Finite Element Method and Computational Structural Dynamics
Prof. Manish Shrikhande
Department of Earthquake Engineering
Indian Institute of Technology, Roorkee

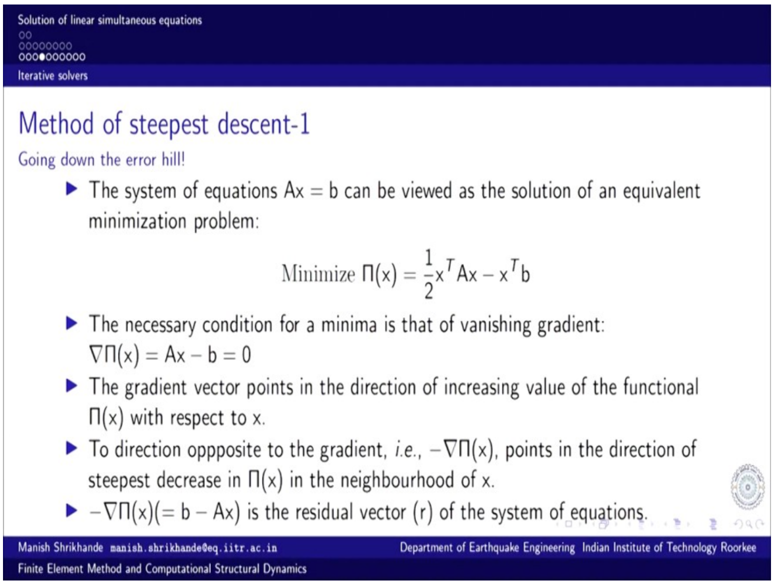
Lecture - 46
Solution of Linear Simultaneous Equations - IV

Hello friends. So, we have seen the Gauss - Seideland Jacobi iteration schemes for_implementing the iterative solvers. So, those are very simple matrix multiplication and series of matrix multiplication with recursion of I mean previous recomputed solution going back to in the computation to generate the next iteration next improved estimate and slowly the methods these techniques they gradually converge to the true solution although the rate of convergence is very very slow as compared to the methods that we will discuss now.

But still they provide us the basic idea, basic structure and basic feel of how the iterative schemes actually work and why it is advantageous, because all that we need is a simple matrix multiplication routine and we only deal with non-zero elements of matrix A because there is no point in storing something as 0 and then computing multiplication of something with 0 as 0. So, we can that does not affect any of our computations what is soever.

So, we use with iterative schemes we make use of a sparse storage and using sparse storage schemes, we implement these using standard matrix multiplication algorithms with the proper addressing mechanism, using data structures that are used and implement these methods.

(Refer Slide Time: 02:06)



So, the next method that we discuss is the method of steepest descent. So, in this case the system of equations A x = b can be viewed as the solution of an equivalent minimization problem, that is we minimize a functional of vector x given by a quadratic functional $1/2(x^TAx - x^Tb)$. So, if we minimize this with respect to x then the solution is of course, A x = b.The minimization is going to be A x - b = 0 the first derivative of first variation with respect to x being 0. So, that is the conditionof minimization with respect to x.

So, that vector x which will minimize this functional pi would correspond to the condition that first variation of this quadratic functional with respect to x should be 0. So, that is the condition and condition necessary condition for minima is that the vanishing gradient. So, $\nabla \Pi$ as a function of x should = A x - b and that should be equal to 0. So, null vector.

So, gradient vector this gradient vector that we have gradient of this functional Π . So, this points in the direction of increasing value of the functional Π So, I mean that is our standard definition of a derivative gradient is a derivative. So, $\frac{dx}{dy}$ the positive sense as we increase with respect to variation of y with respect to x, so that increases that indicates the positive sense is that it is a positive gradient. So, the function value is increasing with respect to x.

