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

Lecture - 48
The Algebraic Eigenvalue Problem - II

Hello friends. So, we have seen the power iterations in action for computing the how any arbitrary vector can be nudged towards the eigen vector corresponding to dominant eigen value. And we explained that this is why the search directions for higher powers of matrix A in Krylov sub space they tend to lose their independence. And they need to be re orthogonalized or made independent of each other by virtue of by using the Gram Schmidt orthogonalization process.

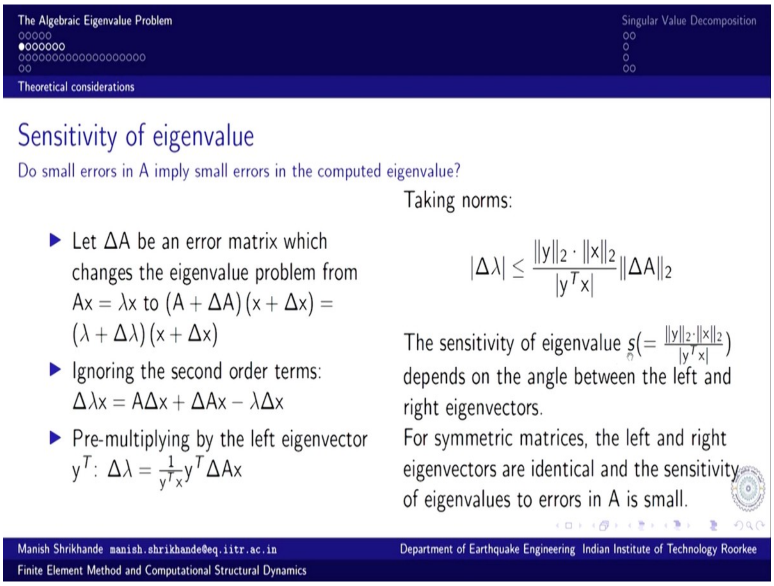
So, before discussing some of the algorithms for solution of this algebraic eigen value problem. Let us first go through the tools basic tools of the trade, the basic workhorse that are used in these solution of this eigen value problem, algebraic eigen value problem. So, first of all let us ascertain I mean, since we are working with the digital computers and we all know what are the problems with floating point operation and everything is approximate.

So, what is the sensitivity of the eigenvalue computing computed eigen value to the approximation? I mean, computed approximation because the matrix A that we are targeting for which we have we are trying to compute the eigen values and eigen vector that matrix A itself may be approximate. With that may not be exact representation of the matrix A that is actually that should actually be representative of the system because of floating point arithmetic and round off errors.

So, matrix A can still be can itself be in error. So, the question natural question arises do small errors in matrix A? It can be a good approximation, but it is a approximation nonetheless with some errors. So, do small errors in matrix A imply small errors in the computed eigen value, under what conditions?

If it is true if small errors in matrix A are they mean they imply small errors; I mean, the errors in the computed eigen values are also small. Then; obviously, the computation is robust otherwise we may need to take precaution. So, under what conditions does this happen?

(Refer Slide Time: 03:24)



So, let us derive this result, let us consider that ΔA is the error matrix. So, instead of A it is $A+\Delta A$ which is the true matrix representing the system. So, that will change the eigen value problem from $Ax=\lambda x$ to $(A+\Delta A)(x+\Delta x)$. So, $x\Delta x$ is the incremental change in the eigen vector.

Because for small changes in matrix A we do not expect very large changes in the eigen value solution. So, this is a small change in matrix a in the neighbourhood. So, eigen vectors will also be in the neighbourhood and eigen values will also be it be in the neighbourhood. So, $x+\Delta x$, so $(A+\Delta A)(x+\Delta x)=\lambda+\Delta\lambda$.

So, small increment in eigen value multiplied by the incremental eigen vector now let us work out these expansion and this results into this change in eigen value. So, λx because this is what we are interested in small errors in the computed eigen value. So, what is the change in eigen value? So, $\Delta\lambda x=A\Delta x+\Delta A x-\lambda\Delta x$.

