Matrix Solvers
Prof. Somnath Roy
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur

Lecture - 56
Preconditioners

Welcome, we will start our discussion on Preconditioning techniques. So, till now we have looked into different matrix solvers; we started with the basic matrix solvers like Jacobi Gauss Seidel. And, then we looked into projection based matrix solvers, say steepest descent, minimum residual residue norm. Then we discussed about Krylov space based solvers, which is full orthogonal method GMRES conjugate gradient which are faster methods than the previous methods we have discussed.

And we have seen that conjugate gradient is a very first method; however, it has a limitation for symmetric positive definite matrices. So, we looked into bi orthogonalization methods in which we looked into development of bi conjugate gradient method, which is a conjugate gradient like method, but applicable to a different wider class of matrices. Then we also looked up into block relaxation methods and parallelization strategy.

So; however, as we have looked into a class of methods, what we found that these methods has certain restrictions in terms of their applicability like the Jacobi or Gauss Seidel is applicable only to diagonally dominant matrices conjugate gradient or steepest descent is applicable only to symmetric positive definite matrices. Although GMRES or bi conjugate gradient can be applied on much wider range of matrices at least the matrix has to be non singular then they can solve it.

However the issue is that they are iterative methods and therefore, their solution needs to the solution vector needs to be iterated over a number of steps, and the number of steps are usually of the order of the size of the number of rows of the matrix. And after these iterations only it will converge in some cases the number of iterations are much more than the size of the matrix also.

So, the number of iterations is a binding parameter or a restrictive parameter when we are looking into an iterative solver whatever iterative solver we design through very refined linear algebra if we can define a very good iterative solver, still this iterative solver will take a number of steps to converge. And, this number of steps or the rate of convergence of an iterative solver, which is the rate at which the residual reduces to 0 or the difference between the last updated solution and new updated solution reduce.

This rate is a function of some of the matrix properties; say think of a Jacobi or Gauss Seidel type of solver. The rate of convergence is a function of the spectral radius of the iteration matrix. If I look into conjugate gradient matrix, it is some way of function of the condition number or spectral root over of condition number the rate of convergence is function on that.

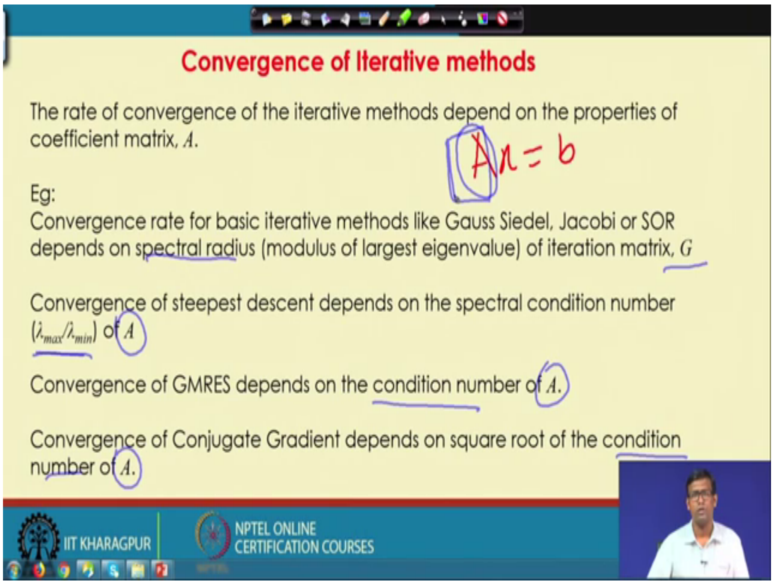
So, if we can change the matrix we have better solution, but for some matrices especially long back we discussed about condition number the matrices which are ill condition; that means, the singular values one singular value is very large and another singular value is very small. In these type of matrices the number of steps will be very high because the convergence rate is low.

So, what should we do there? One idea is to we will employ a lot of steps to solve it, but remember as the matrix is larger in size in each step lot of mathematical operations are done. So, if the number of iteration steps are very high the total number of mathematical operations are also very high.

So, that is a why the question comes that how we can reduce the number of iteration steps in a sense, how we can reduce the number of mathematical operations and can have a faster solver. We should consider the fact that the quest for looking into different matrix solvers, starting from the age old Gauss elimination method to what we are discussing like by conjugate gradient or GMRES type of solver, the quest for developing different matrix solvers came simply due to the fact that we want to have faster convergence.