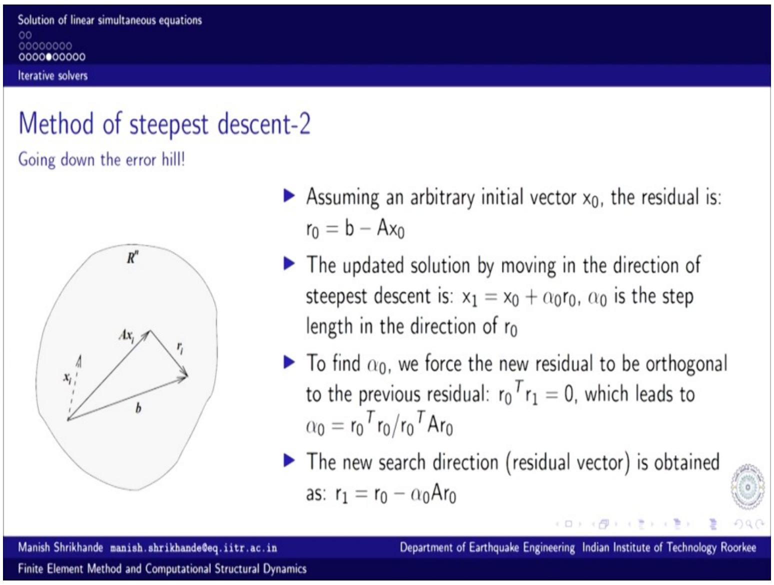
So, the gradient vector_it will point to the increasing value of functional Π with respect to the vector x. So, if we are interested in decreasing the functional value, because we need to minimize with the whole idea is we have to minimize the functional. So, the decrease if the gradient vector points to this increasing value the decrease will be in the opposite direction and that is what gives us to the solution leads us to the solution. So, we should actually come down the hill.

So, gradient actually points to up upward the hill let us say it is a the functional Π represents a hillock and the gradient vector at any point x points to the upward direction towards that hill right. So, the negative value of the gradient that is computed will point to the decreasing down the hill direction.

So, how much we should move in that direction, because it may. So, happen that at that point this one particular direction we have the steepest decrease most rapid decrease in the functional value, but that will continue only up to certain point, after that may it may be some other direction which is decreasing the functional value at a more rapid rate. So, that is what we call as line search how long we should move along this descent line.

The direction opposite to the gradient that is negative of nabla pi points to the direction of steepest decent steepest decrease in pi in the neighbourhood of point x. And gradient is of course, negative of gradient is of course, the residual vector that we had defined earlier of the residual vector of the system of equations. So, that also serves as a monitoring or tracking of the convergence of equations..

(Refer Slide Time: 06:35)



So, going down the hill so, this is what it looks like. So, for any particular_iteration let us say the trial vector or the solution is represented by this vector x in ith iteration and this is the complete space of n number of variables real numbers. And A times x_i points is a vector pointing in this direction and b is a vector which is pointed in this direction..

So, this r_i at the ith iteration so, the error_in solution in ith iteration is this particular vector and this is what tells us that we should nudge the solution in this direction which will take us the shortest possible way to meet the vector b and how far do we go in this direction, I mean this length how do we measure this length what should be the length of our step that is what we use as the line search algorithm..

So, assuming any trial vector_initial vector x_0 the residual is r_0 given by b - A x_0. So, matrix vector multiplication and subtracted from the right hand side. The updated solution by moving in the direction of steepest descent is $x_1 = x_0 + \alpha_0 \times r_0$.

So, this α_0 is the step length in the direction of r_0. So, r_0 so, this is the how much should we move along this particular direction in one iteration. So, that we keep on moving on the steepest descent what the most rapid decrease in the functional value we move in that direction for the time until it is the maximum reduction possible.

So, to find this step length we force the new residual to be orthogonal to the previous residual. So, this is only possible when there is no further decrease possible so; that means, the error all that we have we have already accounted for all that is there to be gained in this direction. So, there is no new information that can be accommodated from this direction by following this particular direction. So, new additional information in the representation of expansion of the basis is possible by following this direction.