And we are ignoring the higher order terms. So, product of small terms for example, product of ΔA with the $\lambda\Delta x$ is ignored and product of $\Delta\lambda$ with Δx is ignored. And by invoking the orthogonality we pre-multiply by the left eigen vector. So, we have $\Delta\lambda = y^T x$. So, this will be scalar and that is what we take it in the denominator here.

So, for the eigen left eigen vector for the same eigen value, so not orthogonality. So, of the same eigen value, so we multiply this by $y^T A \Delta A x$ and this is scaled by $y^T x$ that is the scalar. And we take the norms what happens is if we take these norms then $\Delta\lambda$ will be equal to; I mean, now we can look at the individual terms. So, this norms will be equal to product of individual norms that will be less than individual norms.

So, $\Delta\lambda$ the magnitude of the error in eigen value λ will be equal will be less than the norm of y_2 multiplied by norm of x_2 . So, Euclidean norm divided by the norm; I mean, the whatever is the magnitude of the scalar $y^T x$ and then this second norm of this matrix ΔA . So, this will be the spectral norm.

So, largest eigen value of ΔA . So, sensitivity of eigen value s this is referred to this sensitivity of the eigen value. So, this is what scales the error this is the measure of error in matrix A. So, this error in matrix A is scaled by this constant and this is called the sensitivity. If this sensitivity is small then the error in the corresponding eigen value is also going to be small. And this depends on of course, this is of course, if we scale then scale to unity then this will be of unit length. y and x they can be scaled to unity, so they will be of unit length.

So, this scaling sensitivity factor depends on this denominator y transpose x. So, what is this y transpose x? This is the projection of y along x, so this is the cosine of the angle between these two vectors. And value of the cosine function varies between + 1 to - 1 and it will be + 1 or - 1 only when the angle between the two vectors is 0 or 180 degree.

So, the only way this becomes equal to 1 that is when the sensitivity of the eigen values would be would not be; I mean, the error in eigen values will not be amplified by a factor greater than 1 with respect to the error in the matrix A. So, that is when y and x if they are identical that is when the left eigen vectors and right eigen vectors are identical.

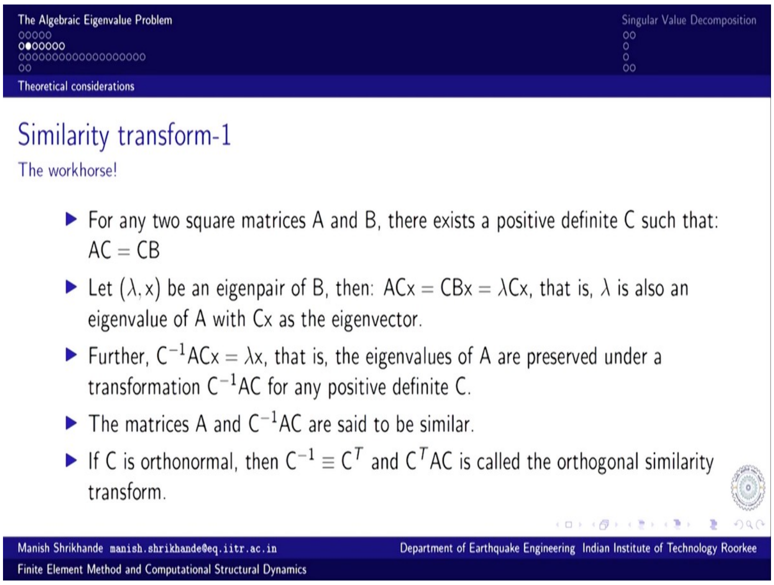
So, if they are identical then the that angle is going to be 0 angle between them is going to be 0 and this product y transpose x is going to be equal to unity. Because these are the cosine between the two vectors is going to be unity. And therefore, we can conclude that for symmetric matrices the left and right eigen vectors are identical and the sensitivity of the eigen values to errors in A is small. And that is why it is important; I mean, in the beginning we emphasize that generalised eigen value problem can be converted to standard eigen value problem.