And in reality we are say you are doing a weather simulation where you have a very large matrix and you have to get very fast result to predict path of a hurricane, and you really cannot afford the number of computations to be so, high that it takes lot of time and the solution is not available when it was required to sub certain purpose. So, you need to look for faster solver; and this precondition technique is in a way we will see that how we can do some alteration with the matrix properties. So, the convergence can be obtained in a faster manner.

(Refer Slide Time: 05:35)



So, what we are discussing is that the rate of convergence of the iterative methods depend on the properties of the coefficient of matrix, A. So, if you are solving Ax is equal to b, the matrix A determines what should be the rate of convergence. Because whatever theorem we read about convergence, the theorem says that it the method should converge for any initial guess x 0 only what will be the rate of convergence, that depends on certain properties of the coefficient matrix A.

For example convergence rate of basic iterative methods like Gauss Seidel, Jacobi or successive over relaxation depends on spectral radius or the modulus of largest eigenvalue of iteration matrix G. If the spectral radius is close to 1, the convergence is fast if the spectral radius is large if is very if the spectral radius must be less than 1; I am sorry if the spectral radius is much smaller than 1, the convergence is faster if the spectral radius is close to 1 the convergence is slower or it will take more time. And how G is determined? We have different methods like for Jacobi there is one way to obtain G for Gauss Seidel there is another way to obtain G for SOR there is another form of G.

However if we using one particular method, for one particular matrix G is fixed so, we are kind of tied with the eigenvalues of G and the rate of convergence of this method. Convergence of steepest descent method depends on the spectral condition number which is the radius of magnitude of maximum eigenvalue by minimum eigenvalue of A.

And as this particular value this spectral condition number is close to 1 the convergence is better if the number is larger the convergence is poor.

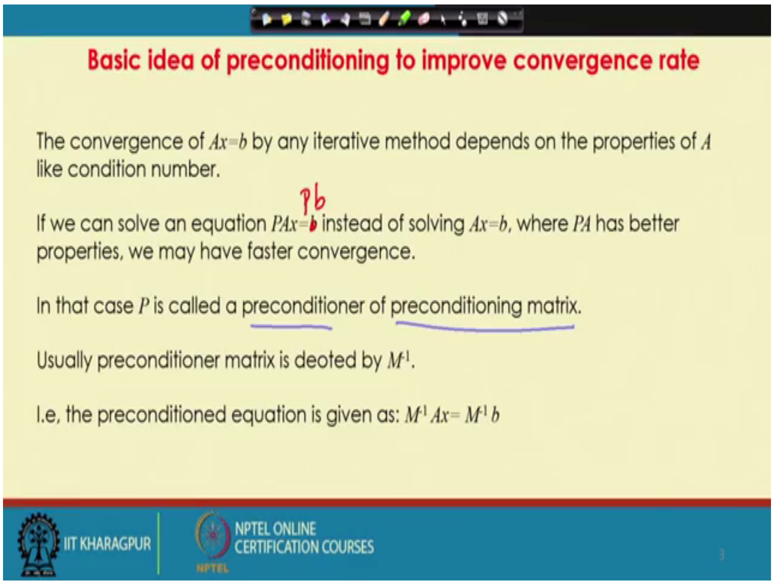
Convergence of GMRES depends on the condition number of A which is that ratio of the singular values of A and or mode A by mode A inverse and convergence of conjugate gradients depends on square root of condition number of A. So, as smaller is this value the condition number is always the number is the ratio of largest eigenvalue largest singular value by smallest singular value. So, it is always greater than 1.

Now, once we have a square root of this value, this number is smaller as smaller as this number the convergence is faster that is a conjugate gradient is usually a very fast solver because a because it depends on square root of the condition number of A. However, the property is related to this matrix A like this or again condition number here A spectral radius of G, all these properties are related to the property to the nature of A. And, for certain matrix which has poor conditioning or ill conditioned matrix the convergence rate might may be very small.

But in reality we might encounter lot of matrices which has ill condition poor condition number. So, what should we do for them the like say the quest is to get faster solver. So, you should not be restricted by the fact that the condition number if A is bad. So, you will take a lot of time which will we will see the methods by which we can still improve the convergence properties.

