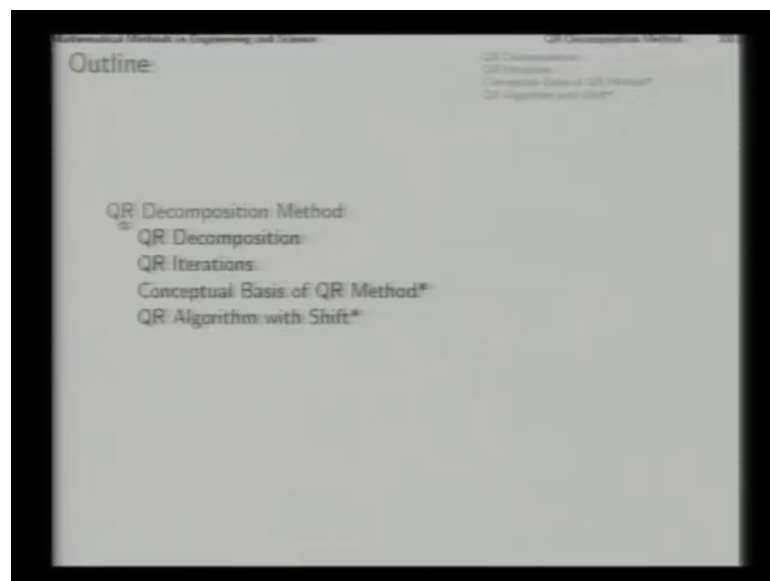


Mathematical Methods in Engineering and Science
Prof. Bhaskar Dasgupta
Department of Mechanical Engineering
Indian Institute of Technology, Kanpur

Module – II
The Algebraic Eigenvalue Problem
Lecture – 05
QR Decomposition, General Matrices

In the previous 2 lectures we have considered two ways of finding suitable similarity transformation for solving the algebraic eigenvalue problem. In this lecture we will briefly discuss the other two. The first that we discuss now is method based on matrix decompositions or factorizations.

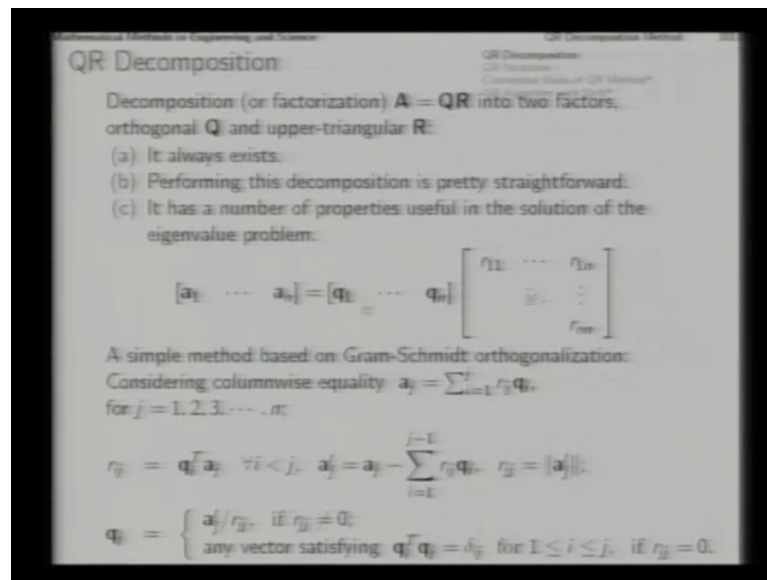
(Refer Slide Time: 00:41)



The most important matrix decomposition for eigenvalue problem is the QR decomposition.

In these 2 chapters of the book you will find that there are quite a few sections with asterisk. These star marked sections are a little advance, so we will not cover these topics in depth and rather we will try to concentrate on methods so far as its use is concerned.

(Refer Slide Time: 01:24)



First idea of QR decomposition, the decomposition itself is quite straight forward the QR decomposition is the decomposition of a matrix into 2 factors Q and R in which the first one the Q factor is orthogonal and the second one the factor R is upper triangular. The good point about QR decomposition is that it always exist whatever matrix you give one can decompose it into Q and R factors. So, as long as you give a square matrix the Q and R factors will be square. So, for every square matrix you have QR decomposition, it will always exist.

Next performing this decomposition is pretty straight forward. In fact, if you have come to this point through the exercises in the previous chapters of the textbook then you would have already decomposed one matrix into Q and R factors. In chapter 4 one very simple QR decomposition problem was a set for you as an exercise in which the steps for the decomposition were also suggested. And the third important issue in QR decomposition is that it has a number of properties which are extremely crucial for the solution of eigenvalue problem.

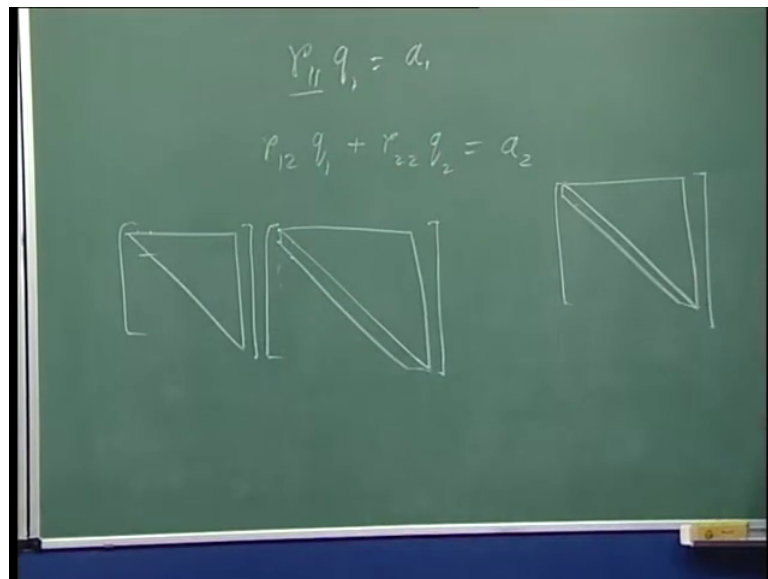
First we see how we conduct the QR decomposition of a given matrix.

Say this is the matrix a in which a 1, a 2 etcetera are columns. So, this is to be decomposed into Q and R factors this matrix is orthogonal; that means, its columns q 1, q 2, q 3 are all unit vector vectors which are mutually orthogonal and this is an upper triangular matrix below the main diagonal everything else is 0. Now first we quickly

review the same process of affecting this decomposition which was suggested in the exercise of chapter four, this method is based on the procedure of Gram Schmidt orthogonalization and Gram Schmidt orthogonalization process itself is actually another very simple algorithm which was introduced even earlier in the exercise of chapter 3.