But this trivial procedure of computing B^-1 A and writing that as a standard eigen value problem is not a preferred technique, preferred way. Because of this reason it will increase the sensitivity of the computed eigen values to the errors in the matrix of; I mean they operator matrix the B^-1 A. Because $B^{-1}A$ is not going to be symmetric, is not guaranteed to be symmetric and left eigen vectors will be in general different from the right eigen vectors.

And then the sensitivity will be greater than 1. So, that is the basis for doing all that jugglery to convert generalise eigen value problem into standard eigen value problem instead of going through the straight root of $B^{-1}A$. So, this we discussed only to emphasize the need for that Cholesky decomposition of matrix B and always maintaining the symmetry of the matrix if the original matrices are symmetric to begin with. That is a very very valuable property and we should not lose that in computation.

So, symmetry is always a symmetry of the operator matrices is always a desirable property for any numerical computation. Now, we come to one of the very powerful tools of eigen value computation that is called similarity transform. So, similarity transform refers to a transformation of these system matrices in such a way that although the matrix may change, but its eigen values are not changed.

(Refer Slide Time: 11:59)



So, the basic idea is if we have two matrices, two square matrices A and B. Then there will exist there exists a positive definite matrix C such that A C = C B that can be proven. And if C is positive definite then; obviously, I can write C^-1 A C = B. And if I can do that, so if λ and x they are the eigen values and eigen vectors of matrix B then I can write this as AC x = CB x by post multiplying with x and $CBx=\lambda x$.

So, $Bx=\lambda x$, so that will be λCx . So, that is λ is also an eigen value of A with C x; I mean, if I can look at this as a vector C x C times x is an is a vector. Then this is simply the eigen vector of a and λ is also the eigen vector of eigen value of matrix A. So; that means, I can take C from right hand side to left hand side.

So, $C^{-1}ACx=\lambda x$ so; that means, that eigen values of a are preserved under the transformation $C^{-1}AC$ for any positive definite C right. So, λ is the eigen value of A and if I can find some matrix $C^{-1}AC$ where C is a positive definite matrix then this $C^{-1}AC$ also possesses the same eigenvalue λ .

And that is a very powerful result because I can nudge matrix A to a form by similarity, by using these similarity transform to a form which is which allows for easier computation of the eigen values and eigen vectors. I mean, we can find at the whole point is finding the determinant; I mean, by definition it is the determinant should vanished. And the easiest way to find the determinate if the simple structure, simplest matrix for which determinant is easiest to compute is the diagonal matrix.

So, if I can somehow arrange the sequence of operation such that $C^{-1}AC$ gradually changes to a diagonal structure then the problem is solved, we have computed all the eigen values of the system. And this basic idea forms the basis of several eigen value computation algorithms.

But as I said earlier I have been emphasizing on this aspect repeatedly C^{-1} ; I mean, this inverse of matrix C should never be computed. I mean, we should never be asked to compute inverse of a matrix that is too much of an effort and too error prone. Because it requires lots of floating point arithmetic operations and each floating point arithmetic operation incurs a round off error.