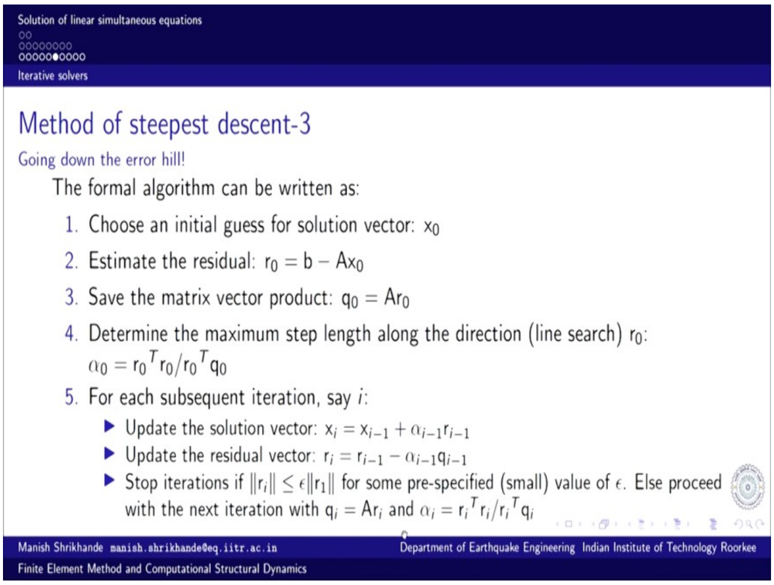
So, when we do that. So, the new vector_is this and from this we can estimate what is the new residual r_1 going to be as a function of %alpha naught %alpha 0 and once we have r_1 we enforce that the new residual should be orthogonal to the previous residual. So, that we have explored all that is possible to be explored in the previous search direction and the new search direction should be orthogonal to the previously searched direction.

And that leads to; so when we enforce the orthogonality condition that is projection of r_0 on r_1 is 0 and that leads us to the condition for the step length; %alpha 0 is just a ratio of two scalars r_0 transpose r_0 divided by r_0 transpose A r_0. So, A r_0 is simply the matrix vector multiplication..

So, new search direction the residual vector_is again obtained as simply by previous residual previous search direction is modified. So, $r_0 - \alpha_0 A r_0$ so that is the new search direction that is available and then the new solution is computed as x_1 as $x_0 + \alpha 1 r_1$ and then the iterations continue in the same way recursively until we reach the norm of residual becoming lower than the acceptable norm.

So, the formal algorithm we can list down as a sequence of steps. So, going down the error hill so, we keep on going down the hill. So, we start with an initial estimate initial guess for the solution vector x_0, the residual is estimated r_0 is estimated as b - A x_0 and that is the gradient negative of gradient.

(Refer Slide Time: 12:01)



And we save this matrix vector product because this is used very often. So, then there is no need to reinvent the wheel again and again and we can save this matrix vector multiplication in a vector and use this vector wherever we need to use this matrix vector multiplication.

So, determine the maximum step length along the direction line search r_0 as $\alpha_0 = \frac{r_0^T r_0}{r_0^T q_0}$. So, q_0 is the vector that we stored as matrix vector multiplication A r_0. So, for each subsequent iteration j let us say ith iteration. So, this is how it starts and for each subsequent iteration we keep on updating the search direction and compute the step length.

So, update the solution vector. So, previous solution multiplied by whatever_is the step length multiplied by the residual vector or the search direction corresponding search direction and that gives us the new updated solution and with this new updated solution we have we find what is the residual vector.

So, residual vector_is again related to previous residual vector and - this matrix vector multiplication and step length. So, that gives us the orthogonal residual vector to previous vector and if this residual vector new residual vector norm of new residual vector_is less than certain fraction of initial residual first residual some for some pre specified value of epsilon it can be very small number then we stop the iterations or else we proceed with the next iteration with updating the vector q. So, q_i for_ith iteration is given as A times r_i. So, residual of ith iteration and step length in ith iteration is computed as $\frac{r_i^T r_i}{r_i^T q_i}$ and we again go back to computing the next iterate next value of the residual vector and the updated solution. And the cycle proceeds until we compute the up to convergence.