So, one thing which you are not going to do any further in this course, not to look into new solver methods rather we will use the solvers we have discussed already and see that how certain other techniques which is one of this is preconditioning which we are discussing, can be applied over this solvers to get better convergence.

(Refer Slide Time: 09:39)



The basic idea of preconditioning is that that the convergence of Ax is equal to b by any iterative method depends on properties of A like condition number. And preconditioning means, precondition a multiply something with A and something with b, something with the left hand side of the equation and something with the right hand side of the equation. So, that the new matrix say pa has a better condition number and you can solve it better.

So, if we can solve an equation which is sorry which is PAx is equal to instead of this is PAx is equal to P b instead of solving Ax is equal to b, where PA has better properties we will have faster convergence. And we can see that x will remain same instead of x is equal to b if we solve PAx is equal to b the solution x will remain same. Only advantage will be that PA will probably have a smaller condition number or the iteration matrix will have a larger spectrum smaller spectral smaller spectral radius so, that the methods are faster.

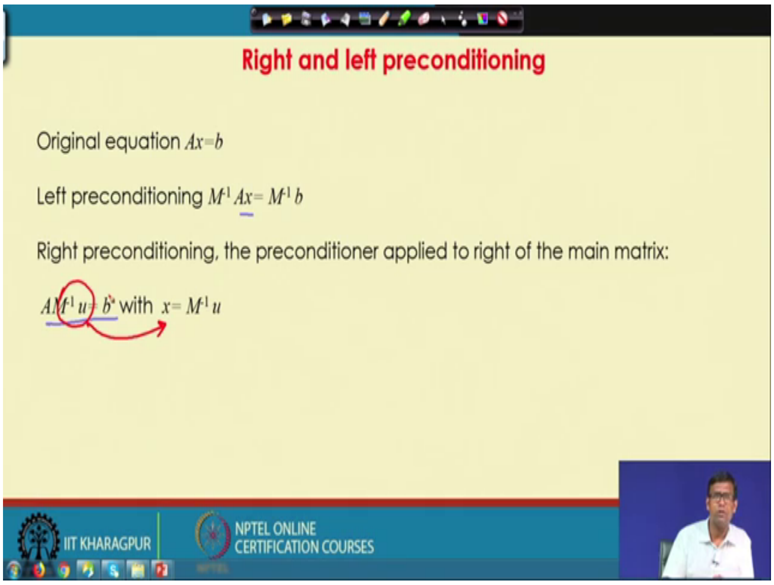
So, you can have a faster convergence if instead of PAx we can solve P b. So, we have to very carefully find a p. So, that when p is multiplied with a gives a matrix which has better properties. So, the convergence rate increases. Because the convergence rate is anyway function of the properties of the solution matrix or the coefficient matrix. In that case this P is called a preconditioner or preconditioning matrix, P which is multiplied with A before we go for solution of Ax is equal to b, actually improves the conditioning of the coefficient matrix and we call it to be a preconditioner or preconditioning matrix.

Usually a preconditioner matrix is defined by denoted by M inverse inverse of some matrix which you have to find out. And the preconditioned equation is given as M inverse Ax is equal to M inverse b. So, Minverse which is the preconditioner multiplied in the left of the. So, coefficient matrix a gives a new equation system M inverse Ax is equal to M inverse b. And our presumption is that this matrix M inverse a has better condition number or rather smaller condition number so, that the equation M inverse Ax is equal to M inverse b can show faster convergence when you look into a iterative method for solution.

So, therefore, the goal moves that will we have to see that how multiplying a inverse with the matrix with a matrix change our equation system. Do we need to find out how should we find out an M inverse, which increases the condition number there is a first question. While increasing the condition number the few properties of the matrix like if the matrix is symmetric, M inverse a must be symmetric otherwise we cannot use it for conjugate gradient method.

So, these properties has to be preserved some of the properties of the matrix has to be preserved, and we need to see how M inverse can be found out whether we need to explicitly find M and find its inverse which will finding inverse of a matrix is again a very very complicated job, because the numerical operations are same as the Gauss Jordan method order of n cube. So, do we need to explicitly find out inverse of the matrix; what are the techniques for that and how smartly we can apply this idea that we will find out an M inverse will multiply it with the equation M Ax is equal to b and M inverse a will have a better conditioning number will get faster solution how smartly this can be imposed that we have to we can do.