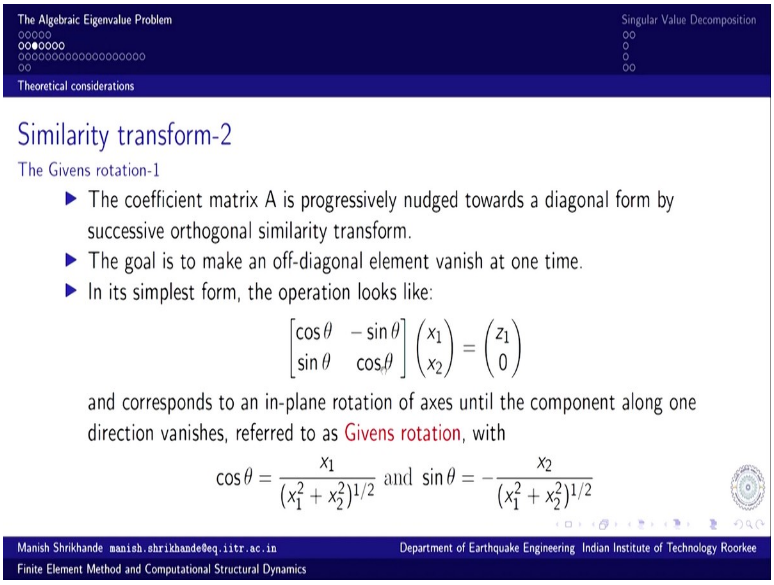
So, this C^{-1} computation should be avoided. So, the solution is of course, if I choose C as an orthonormal matrix orthogonal matrix right. So, if I choose C as an orthonormal matrix then the inverse of C is computed is given by the transpose of C. So, C^T becomes its inverse. So, I can define similarity transform by using a transpose of a matrix orthonormal matrix.

So, I have to choose the matrix C in such a way that it is comprising of orthonormal representation. And because the eigenvalues of these two matrices are similar are identical these two matrices A and $C^{-1}AC$ they are said to be similar. And if C is orthonormal then C^{-1} is identical with C^T and C^TAC operation is called orthogonal similarity transform.

So, if C matrix is an orthonormal matrix then C^TAC is referred to as orthogonal similarity transform. And that forms the backbone of most of the algorithms for solving eigen value problem as we will see. So, the first example of this orthogonal transformation is what we refer to as Givens rotation.

Now, this Givens rotation is similar to what we have in the coordinate transformation for reorientation of frame element local coordinate system to global coordinate system we will see that. So, something similar to rotation matrix that is orthogonal matrix.

(Refer Slide Time: 18:14)



Coefficient matrix A is progressively nudged towards a diagonal form by successive orthogonal similarity transform. And the idea the goal is to make an off diagonal element vanish at one time. So, what happens is we define I mean we have some elements let us say x1 is on some diagonal and x2 is an off diagonal element in the same column. So, let us just look at these two elements and it is pre multiplied by this operation.

So, cos theta - sin theta sin theta cos theta. So, this is what if you recall this is what we had for transformation of coordinate system from local coordinate to global coordinate in trust problems or even for frame problems. So, we try to map this as our objective is to make this off diagonal term x2 to vanish. So, that this is the result, so we want this term to vanish. And if we want this term to vanish that is possible when I look at this second line of equation $x_1 \sin\theta + x_2 \cos\theta$ should be equal to 0.

And that gives me the result by knowing this x1 and x2 I find out what should be the values of cos theta and what should be the value of sin theta which will give me this result right. So, if it is $x_1 \sin\theta + x_2 \cos\theta = 0$ then the only if the simpler arrangement is I just make x1 or $\sin\theta$ to be proportional to - x2 and $\cos\theta$ to be proportional to x1 and then it should be done.