Now, here we again just like our normal decomposition processes we try to determine this decompose parts through term by term multiplication. So, first we then write a 1 as the first column of this matrix product and that will be this matrix multiplied with the first column, now that will give us $r_{11} q_1$ plus everything else is 0; that means, we get, $r_{11} q_1$ as the first column of the matrix a .

(Refer Slide Time: 04:33)



Now, the matrix q is supposed to be orthogonal; that means, this vector column vector q_1 should be a unit vector and that will mean that the entire magnitude of this vector a_1 should be r_{11} . So, what we do? We take the norm on both sides and we find that the norm of a_1 is r_{11} . Once we find r_{11} then we can divide this vector a_1 by r_{11} that gives us the unit vector q_1 . Precisely this step you will find when you try to conduct this whole thing for j equal to 1 plus, this part this part is missing for j equal to 1 because there is only 1 term in the sum and there is no previous i . So, this terms is this part is missing. We start from this part, here also this sum is completely missing and you have this a_j itself a_1 itself sitting in the place of a_j prime and then r_{11} is this norm as I just now mentioned and when you divide that column vector a_1 with r_{11} you get q_1 .

After that you equate the second column's second column a_2 is r_{12} into the first column here plus r_{22} into the second column here right, other terms are 0. So, from here the actual work starts. So, r_{12} into q_1 plus r_{22} into q_2 is the second column you do not know anything on this side except q_1 on this side you know a_2 . Now, if you know q_1 and you also know that q_2 is supposed to be orthogonal to q_1 then both sides if you take inner product is q_1 that is you multiply this both sides of the equation with q_1 transpose then here q_1 transpose q_2 will become 0 and here you will have q_1 transpose q_1 is 1 and then what will remain on this side is r_{12} and that is q_1 transpose a_2 that is what you find here.

For j equal to 2 only i equal to 1 is possible and r_{12} you get as q_1 transpose a_2 . After you have found that, after you have found that you can subtract that from a_2 and in this particular case this summation will include a single term this q_1 transpose a_2 . After you remove that that is this r_{12} into q_1 this part r_{11} to q_1 . Now that you have got r_{12} r_{12} q_1 you subtract from a_2 what you are left with is this that is demarked at here as a_2 prime, and if you are left with this thing only then its norm will give you r_{22} right its normal give you r_{22} . Once you have found r_{22} then you can divide the remaining thing a_2 minus that stuff with r_{22} and you get the unit vector q_2 that is here. This way you keep proceeding.

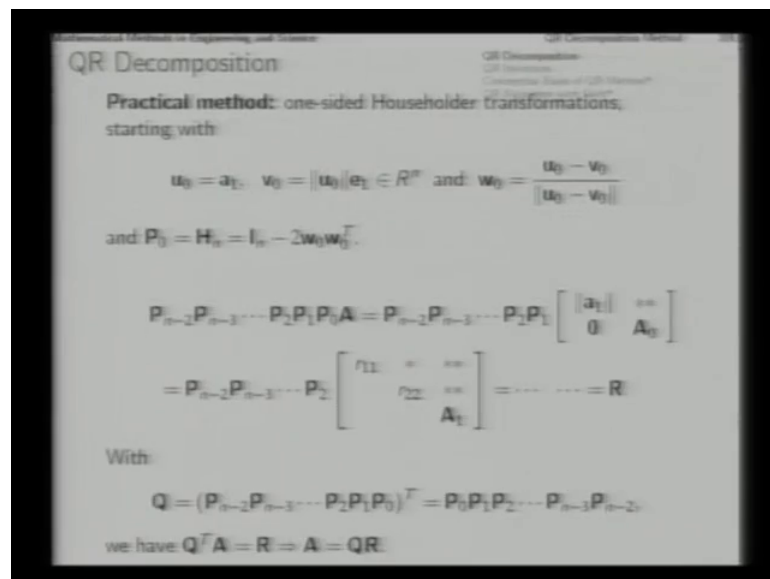
So, in the next step you will find r_{13} , r_{23} and then the 2 terms subtracted from a_3 will give you r_{33} into q_3 its not will give you r_{33} and when you divide with r_{33} you will get q_3 . The trouble will arise if you find that r_{kk} at the k th point if r_{kk} turns out to be 0 then what. So, you cannot divide this still r_{kk} equal to 0 is a valid output which you will place in this location wherever in the diagonal entry it is supposed to come. That will signify that the given matrix is singular.

Apart from that so far as the question of determining q_k is concerned it is not a problem at all because q_k in the sum for a_k is actually not making a contribution because it is suppose to get multiplied with 0 to give the contribution. The trouble started because this entire vector a_j prime are turned out to be 0. At that stage you cannot divide it like this and therefore, you cannot get q_k by this formula, but that does not means that the process stops there. This r_{jj} or r_{kk} being 0 only means that q_k is left unconstrained, q_k expression is not available from here and; that means, that as long as q_k satisfies that column satisfies the orthogonality requirements it is fine.

So, what you do, till now whatever columns of matrix q you have found q_1, q_2, q_3 upto q_{k-1} any new unit vector which is orthogonal to all this $k-1$ all column vectors is acceptable as q_k . So that means, that r_{jj} is non zero then this is a constrain based on which you determine q_j , if r_{jj} is 0 then that constraint is revoked and in that situation any vector satisfying this orthogonality requirement is acceptable for q_j . With this process till the end you can go and decompose the matrix into Q and R factors.

Now, this is a very simple straight forward method by which can you effect the QR decomposition. However, I should mention for record that the sophisticated or practical methods which are utilized in most of the computational algorithms is actually not this, though this is a valid method, but actually the your decomposition is affected in most professional programs is through householder transformations. The same house holder transformations which we studied in the previous lecture, but with a little difference.

(Refer Slide Time: 11:28)

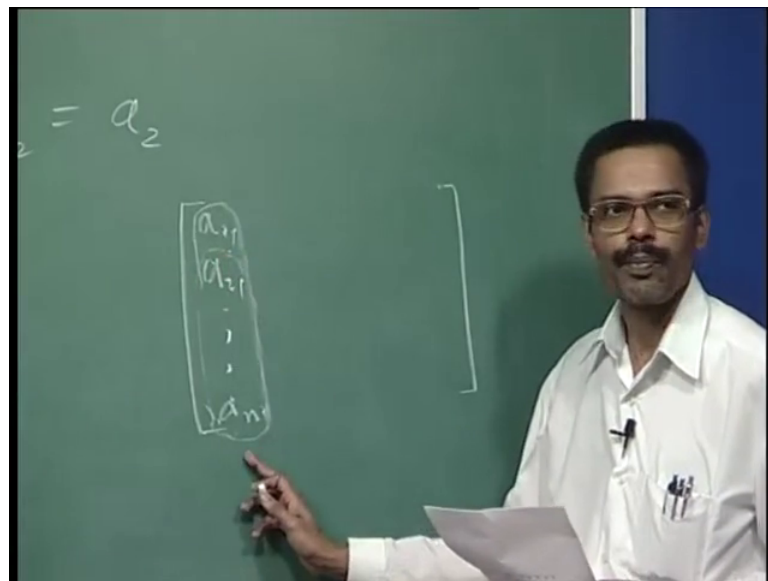


Here there are 2 small differences compared to the way we applied household transformation in the previous lecture. In the previous lecture we were using householder transformation matrix on both sides left as well as right because we were concerned with similarity transformations. Right now we are concerned with actually factoring a given matrix into 2 factors Q and R we are not yet so much concerned with a similarity transformation will be talking about similarity transformation based on this a little later, but right now our focus is on effective a QR decomposition and therefore, there is no

compulsion for us to multiply on both sides to maintain the similarity, our main focus is to effect a QR decomposition is one point.

