice very suitable for practical implementation. The only thing is it yields an approximate solution, of course; it is solution that is actually shown to be very close to the optimal ML solution. So, this yields approximate solution close to the ML solution,.

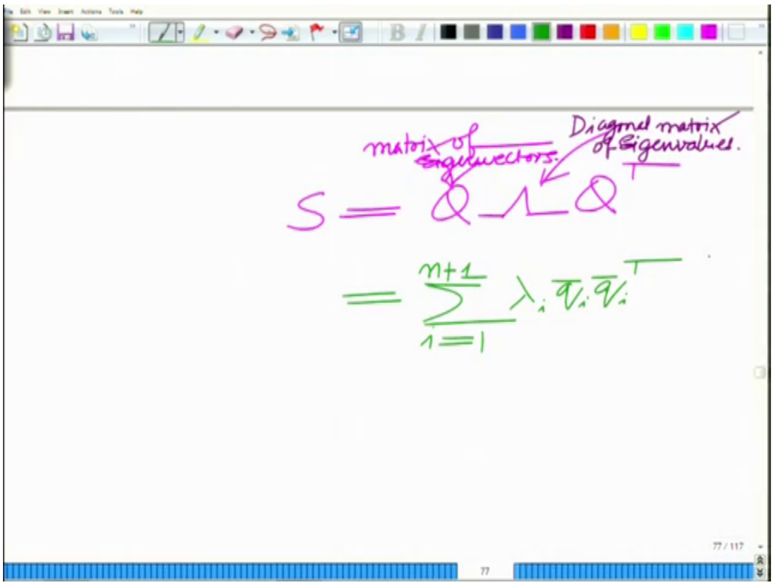
Now, once you find S now once you find S how to find S or how to find S bar rather. So, to the point is you find this matrix S; S equal to s bar s bar transpose, but not guaranteed because we have ignored the rank 1 constraint not S is not guaranteed to be a rank – 1 matrix. S is not guaranteed to be of the form s bar s bar transpose.

(Refer Slide Time: 19:09)



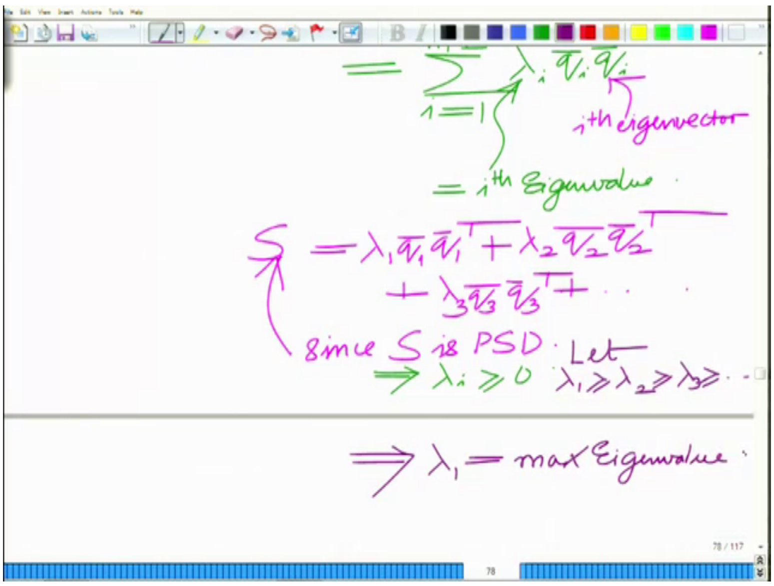
So, now, since we ignored it S is not guaranteed. So, S is not guaranteed. So, in this context how do you find S bar? Is very simple answer is where if you find the best rank 1 approximation to S that is the key here. Find best rank 1 approximation find best rank 1 approximation S, for that implies we use the Eigenvalue decomposition of S and this is also very interesting. How do you find the best rank 1 decomposition of a matrix of square matrix or symmetric matrix use the Eigenvalue use the Eigenvalue decomposition.

(Refer Slide Time: 20:23)



Remember the Eigenvalue decomposition is as follows I can write S as Q lambda Q transpose, where Q is the matrix of eigenvectors and this is diagonal matrix of Eigenvalues. This is the diagonal matrix of Eigenvalues and then I can write this as summation i equals 1 to n plus 1 lambda i q i bar q i bar transpose.

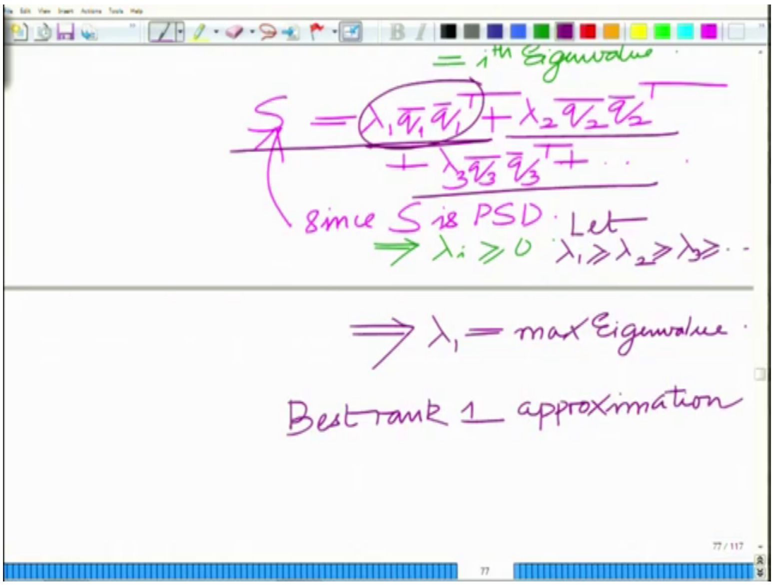
(Refer Slide Time: 21:17)



Where lambda i equals i-th Eigenvalue and q i bar equals the i-th Eigenvector and therefore, this is simply nothing but lambda 1 q 1 bar q 1 bar transpose plus lambda 2 q 2 bar q 2 bar transpose plus lambda 3 q 3 bar q 3 bar transpose. Now, since this is positive semi definite since S is PSD S is PSD note that lambda is are greater than or equal to 0. So, this implies lambda is greater than equal to 0.