So, this is a very descent improvement in convergence and I mean dramatic actually dramatic improvement in convergence compared to Jacobi or Gauss - Seideliterations and for particularly well behaved and well-conditioned system of equations with small condition number the method actually converges very very quickly.

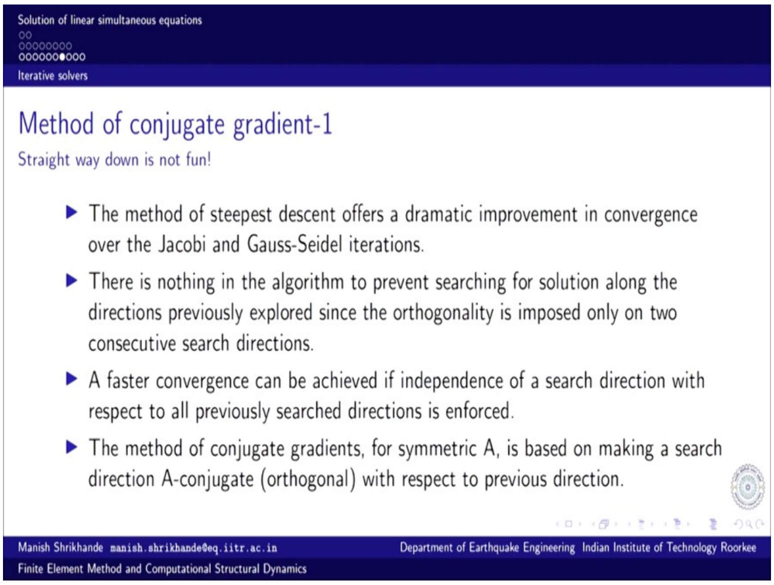
But there is a problem here that for some system it is quite possible for the method to get lock down into the two search directions only see the condition that we impose here orthogonality condition. So, this orthogonality condition is only for two consecutive search direction..

So, it may so happen that r_1 is orthogonal to r_0 and then r 2 will again be computed as orthogonal to r_1 which will be in the same direction as r_0 and then the improvement is not really as expected, because we have already traverse the direction of r_0 there is nothing more to be gained there, but there is nothing in the algorithm to prevent traversing in r_0 direction again, because the only thing that is imposed here is the new search direction should be orthogonal to the previous search direction.

Now, that previous search direction could have been could very well be what has been traversed earlier as well and if that happens then there is no way to improve upon the solution and it gets locked into these two direction. And that way the solution will actually suffer we may not actually go beyond certain limit, I mean after a while the residual will stop decreasing and once that happens if that is monitored then; obviously, there is a need to restart the entire process again.

So, to address this particular problem, that we may get logged into two directions to mutually orthogonal direction and we may alter and the algorithm may alternately switch between two directions and with no further_improvement in the computed solution. So, the key point is straight way down is not fun it may get locked I mean between two directions.

(Refer Slide Time: 18:00)