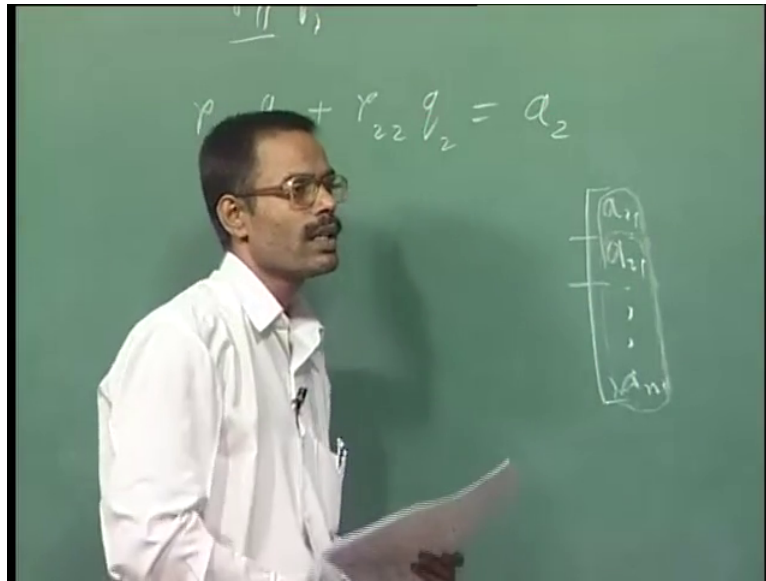
Second point we arrive at when we consider this situation that we apply a householder transformation on a given matrix in which we do not consider, we do not consider this much as u , but because of this whole thing as u .

(Refer Slide Time: 12:37)



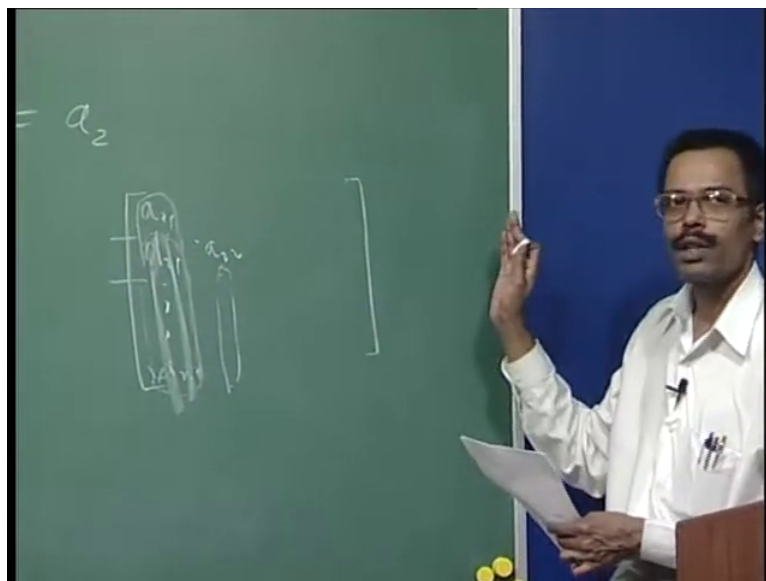
If you consider this whole thing as u and work out the corresponding v as we did in the previous lecture, then what happens is that v will have its first entry same as the norm of this entire vector from top to bottom and all other entries below that will be 0. That will mean that contrary to the case of the previous lecture where this much of the first column was converted to 0 was annihilated now this much will be annihilated in the first step that is why here I am saying not u_1 , but u_0 .

(Refer Slide Time: 13:21)



So, u_0 is the entire first column. v_0 is a matrix of the same a vector of the same dimension with only the first entry being non zero and having a magnitude same as the norm of u_0 . And based on that we work out w_0 and then h_n and therefore, P_0 in this manner and when we apply that only on the left side not on the right side, then from here to here all of these become 0.

(Refer Slide Time: 14:08)



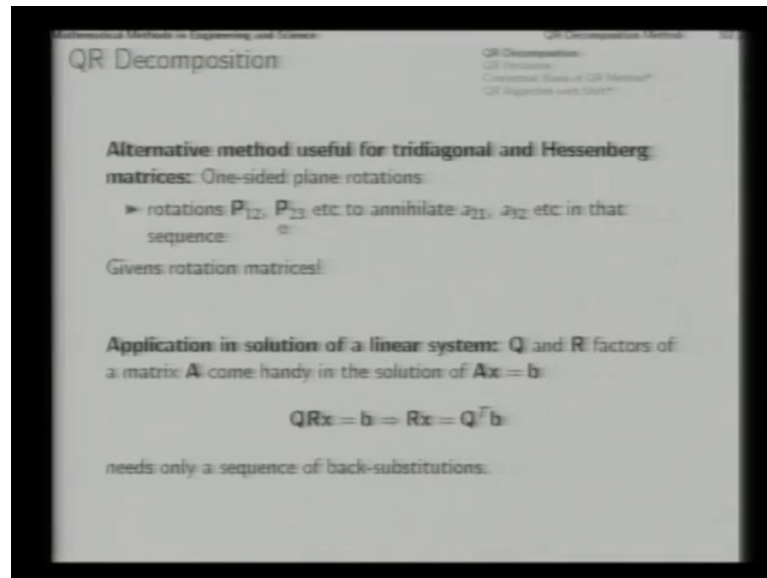
And next we will apply P_1 based on $h_n - 1$ to be operating on this much to get everything below a_{22} to be 0 and so on. Note that this same thing we could not do in the

previous case when we were trying to effect similarity transformation because the way the left side multiplication with the householder transformation matrix effect these entries and these entries if over the full thing. So, the similarity transformation in the previous case would require us to multiply the householder transformation that is on this is also that would spoil the 0s set in the left side multiplication. And that is why in the case when we were using householder transformation for tri diagonalization then we did not want the already established 0s to be spoiled by post multiplication that is why there we left 1 extra term free and applied householder transformation with 1 dimension less, but here we are not going to multiply the same matrix from this side we are applying the householder transformation only 1 sided, so we can take the 2 vector and therefore, under the diagonal all terms we can make 0.