And that is what is done here problem is solved by using $\cos\theta$ defining

$$\cos\theta = \frac{x_1}{\sqrt{x_1^2 + x_2^2}}$$

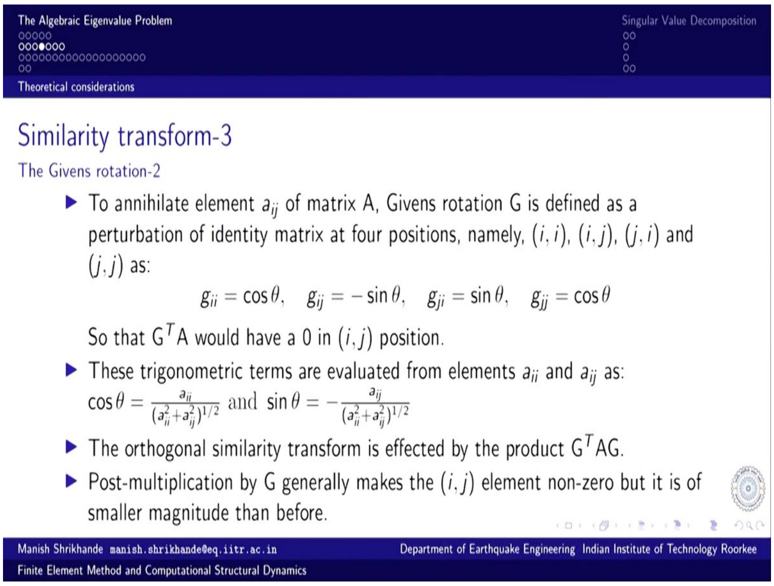
and $\sin\theta$ as

$$\sin\theta = \frac{-x_2}{\sqrt{x_1^2 + x_2^2}}$$

With these choices of $\sin\theta$ and $\cos\theta$ if we evaluate then this problem will be solved.

So, the off diagonal term x2 is the off diagonal term and that will be solved, that will be annihilated. And that is the objective gradually we need to nudge the matrix to a diagonal form. So, if we call this as a, so this forms the basis of Givens rotation.

(Refer Slide Time: 20:55)



So, Givens rotation is essentially an modification of an identity matrix with these four terms cos theta - sin theta and sin theta and cos theta they occupying, they only modify the four places of the identity matrix. So, i i, i j, j i and j j and this is to annihilate the element i j of matrix A right.

So, identity matrix rest of the matrix rest of the identity matrix remains same we only look at i i, i j, j i and j j. And these coefficient actually go as i i element goes as $\cos\theta$, i jth element goes as - $\sin\theta$, j i goes as $\sin\theta$ and j j goes as $\cos\theta$. So, g i i

$\cos\theta$, g i j - $\sin\theta$, g j i $\sin\theta$, g j j $\cos\theta$. So, that $G^T A$ would have a 0 in i jth position.

So, if we multiply this then i jth position would be vanished. So, this trigonometric functions they are terms are evaluated based on the same evaluation that we had in terms of x1, x2. Instead of x1, x2 we now have a_ii and a_ij corresponding elements of the matrix and these cos theta and sin theta are evaluated and the cosine the Givens rotation matrix is defined.

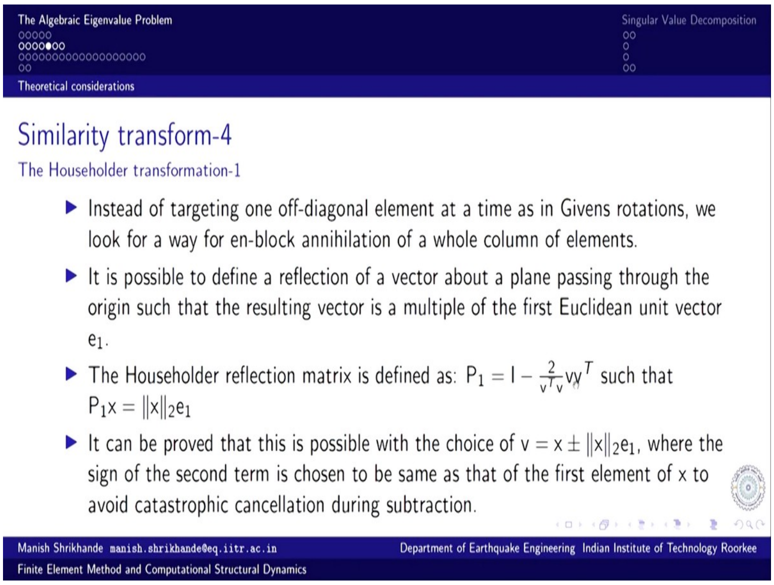
Now, to complete the similarity transform we have to post multiply it with G that would be the complete similarity transform. So, after that would be orthogonal similarity transform that is affected by the product $G^T AG$. Now, $G^T A$ has annihilated i jth position to be 0, that has first i jth position to be 0 this post multiplication by G may again make it non zero. But that magnitude wise it will be smaller than the original value.