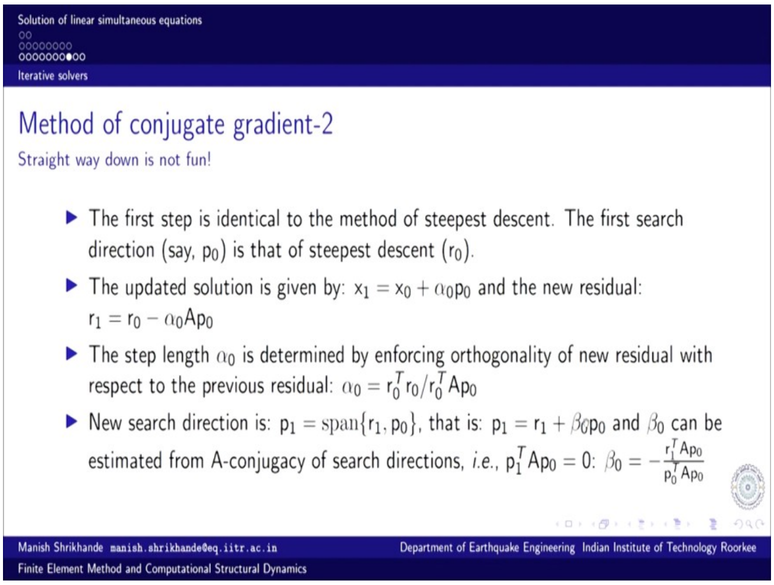
So, method of steepestdescent offers a dramatic improvement in convergence over the Jacobi and Gauss - Seideliterations, but there is nothing in the algorithm to prevent searching for solutions along the directions previously explored since the orthogonality is imposed only on two consecutive search directions and when we do that than previously search directions could be traversed again there is nothing to prevent that from happening.

A faster convergence can be achieved if independence of a search direction with respect to all previously search directions is enforced. So, if we have some mechanism in which the new search direction is guaranteed to be independent of what all has been done earlier then we are looking at possibility of developing a basis for expressing the solution and that is what is achieve that is what is aimed at in the method of conjugate gradient.

So, method of conjugate gradient for symmetric matrix if matrix A is symmetric that is a very simple algorithm for non-symmetric matrices also there are versions of conjugate gradient method available, but those are little more tricky little more difficult to understand, but for_if the matrix A is symmetric then it is the algorithm is very very straight forward.

So, it is based on making a direction A- conjugatewith respect to previous direction and how does it help. So, what is done is instead of saying that matrix or the direction r_1 is orthogonal to r_0 I would rather I rather_impose that direction new direction r_1 is orthogonal to A r_0, a very minor seemingly very minor adjustment, but with this has a lot of theoretical basis to ensure independence of the search direction.This simple A conjugation allows us to enforce that the new search direction is always guaranteed to be a direction which has not been explored earlier. So, the first step is identical to that of the steepest descent.

(Refer Slide Time: 21:10)



So, let us say first search direction let us say p_0 is same as that of the steepest decent that is b - A x_0 and updated solution is given by $x_1 = x_0 + \alpha_0 p_0$. So, α_0 is again the step length how much we should go along p_0 and the new residual is given by $r_1 = r_0 - \alpha_0 A p_0$. So, that is the first step. So, first step is common with steepest descent because that is the first operation we are doing.

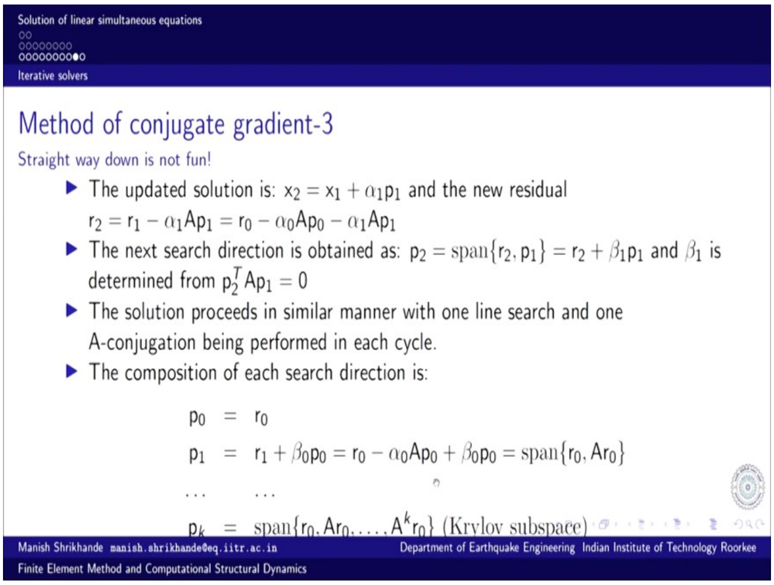
The step length α_0 is determined by enforcing orthogonality of the new residual with respect to previous residual. So, again first length this step length is determined in the same way as the steepest descent. So, no change here, except that we will in practical implementation we will be replacing A p_0 with vector q_0.