So, with one step we with the u_0 , u_0 and therefore, with P_0 we get the subdiagonal entries in the first column 0 and then we get this the entire magnitude of a 1 the first column sits here below that everything else is 0. And then P_1 is applied which will make a_{32} , a_{42} , a_{52} etcetera upto an a_{n2} all 0 and so on that will look like this. Here there will be some non zero i term and so on and then we apply P_2 to get everything under a a_{33} , a_{43} as 0 and so on. Like that by the time we come up to P_{n-2} then we have got a completely upper triangular matrix.

Now, note that all this transformations from P_0 upto P_{n-2} are orthogonal matrices and therefore, this entire product P_{n-2} to P_0 is orthogonal, product of orthogonal matrices is also an orthogonal matrix. And now if we call Q as the transpose of this whole thing which is this then whatever we have pre multiplied can be called q transpose. So, what we have got we have got Q transpose a as r upper triangular matrix which means A is equal to QR and Q we have got stored by cumulative multiplication of the intermediate householder transformation matrices. So, this is one practical method of effecting a QR decomposition.

(Refer Slide Time: 17:33)

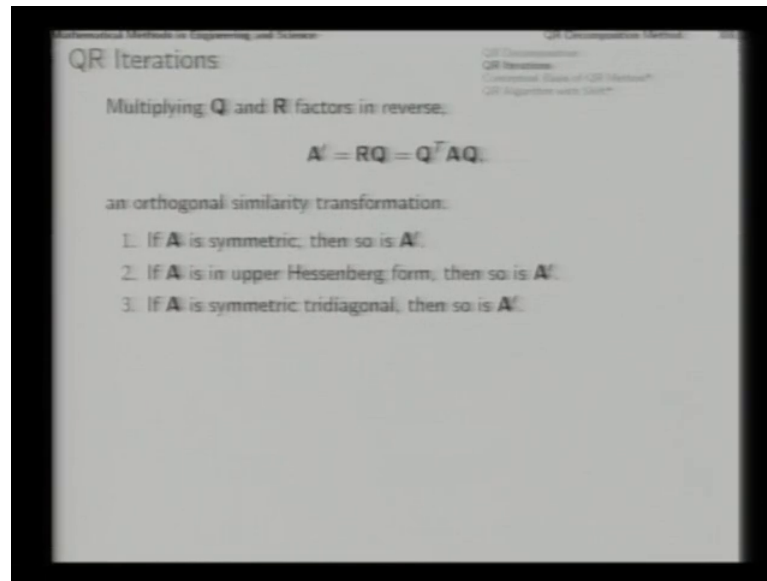


There is one more based on the givens rotations through one sided plane rotations.

Now, before going into the application of QR decomposition in eigenvalue problems let us quickly have a look quick look at a side ratio. If you already have the Q and R factors of a matrix then even in the solution of a linear system of equations Ax is equal to b it can be utilized because in case of A if we write QR with Q and R factors already available in our hand then first step is pre multiplication with Q transpose that will mean $R x$ equal to Q transpose b . And transposing A matrix and multiplying that to a vector is a task of very little computational effort. After that what remains is Rx equal to a known vector R being an upper triangular matrix; that means, only a sequence of back substitutions will give us a solution. The solution is actually extremely cheap if we already have Q and R factors in hand, but otherwise in order to solve this problem Ax equal to b typically you would not go into the process of Q and R , QR decomposition method because other methods that we have studied earlier in the course are comparatively better.

The actual use of QR decomposition algorithm is in solution of eigenvalue problem.

(Refer Slide Time: 19:11)



Let us study the decomposition in that light. First point is that if we multiply the Q and R factors in reverse this is a very weird proposition to begin with we have decomposed a matrix into QR factors and the Q and R appear Q first R next and now we are suggesting that let us see what we get if we multiply them in reverse RQ. It a weird proposition to begin with, but if we do that then we notice something interesting.

Just now we have seen that R is Q transpose because A is QR. So, R is Q transpose. So, if in case of R we write Q transpose A then see what we have got, Q transpose AQ wow, what we have got is a similarity transformation. That means, that if we factor a given matrix a into Q and R factors and then multiply them in reverse then what would happen is that would have a matrix new matrix A prime which is the result of a similarity transformation over a itself the similarity transformation matrix being this same Q.

So, this is similarity transformation not only that it is an orthogonal similarity transformation because Q is orthogonal with a few interesting properties. First is that if A is symmetric then A prime is also symmetric that we can see because we have multiplied from both sides u and Q transpose that results the symmetry.

Second is if A is in upper Hessenberg form then A prime is also in upper Hessenberg form. That you can see very easily because in the factorization of A into Q and R factors u 1 the first column of Q, Q 1 must be in the direction of a 1. Now if the original matrix a is in upper Hessenberg form which is this; that means, if the first column of the matrix A

has only the top 2 terms then first column of Q also will have only top 2 terms below that everything else is 0.

Next the second column of a has 3 term right, but then that is the linear combination that is a linear combination of the first 2 columns of Q . Now first 2 columns of Q through a linear combination is giving us the second column of a and the first column is only having 2 top terms. So, the second column can have only 3 top terms which are non zero and so on.

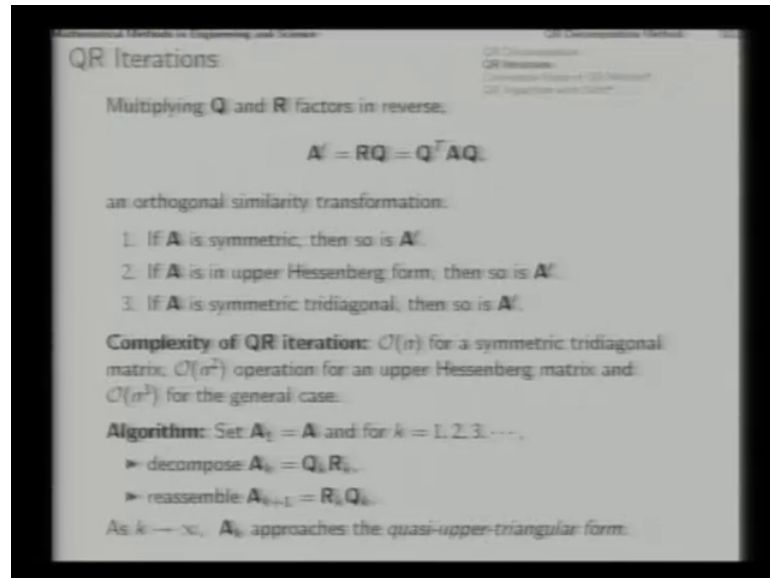
So, that way proceeding that way we find Q also will have the same shape as a , if a is assembled, so if a is assembled then Q is also assembled. Now in the reverse multiplication what is happening is that we are multiplying RQ right, RQ it is this. So that means, this matrix the first column of this matrix having only top 2 entries will mean that the first column of the product will have only top 2 entries because the this matrix multiplied with the first column gives us the first column of the product. So, this number into first column plus this number into second column and first column has only 1 entry and second column has only 2 entry at the top which are non zero. So, the composition will have 2 top entries as non zero

Next the second column of the product will be this 3 things into these 3 columns and these 3 columns have upto only third entry. So, the second column will have only top 3 entries which are non zero and so on. So, that way we will find that if the original matrix is assembled then so is Q and so is RQ . So that means, if A is in upper Hessenberg form then so is A prime. And now symmetry preservation and upper Hessenberg form preservation together will mean that if the original matrix is symmetric tridiagonal then the new matrix is also symmetric tridiagonal.