(Refer Slide Time: 13:37)

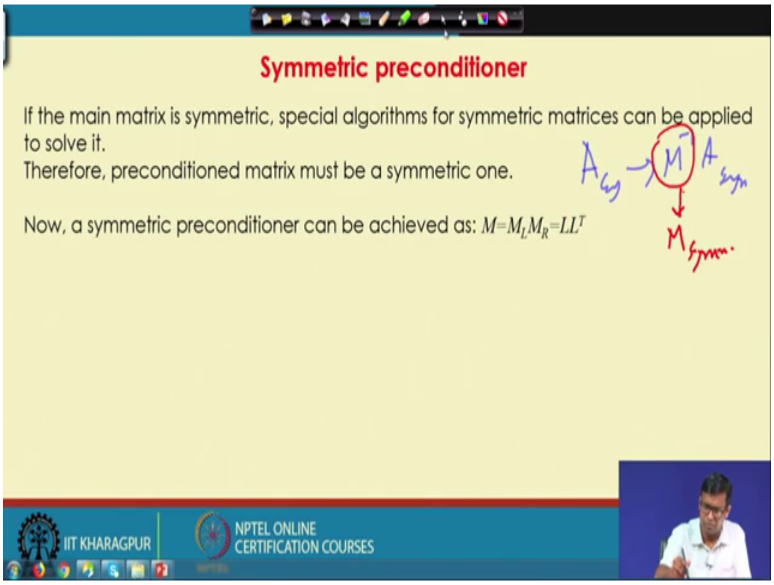


So, there is there is the original equation which is Ax is equal to b, we call it left preconditioning if f M inverse is multiplied on the left of a. So, we get M inverse Ax is equal to b. Essentially this preconditioning does not change the solution vector x x will remain same if we solve it. There is another way which is called right preconditioning where the precondition is applied right to the matrix and here we change the solution vector. Instead of solving Ax is equal to b, I solve a M inverse u is equal to b where this term M inverse u is now the actual solution vector x. So, I have to solve basically two equations first is AM inverse u is equal to b and second is x is equal to M inverse u.

There is one difficulty in that that I have to again solve our matrix equation M inverse u. So, the inverse of M should be readily available or it should be very simple matrix like M should be something like a diagonal or a lower triangular matrix for which or an upper triangular matrix you can solve it directly. So, that also you have to look into it if you are using right preconditioner.

The issue in using right preconditioner is that, that the right hand side b remains same instead of M inverse b; we are keeping the right hand side b. So, you are not doing saving on matrix multiplication and also certain properties certain advantages we might have when we multiply a M inverse on the right of a in some cases like M inverse a might give a M inverse a might give a bad condition matrix, but M m inverse by give better condition matrix. So, these things are possible in some cases.

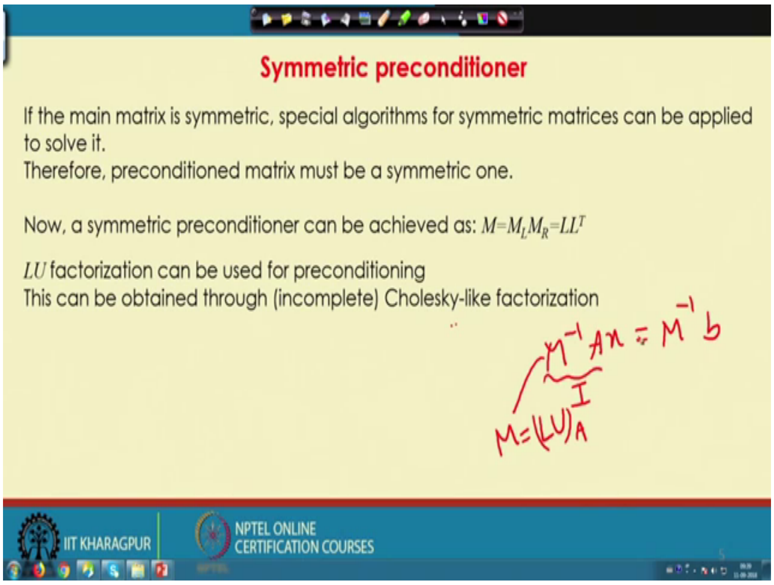
(Refer Slide Time: 15:36)



If the now there is a need of preserving symmetricity which I was discussing at the beginning. If the matrix is symmetric special algorithm for symmetric matrices can be applied to solve it like conjugate gradient, like steepest descent which is only applicable for symmetric matrix. Therefore if I am using a preconditioner, the precondition matrix M inverse a or a M inverse must also be a symmetric one otherwise. So, I thought of using conjugate gradient on a symmetric matrix I multiply do you through the M inverse, M inverse A is no longer the symmetric matrix. So, you cannot use conjugate gradient there.