New search direction p_1 is taken in the span of r_1 that is it is going to be a linear combination of the new search new residual vector r_1 and previous search direction p_0. So, that is p_1 is going to be $r_1 + \beta_0 p_0$. So, that is a linear combination of 2 vectors. So, it is the spanning ofnew residual r_1 and the previous search direction p_0.

So, $r_1+\beta_0 p_0$ and β_0 can be estimated from the A-conjugacy of search direction that is p_1 has to be orthogonal to A p_0. So, that is $p_1^T A p_0$ should be equal to 0 and when we substitute this p_1 as this it gives us an equation for β_0 to compute this length factor β_0 as $\frac{-r_1^T A p_0}{p_0^T A p_0}$.

So, what has happened here, that is different from previous I mean steepest descent, it is this conjugation and the fact that previous directionis taken as the spanning span of the previous two residual vector and previous search direction and that we will see how that makes a difference.

(Refer Slide Time: 24:16)



So, the updated solution now looks like $x_2 = x_1 + \alpha 1 p_1$ and the new residual is now if I substitute it back and new residual is going to be b - A x_2. So, and x_2 is of course, this. So, I can expand this. So, $b - A x_1 + \alpha_{1A} p_1$ and A x_1, x_1 is again $x_0 + \alpha_0 p_0$. So, I can substitute it back and then if I expand it all in terms of A r_0 then it becomes r 2 is essentially what I havehere is $r_0 - \alpha_0 A p_0 - \alpha_{1A} p_1$ and how is that different what does it mean.

So, next search direction we will see how this expands how this term expands with each iteration. So, the next search direction p_2 is obtained as span or linear combination of r 2 and previous vector search direction p_1. So, search direction $p_2 = r_2 + \beta_1 p_1$ and

β_1 is again obtained from the A-conjugacy that is $p_2^T A p_1$ should be equal to 0 and that leads to solution β_1 is again obtained in the similar fashion as a ratio of two scalar products.

And solution proceedsin the similar manner with one line search and one A- conjugation being performed in each cycle so, composition of each search direction. So, this is what I meant when I said let us look at how this term expands. So, first search direction is r_0 second search direction is $r_1 + \beta_0 r_0$. So, r_1 is $r_0 + \alpha_0 A p_0 + \beta_0 p_0$ and that is essentially = r_0 and A r_0. So, p_0 is similar same as r_0 in this case..

So, what we have here is the new search direction is a linear combination of r_0 and A r_0. So, this A r_0 the linear combination hasforce this vector to be away from r_0 and this will keep on happening and this will keep on adding more and more powers of A, such that if I expand for the kth iteration the search direction is a span of a linear combination of r_0 A r_0 A square r_0 and so on up to A raised to the power k r_0.