So, till now we have found through these properties that through this transformation that is factoring QR and then pre multiplying in the multiplying in the reverse order RQ nothing is going wrong, that is if you have given a matrix to begin with we have got a similarity transformed form of it. If we have given a symmetric matrix then we get back another symmetric matrix, if we have given an upper Hessenberg matrix we get back an upper Hessenberg matrix, if we have given a symmetric tridiagonal matrix then what we get back is also a symmetric tridiagonal matrix. That means, in the form and feature of the matrix nothing is spoiled nothing is lost, but the question is what is gained and in that

we find that there is a deep result which shows which establishes that if we follow this algorithm in one iteration not much will take place, in one step like this not much will take place, but the advantage accrues when we follow this repeatedly.

(Refer Slide Time: 25:17)



What is the algorithm? Algorithm is very simple set A_1 is equal to A and then for k equal to 1 2 3 4 5 6, go on doing these 2 steps. Decompose A_k in to Q_k, R_k factors and then multiply in reverse call it $A_{k+1} = R_k Q_k$. So, algorithm is just 22 step algorithm.

Factorize, multiply in reverse what you get? Factorize that multiply that in reverse and so on. So, from A_1 through these 2 systems you get A_2 , again from that through these 2 steps you get A_3 and so on, you go on doing this and the deep result assures you that as k tends to infinity that is as you go on doing this step again and again and again and till by you reach a convergence situation where the matrix A_k approaches a quasi upper triangular form. Not really upper triangular, but quasi upper triangular form and this happens for any kind of matrix. Now, what is this quasi upper triangular form? It is this form.

here. As long as we are operate with real arithmetic this kind of 2 by 2 block will remain or non symmetric matrices if there are complex pair of eigenvalues.

In particular for symmetric matrices a quasi upper triangular form actually will mean a quasi diagonal form because for a symmetric matrix we have seen that symmetry is preserved. So, all these 0s will also mean that here also we have got corresponding 0s. So, we will get actually quasi diagonal form all right. Symmetric matrix and such things are anyway not possible for symmetric matrices. So, this is the form to which the qi iterations will make the process converge and finally, after we have got this kind of a form eigenvalues are mostly available with us except for this kind of situations where we may need to conduct a little further post processing.

Now, I have told you what happens, but how this happens this I have not told you. In my teaching typically in mathematics teaching particularly I try to establish most of the results, but this particular result I will not try to establish in the class rather I will tell you what actually happens behind because this is very time consuming and it is important that when you got ready for appreciating the actual process that is happening behind, you should actually conduct the analysis on your own possibly guided with the book. So, if you are ready for going deep into it then I will suggest that you go through the section in the book and try to work out the whole thing on your own then you will get better (Refer Time: 30:33).

In a nutshell what actually happens in the background in QR method in QR decomposition method is that it operates on the basis of relative magnitudes or different eigenvalues, almost the way the power method is used to work. But here the crude operation of power method is actually conducted in a much more sophisticated manner and at a global level, at every level the same refinement of such cases based on the magnitude for their eigenvalues and the segregation of eigenspaces of small eigenvalues from the large once take place at all levels.

So, as iterations proceed this separation take place over the entire spectrum of eigenvalues and that is how the eigenvalues gets sorted, eigenvectors get arranged in suitable subspaces. The way power method works on a single vector here the QR decomposition method at the same time operates over a multitude of eigenspaces and inside each eigenspace it was for finer adjustments finer sorting and so on, all these

processes take place together. And this is the reason why the diagonal blocks it cannot break down completely if quit a few eigenvalues are of almost equal magnitude that is the only limitation.

(Refer Slide Time: 32:08)

Conceptual Basis of QR Method*

QR decomposition algorithm operates on the basis of the *relative magnitudes* of eigenvalues and segregates subspaces.

With $k \rightarrow \infty$,

$$\mathbf{A}^k \text{Range}\{\mathbf{e}_1\} = \text{Range}\{\mathbf{q}_1\} \rightarrow \text{Range}\{\mathbf{v}_1\}$$

and $(\mathbf{a}_1)_k \rightarrow \mathbf{Q}_k^T \mathbf{A} \mathbf{q}_1 = \lambda_1 \mathbf{Q}_k^T \mathbf{q}_1 = \lambda_1 \mathbf{e}_1$.

Further,

$$\mathbf{A}^k \text{Range}\{\mathbf{e}_1, \mathbf{e}_2\} = \text{Range}\{\mathbf{q}_1, \mathbf{q}_2\} \rightarrow \text{Range}\{\mathbf{v}_1, \mathbf{v}_2\}.$$

and $(\mathbf{a}_2)_k \rightarrow \mathbf{Q}_k^T \mathbf{A} \mathbf{q}_2 = \begin{bmatrix} (\lambda_1 - \lambda_2) \sigma_1 \\ \lambda_2 \\ \mathbf{0} \end{bmatrix}$.

And, so on ...

So, if you follow through these, this conceptual basis then you will find that towards the end you get subspaces which are sort of separated from each other based on their eigenvalue magnitudes.

(Refer Slide Time: 32:23)

QR Algorithm with Shift*

For $\lambda_1 < \lambda_2$, entry a_{ij} decays through iterations as $\left(\frac{\lambda_1}{\lambda_2}\right)^k$.