So, in order to use the algorithm for symmetric matrix symmetricity has to be preserved; a symmetric precondition right. So, because we are instead of A, A is symmetric and we need a M inverse A which is symmetric that shows that M inverse must has to be symmetric or M must have to be symmetric, if M is symmetric M inverse will also be. So, we need to get a symmetric M matrix there and a symmetric preconditioner can be obtained as M is equal to ML M R as a decomposition which is equal to LL T any symmetric matrix any symmetric non-singular matrix is decomposable into LL transpose, L is a lower triangular matrix, L transpose is an upper triangular matrix this LU decomposition.

(Refer Slide Time: 17:17)



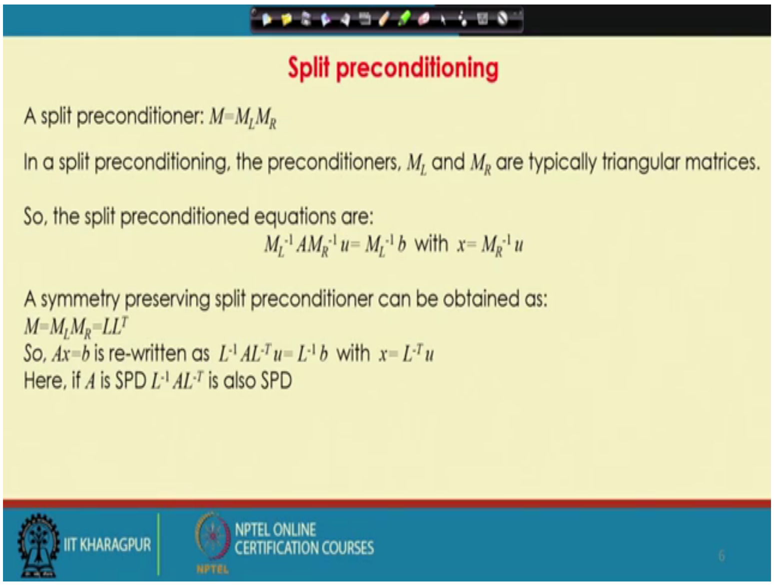
LU factorization and also we can see that if any symmetric matrix can be obtained through an LU decomposition where M is equal to LLT, LU factorization can also be applied for preconditioning in and this is a very very useful technique of preconditioning we will look into the later parts of this discussion that M can be obtained as an LU decomposition or in as a LU factorized matrix.

And if it is a symmetric, then it will automatically an ma L transpose matrix this can be obtained through incomplete Cholesky like factorization or using first few steps of a Gauss elimination method where, a gauss elimination can be used to get an LU decomposition of that. So, something like a Gauss elimination or two or Cholesky factorization can be used to get an LU form of M.

However we call it in complete LU factorization because complete LU factorization of a matrix takes n cube number of steps. So, you really you do not go for this n cube number of steps, rather we run the steps for certain number of points we get a LU factor which is which is equal to M m is always equal to mu, but M is not equal to a inverse there is some benefit if M m is sorry M is not is equal to a there is some benefit if M is equal to a, but it is not that we will come into it. Because we are solving I will I will probably emphasize it here, we are solving M inverse A x is equal to M inverse b the best solution could have been obtained if this is very close to identity. So, if M is equal to LU of A then this equation is solved in a step we do not even need any iterative method.

However getting LU decomposition of a is a costly step. So, you do not do it in practice we run it up to few steps and then stop it, but this part I will again elaborate when we will we will come to the relevant discussions; what we can say that a symmetric preconditioner can be obtained as M is equal to LL transpose, and it is a general practice to use an LU matrix M is equal to LU as a preconditioner.