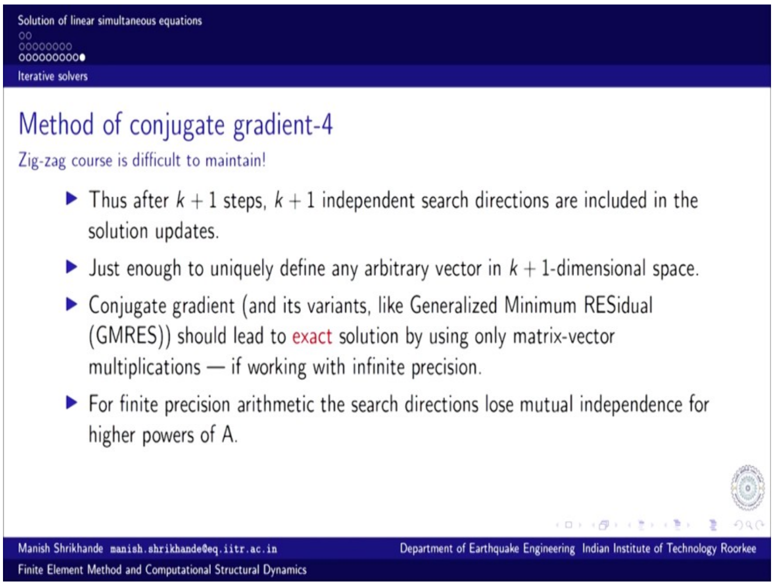
So, individually these are all independent vectors because this is essentially matrix multiplication of a vector_in force I mean that is like rotation of a vector. So, vector r_0 is being continuously rotated and therefore, these are all independent directions and this is what we know as krylov sub space..

So, in this by ensuring this A-conjugacy what we have been able to achieve here is that the search directions are always independent and the algorithm would then because; obviously, for an n dimensional system we do not need more than n number of base vectors.

So, the algorithm should converge in at most n number of iterations n number of steps. So, the moment we have n number of independent vectors here the solution is complete. So, there is no further_iteration, but unfortunately this is fine as far as theory goes and this is; obviously, independence works assuming infinite precision arithmetic is at work, but for floating point operations with large powers of matrix A the directions begin to lose independence and the convergence suffers.

So, that is an implication of limitation of floating point arithmetic nothing wrong with the algorithm of conjugate gradient, but because we have to implement this on a digital computer with its own vagaries of floating point operations the krylov sub space that we try to generate the base vectors may start losing their_independence and then the convergence slows down or convergence takes a hit.

(Refer Slide Time: 29:42)



So, what happens is after k + 1 I mean steps k + 1 independent search directions are included in the solution updates and so.Just enough to uniquely identify any arbitrary vector in k + 1 dimensional spaceand this is generally referred to as generalized class is Generalized Minimum Residual Methods GMRES and this should lead to exact solution by using only matrix vector multiplication.

Provided that we are working with infinite precision, but for finite precision search directions lose mutual independence for higher powers of A, we will see that in our next lecture how this happens losing independence for higher powers of A and the solution to prevent this or prevent this loss of independence is to use Gram Schmidt orthogonalization periodically to restore mutual independence of search directions.

So, after certain number of steps we orthogonalize all the vectors using Gram Schmidt orthogonalization and that will providethe previous I mean basic independence of the search direction. So, if there is any loss of independence because of higher powers of matrix A then those would be set right by using Gram Schmidt orthogonalization.

So, once we have this. So, Gram Schmidt orthogonalization on its own is a very expensive process. So, it should not be used indiscriminately. So, a judicious call is required to implement Gram Schmidt orthogonalization after certain number of steps have been taken and to set the independence of the vectors again in the solution process and with that the although initially when it was conjugate gradient method was proposed it was it led to a lot of excitementin the community that it looks like a iterative scheme, but it is a exact solution because theoretically you have as many base vectors as required to represent any vector.

So, you should be we should be able to compute exact solution, but the finite precision arithmetic they actually prevent the independence of the search direction in krylov sub space that is generated and gradually it was realised that fine there is a limitation due to finite precision arithmetic, but it can still be used as a iterative solver and that is with much better convergence characteristics than steepest descent and with this modification of Gram Schmidt orthogonalization thrown in between two subsequent directions and that will ensure the convergence of the gram conjugate gradient method.

And with that we close this discussion of Solution of Linear Simultaneous equations and we will now move on to the discussion of Eigen value problems. So, iterative schemes again I repeat they are very popular and very very powerful and particularly conjugate gradient and its variant they can make use of very efficient data structure using only non-zero elements of the matrix coefficient matrix stored and that can lead to very very effective and very very fast computational scheme.

Thank you.