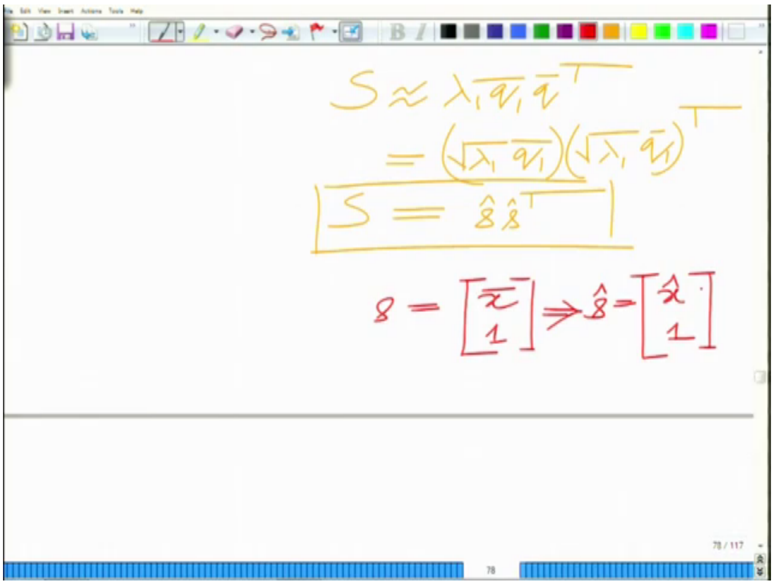
So, I can always arrange them in decreasing order, alright. So, this is lambda is a real quantity is in fact, there greater than equal to 0. So, I can always arrange lambda 1 greater than equal to lambda 2 greater equal to lambda 3 so on. So, so let lambda 1 greater than equal to lambda 2 greater than equal to lambda 3 and so on. So, implies lambda 1 equals maximum Eigenvalue.

(Refer Slide Time: 22:42)



Then, the best rank – 1 approximation, best rank – 1 approximation is simply this that is you choose S equal to lambda 1. So, ignore the rest ok. Just simply choose S equal to the largest Eigenvalue Eigenvector outer product of Eigenvector corresponding larger Eigenvector.

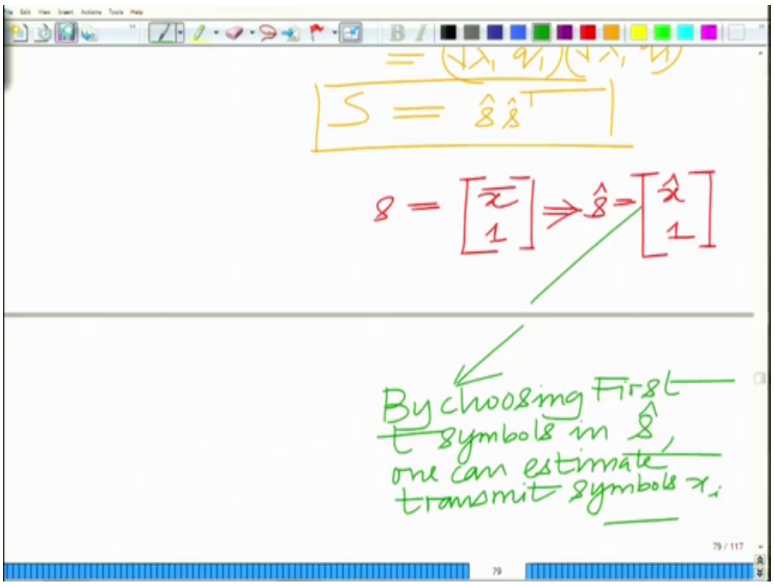
(Refer Slide Time: 23:11)



So, the best rank 1 approximation is simply S is approximately equal to lambda 1 q 1 bar q 1 bar transpose which I can always write a square root lambda 1 q 1 bar square root lambda 1 q 1 bar transpose which is now equal to here s hat s hat transpose. So, S equal to or rather s hat equal to s hat s hat transpose.

And, the final step is s equal to remember your vector x bar comma 1. So, that implies s hat equals x hat comma 1, ok. So, by choosing the first t symbols of s hat you get transmitted symbols x in x bar.

(Refer Slide Time: 24:04)



So, by choosing 1 can estimate or detect rather the transmit symbols x and from that of course, you can get the transmit vector x that is how you basically.

So, you take the original ML decoder right, recast it in a different form alright and then you relax the critical thing here is that relax the rank 1 constraint that makes it a semi definite program, this process known as SDP relaxation. From the SDP relaxation you get S which is a positive semi definite matrix from that you perform the Eigenvalue decomposition, but get the best rank when approximations. So, that will be. In fact, there is nothing, but the principle Eigenvector alright the scale to principle I can given in terms of the scaled principal eigenvector of S from that you take the top t symbols those are your MLS, alright.

So, let us stop here and continue the subsequent modules.

Thank you very much.