So, post multiplication by g generally makes i jth element non zero, but it is of a smaller magnitude than before. So, what we are doing is we are shifting the mass of the matrix towards the diagonal. So, that gradually the entire mass of the matrix would be accumulated towards on the diagonal of the matrix. And those diagonal elements would be the eigen values of the matrix, given matrix A.

So, this is Givens rotation which actually targets one element at a time to make it vanish. Another powerful technique is called Householder transformation. In; I mean, in contrast with the Givens rotation where we target only one element at a time Householder transformation actually targets one whole column at a time. So, it when makes one whole column to vanish in one similarity transform.

So, it is possible to define a reflection of a vector about a plane through the origin such that resulting vector is a multiple of the first Euclidean unit vector e 1. So, e 1 is 1 0 0 0 0, so that is the starting vector we start with.

(Refer Slide Time: 24:54)

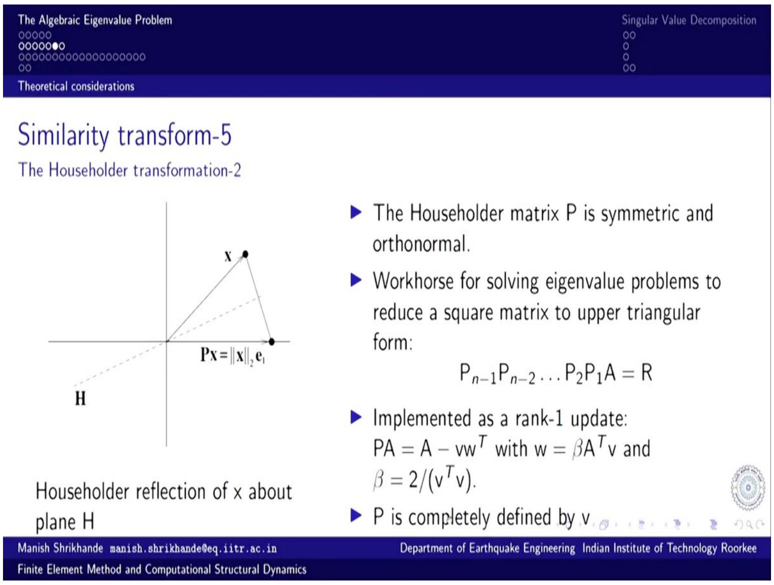


So, Householder reflection matrix is defined as $P_1 = I - \frac{2}{v^T v} v v^T$ such that P_1 when multiplied by x it will make it proportional to e_1. So, that except for the first element all the elements of this vector would vanish very powerful technique.

So, instead of Givens rotation which looks at which operates element by element, one element at a time Householder matrix would make all elements, but one to vanish. And the way to do that is by suitable choice of this vector; I mean, this vector v can be chosen such that x it can be + or - using this scalar multiple; I mean, the norm of length of vector x multiplied by the Euclidean first Euclidean unit vector e_1.

So, sign of this term is chosen to be same as that of the first element to avoid catastrophic cancellation during the subtraction operation. So, if it is of the same sign as the original vector then there is no chance of incurring any errors due to this catastrophic cancellation.

(Refer Slide Time: 26:30)



So, this is essentially the Householder matrix. So, this is the plane and this P matrix is causes the reflection of this matrix along about this plane on to the first Euclidean vector. And Householder matrix P is symmetric and orthonormal and this is a workhorse we can keep on multiplying this repeatedly. So, first column we will make it; I mean, vanish all the elements except the first, second matrix P_2 it will make vanish all the elements except the second diagonal and so on.