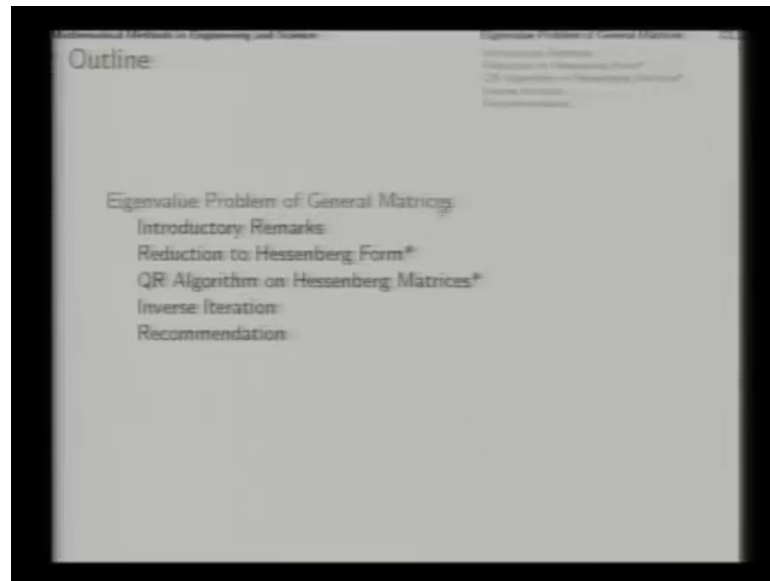
There is another important thing in relation to it is that the rate of convergence depends rate segregation of different eigenspaces is actually directly related to the ratio of eigenvalues λ_i by λ_j and this observation you can make if you go through one actual process of QR decomposition. For that what you can do is that you can take a square matrix and write a small procedure to asset the QR decomposition perhaps based on the method that we did discussed earlier and then actually apply this algorithm and at the end of every iteration you see the result intermediate results, then you will see how the subspaces are getting separated.

Based on this observation of convergence rate there are also some sophisticated algorithms we try to shift the matrix, shifting all its eigenvalues leftward and rightward in order to artificially decreases λ_i minus λ_j which will mean that if this is very small then for successive iterations the errors will drop at very fast rate for such an algorithm is called QR algorithm with shift. It is a little complicated and that is why we will not be discussing it here, but that is what is professionally implemented in most of the professional software library routines.

But we will not go into this, what we will do rather is that just like this brief introduction of the QR decomposition method or general matrices it general and for symmetric tridiagonal matrices particularly we will consider the last topic also the fourth method of effective suitable similarity transformation. Now one point that needs a special mention is that QR decomposition algorithm will operate on any matrix and try to give this kind of a quasi upper triangular form. It operates on hall matrices it is effective, but is it efficient, when you ask that question we need to notice that implemented properly the QR decomposition algorithm the iterative algorithm is a linear time algorithm for a symmetric tridiagonal matrix, a quadratic term algorithm for an upper Hessenberg algorithm and cubic algorithm for the general matrix. This is why we say that if you have got a symmetric matrix then it is very important that you first use it through tridiagonal form and then apply QR decomposition, if you want to apply even though on the full matrix also QR decomposition method would work.

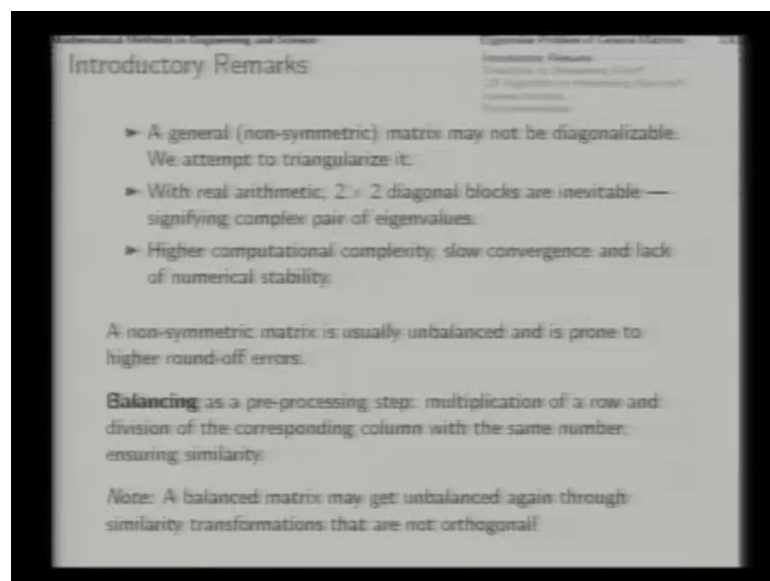
Similarly, in the case of a non symmetric matrix it is better to reduce it first to upper Hessenberg form and then apply QR decomposition. This is the issue which will be considering next.

(Refer Slide Time: 35:50)



The eigenvalue problem of general matrices that is non symmetric matrices is relatively much more complicated and therefore, we find that the theory is much more involved and computational algorithms are a limited and b relatively unstable compared to the orthogonal transformation based method for symmetric matrices. Therefore, in this course we will not go deep in to those theories, rather on these star marked sections will very briefly gross over and try to discuss the basic points here and the important post processing issues here.

(Refer Slide Time: 36:41)



First point regarding non symmetric matrix is that it may not be diagonalizable, a fact that we have discussed earlier. A symmetric matrix is always diagonalizable, but not so for a non symmetric matrix. So, therefore, rather than trying to diagonalize the given matrix if it is non symmetric what we try here is first to triangularize it and get then upper triangular part.

Next point is that with real arithmetic we cannot avoid 2 by 2 diagonal blocks; that means, even if the matrix is actually triangularizable and that also with unitary transformations if we stick to real arithmetic and do not go to complex arithmetic then even triangularization process will not be really complete because 2 by 2 diagonal block will remain which will signify complex pair of eigenvalues. So whenever complex pair of eigenvalues are there such blocks are inhabitable.

Next is that the computational complexity of the algorithm is higher, convergence is low and sometimes the numerical process may turn out to be unstable. These are the typical difficulties with non symmetric matrices.

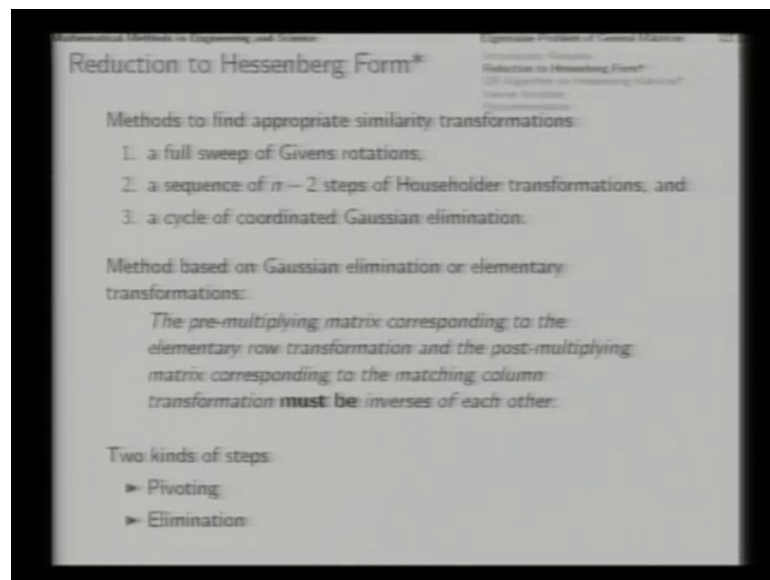
And what we say is a non symmetric matrix is usually unbalanced. You see if the norm of i th row and k th column of a matrix is same for every k then you call that matrix as balanced. A symmetric matrix begin with is balance and through orthogonal transformations it remains balanced a non symmetric matrix is usually unbalanced and therefore, it is going to numerical round of errors more than the symmetric matrix algorithm therefore, in the case of non symmetric matrices as a preprocessing step we sometimes use balancing we try to balance the given matrix a little. And one way to balance it is that if a row is of much higher norm than the corresponding column then we can try to adjust that by multiplication of the row with some number and division of the corresponding column with the second number so that we can balance their norms little bit.