(Refer Slide Time: 19:37)



There is another thing called as split preconditioner because we have seen that M is equal to LU is the general practice of using preconditioner, you just discuss this, but the rationale I will discuss later, but M is equal to LU is a general form of preconditioner let us recall that and so, M is all already we formed M in a split way there is one factor L another factor R. Another idea is that a split split preconditioner when M is equal to M L and M R and ML and M R both are triangular matrix and ML is multiplied left of the coefficient matrix a M R is multiplied right of the coefficient matrix a. So, the split condition equation system looks ML inverse A MR inverse u is equal to ml inverse b where x is equal to M R inverse u.

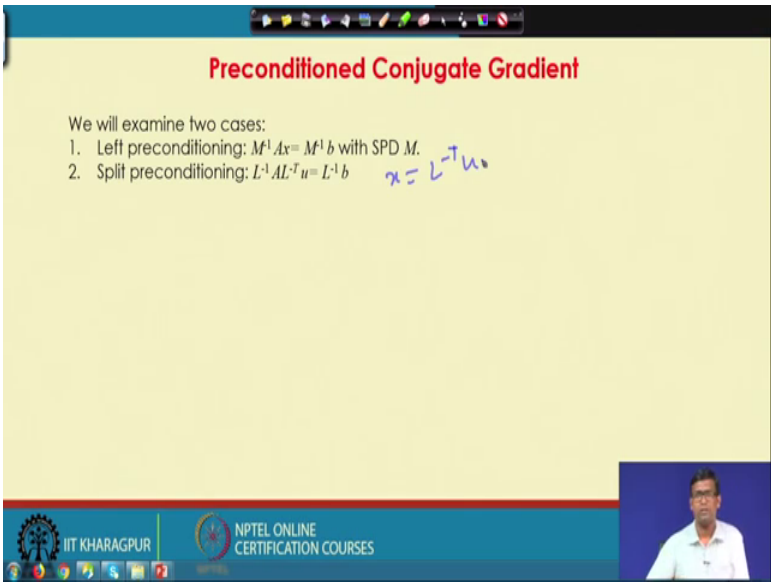
So, this has the right preconditioning part, which gives an auxiliary equation x is equal to M inverse u, and also the left preconditioning part which multiplies L inverse L b; what is the advantage? Advantage is that if we can well design ML and M r we can very well preserve the symmetricity of a matrix that is the high best advantage of this.

A symmetry preserving split preconditioner can be obtained as a LL transpose. So, the equation will be L inverse same L transpose inverse u is equal to L inverse b. So, Ax is equal to b can be written as L inverse AL inverse transpose u is equal to L inverse b. So, all these exercises where you rather writing the equations in little more complicated way by multiplying pre multiplying post multiplying matrix inverses with it has one particular focus and this focus is nothing, but improving condition number of the solution matrix and that we have to investigate in later stages.

But what we will right now see is that, if we can express equations in this form what will be the resulting algorithms say we are trying to use GMRES over x instead of x is equal to b I am trying to use GMRES over this particular equation system what will be the resulting algorithm. And we also have to take care of that fact that L L inverse transpose has to be designed in such a way that this equation should not bring a new overwrite over the solution the total cost should not increase should reduce.

So, we have to also see how well inverse transpose can be reduced, but if L is typically a triangular matrix L is a lower triangular matrix. So, this is the inversion of a lower triangular matrix, which is which actually takes n steps and can be obtained very fast. If A is symmetric positive definite then, L inverse L inverse transpose is also symmetric positive definite.

(Refer Slide Time: 22:39)



So, now, will look into preconditioning of conjugate gradient method and we will examine two cases; one is the left preconditioning where we will try to see M inverse Ax is equal to M inverse b with symmetric positive definite matrix a. And, another is split point preconditioning where we will see L inverse L A L inverse transpose u is equal to L inverse b. The in the left preconditioning we have to choose M very carefully so, that M inverse A remains a symmetric positive definite matrix, and that requires that M also must be symmetric positive definite matrix.

In the right preconditioning the, this matrix is always symmetric positive definite because if a is symmetric positive definite this lower triangular matrix inverse and transpose will give A symmetric positive definite matrix. So, you do not need worry, but we also have to solve x is equal to L inverse transpose u for this.

So, in the next session, we will look into the algorithms through which we can solve this type of equations. We look into conjugate, how conjugate gradient algorithm will be modified so, that we can solve a preconditioned equation both left preconditioning and split preconditioning and what are the relations between them.

Thank you.