So, this way we can actually make the convert the square matrix to upper triangular form very easily in a very very quick manner. And they are also orthogonal matrices, so they are implemented they can be implemented as rank one update. So, this is actually a rank one update.

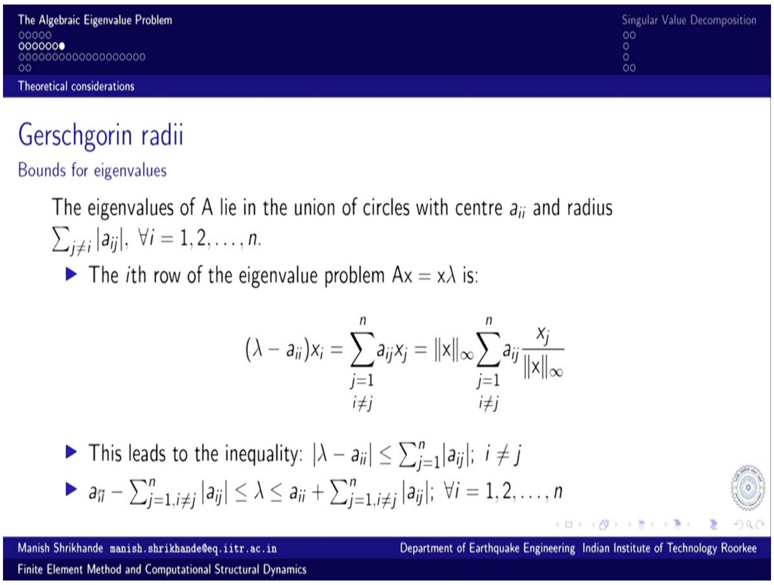
So, P A is implemented as $A - v w^T$ and w; I mean, these are all numerical implementation issues for efficient computation. And this matrix P if the transformation matrix is completely defined once we define this vector x and vector v and which is defined which is completely defined by the vector x which is to be reflected.

So finally, I mean at the beginning of the discussion of iterative schemes we said that we can identify what is the largest eigen value of a matrix A just by inspection and that is defined by Gerschgorin radii. So, we can impose bounds on the eigen values. So, we can find the range between which at least one eigen value of the matrix will lie. And once I have this range by using this Gerschgorin radii Gerschgorin radius there will be as many radius, as many disks as the number of rows in the matrix.

So, for an n n by n matrix there will be n number of disks. So, I can get the largest disk and I can get the estimate of the or bounds on the eigen values that eigen value will lie between these. And I can choose the largest bound upper bound and that can be used for construction of the matrix G that that we define for construction of iteration scheme for Jacobian Gauss–Seidel iteration and that will ensure convergence of the matrix.

So, we wanted that matrix B^{-1} to be equal to $\frac{2}{\mu}\times I$ and μ is the estimate of largest eigen value.

(Refer Slide Time: 29:41)



And that largest eigen value estimate can be obtained from this Gerschgorin radii. And that = simply the diagonal element of the matrix A of each row and multiplied added to the of some of all diagonal elements. So, basically the idea is eigen value is bounded by the diagonal element of any row - the sum of off diagonal elements absolute value of some of all the off diagonal elements and the diagonal element + some of all off diagonal elements absolute value.

So, these are the bounds that are on the available on the eigen values and the larger of these can be computed, can be worked out as the estimate for choosing the matrix for iteration. And that will complete our discussion, that will facilitate convergence of the iterative scheme. So, we stop here and next lecture we will discuss about specific techniques of eigen value computation by using these tricks; I mean, basic tools of computation.

Similarity transform, Givens rotation, Householder transform and also Gerschgorin radii, so that provides us initial estimate of the eigen value. And using these we will compute we will discuss some of the very powerful numerical algorithms for solution of eigenvalue problems.

Thank you.