Now, only multiplication to a row is not correct, is not the right step to do because that will be a transformation which will not be a similarity transformation. If we multiply a particular row with half and the corresponding column with 2 then that will be a similarity transformation because the multiplication of the k -th row with half is equivalent to a premultiplication with a with an elementary matrix the corresponding inverse as post multiplication matrix involves the multiplication of the corresponding

column with 2 and so on. So, balancing is a preprocessing step for non symmetric matrix.

Now, through the similarity transformations which are orthogonal a balance matrix does not go on balance, but if the similarity transformation that we apply if they are not orthogonal then even an originally balance matrix may go unbalance again. Now, what we do with the matrix in order to triangularize it? So, there are 3 possible alternative for the first step for a non symmetric matrix.

(Refer Slide Time: 40:38)



The first step for non symmetric matrix is to reduce it to upper Hessenberg form which you can do with a fixed number of arithmetic operations without any iterative process and then further we will conduct iterations on that upper Hessenberg matrix. So, the reduction to Hessenberg can be done in several different ways one is a full sweep of givens rotations as we have discussed earlier, another is a sequence of n minus 2 steps of householder transformations which also we have discussed in the previous lecture. Now, these two things, one of the two each of them converts a symmetric matrix into symmetric tridiagonal form the same operations in the case of a non symmetric matrix will convert it to the upper Hessenberg form. The zeros under the subdiagonal will be arrived at, but that will not mean that over a super diagonal also you will get 0 that will not get in general.

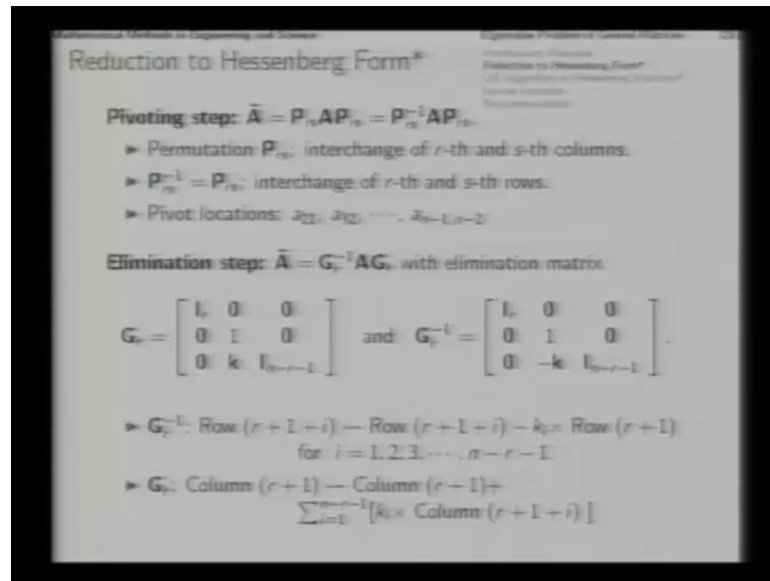
Apart from these two for a non symmetric matrix you can also apply a third method which is a cycle of coordinated Gaussian elimination. To make the record state this kind of a thing is possible for a symmetric matrix also, but in the case of a symmetric matrix this we will typically not do because in the symmetric matrix case the orthogonal transformation based algorithms are superior. Here there is no symmetry to preserve and therefore, it really does not harm to take the third option. In the third option what we do we apply Gaussian elimination in a well coordinated (Refer Time: 42:30) manner.

Note that the straight forward Gaussian elimination which we were applying earlier for solving a linear system of equations that was only based on rows, only row operations we were doing that same thing we cannot do as it is here because every step every transformation of the matrix has to be made through a similarity transformation. That means, whatever matrix we pre multiply with we need to post multiply with the corresponding inverse and therefore, the Gaussian elimination here must be coordinated.

So, this Gaussian elimination or elementary transformations which are applied in a coordinated manner are actually applied from 2 sides - one through pre multiplication and the other through a post multiplication. That means, the pre multiplying matrix corresponding to the elementary row transformation and the post multiplying matrix corresponding to the matching column transformation must be inverses of each other that is how the similarity will be maintained.

So, there are two kinds of steps in these elementary transformations as always one is the pivoting and the other is the elimination. So in the pivoting step typically you try to interchange rows and accordingly interchange column also.

(Refer Slide Time: 44:02)



And in the elimination step typical pre multiplying factor will look like this which will mean subtraction of some multiple of the pivotal row from the lower rows that is why minus k here is a vector. And the corresponding column transformation here which will be equivalent to appropriate addition of all the later rows all the right side rows prove that pivot right side columns sorry columns into the pivotal column this is actually decompose the elimination step.

Through such pivotal through such pivoting and elimination steps we will convert it into the Hessenberg form first and then we apply the QR algorithm on the Hessenberg matrix. And in that process as we keep on applying the QR iterations as soon as a sub diagonal 0 appears we split the matrix into 2 parts and continue with the smaller parts and so on and if towards the end a smaller part turns out to be a 1 by 1 part then that itself is a eigenvalue, if the smaller part turns out to be a 2 by 2 sub matrix then from that through a quadratic equation solving we determine 2 eigenvalues and so on. So, like that we keep on decomposing into smaller diagonal blocks and solve the Eigen value problem in terms of only eigenvalues to begin with.

After that is over then we go in to a method which is called inverse iteration.

(Refer Slide Time: 45:36)

Inverse Iteration

Assumption: Matrix \mathbf{A} has a complete set of eigenvectors.

$(\lambda_i)_0$: a good estimate of an eigenvalue λ_i of \mathbf{A} .

Purpose: To find λ_i precisely and also to find \mathbf{v}_i .

Step: Select a random vector \mathbf{y}_0 (with $\|\mathbf{y}_0\| = 1$) and solve:

$$[\mathbf{A} - (\lambda_i)_0 \mathbf{I}]\mathbf{y} = \mathbf{y}_0.$$

Result: \mathbf{y} is a good estimate of \mathbf{v}_i and:

$$(\lambda_i)_1 = (\lambda_i)_0 + \frac{1}{\mathbf{y}_0^T \mathbf{y}}$$

is an improvement in the estimate of the eigenvalue.

How to establish the result and work out an **algorithm**?

This is a very valuable method because at the end of the reduction up to upper triangular form all that you have is only eigenvalues and not eigenvectors and quite often it may also happen that the eigenvalues are also on the crude estimates. In such a situation when you have got an estimate of the eigenvalues not their exact values then to refine the values of the eigenvalues and the corresponding eigenvectors to get the values of eigenvectors inverse iteration comes as a very good method. For that the assumption is that matrix \mathbf{A} has a complete set of eigen vectors and λ_i is 0 is a good estimate of the i th eigenvalue of \mathbf{A} λ_i is 0 is an estimate of λ_i .

In that case what you do is that in order to find the eigenvalue λ_i plus $i\pi$ and to determine the corresponding eigenvector we first make a test of any random vector \mathbf{y}_0 with magnitude 1 and solve the system. As you solve the system define that the solution \mathbf{y} is a very good estimate of the eigenvector \mathbf{v}_i this is very interesting. The reason is that if we are constructing this matrix $\mathbf{A} - \lambda_i \mathbf{I}$, that will mean that all eigenvalues of this matrix gets shifted by λ_i . If λ_i where an exact estimate of λ_i then this would be a singular matrix and in that case as it is you would not be able to solve it, but if it is a good estimate, but estimate (Refer Time: 47:35) not exact value and that will mean that through this shifting the i th λ_i , i th eigen value becomes of very small magnitude and that means, in the solution which is basically \mathbf{y} that is this matrix inverse into \mathbf{y}_0 you get that particular eigenvector over

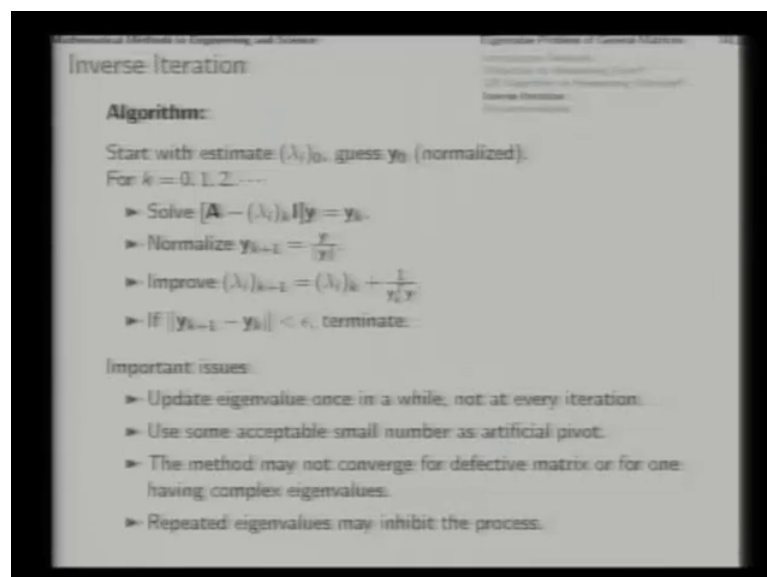
emphasize because the eigenvalue which is extremely small in the inverse the corresponding eigenvalue will be extremely large.

So, that eigenvector gets over emphasized and then therefore, in y you will find that the components of the other eigenvectors become very less in comparison to the eigenvector corresponding to which we have made an estimate here and y is a good estimate of the corresponding eigenvector. And through a little further processing you can see that the difference of the correct value and estimate will be given by this. So, therefore, the expression here gives you an improvement in the estimate of the eigenvalue and then that estimate you can utilize here and continue with this inverse iteration.

Now, this is a very peculiar algorithm in the sense that here you cannot complain of in conditioning in the solution process of this linear system because it is that in conditioning itself that is actually driving this algorithm that is because of that in conditioning your intended eigenvector is actually be getting expressed, is getting emphasized in the solution process and you are able to identify it. Not only that in the later iterations towards the convergence this matrix will become even more and more in conditioning because λ_i estimates will become more and more precise. So, this algorithm is tricky and we have to handle it carefully.

The algorithm is start with an estimated eigenvalue and guess y_0 which is normalized with magnitude 1.

(Refer Slide Time: 49:49)



Inverse Iteration

Algorithm:

Start with estimate $(\lambda_i)_0$, guess y_0 (normalized).

For $k = 0, 1, 2, \dots$

- Solve $[\mathbf{A} - (\lambda_i)_k \mathbf{I}]y = y_k$.
- Normalize $y_{k+1} = \frac{y}{\|y\|}$.
- Improve $(\lambda_i)_{k+1} = (\lambda_i)_k + \frac{1}{y_i^T x}$.
- If $\|y_{k+1} - y_k\| < \epsilon$, terminate.

Important issues:

- Update eigenvalue once in a while, not at every iteration.
- Use some acceptable small number as artificial pivot.
- The method may not converge for defective matrix or for one having complex eigenvalues.
- Repeated eigenvalues may inhibit the process.

And then every iteration you solve this system find a normalize the next y , y_0 in place of y_0 then it will be y_1 y_2 y_3 and so on, next vector to be used here in this system and normalize this and make the improvement over the eigenvalue and then if the previous right side and the new right side are extremely close to each other then terminate that is the convergence. So, this is the algorithm which gives you refinement over the eigenvalues and the values of the corresponding eigenvector. So, this can be done as a close processing step after you have triangularized the matrix. This can work for as symmetric as well as null symmetric matrices.

Now, the important issues here is that every time that you update the eigen value the coefficient matrix actually changes that may mean the solution of the system may turn out to be more and more costly. So, one way to handle this is to update diagonal values only once in a while, not at every iteration.

Next a small artificial number may lead to be used as an artificial pivot while solving this linear system because as we go discussing earlier this matrix is known to (Refer Time: 51:21) condition therefore, when you try to pivot in the linear system solution you will have the difficulty in pivoting because all the entries all the candidates for pivoting will turn out to be extremely small and therefore, a tiny number of your choice you can insert there as an artificial pivot rather than the 0 that might appear otherwise.

The point is that in the beginning we said that this works under the assumption that the matrix has a full set of an eigenvectors. If the matrix is defective or if it has complex eigenvalues in that case it may not converge as usually as you would expect because the underlying theory expects the expression of the chosen vector in the form of a sum of a linear combination of eigenvectors and that may not happen that may not be possible in case of a defective matrix and then in case of complex eigenvalues the pair of complex eigenvalues are both of the same magnitude and therefore, for that eigenvalue the situation may be tricky because the magnitude only one magnitude does not get expressed equally high I mean unequally high compared to others because 2 eigenvalues are of the same magnitude at least. Even repeated eigenvalues or extremely close magnitude eigenvalues we will similarly inhibit the process.

Now, with this much we complete our discussion on the eigenvalue method and at the end we need to have a quick look at the entire spectrum of methods that we have studied

till now in the last five vectors and make a sort of choice on the kind of methods that we typically utilize for solving eigenvalue problems of different kinds of matrices, kinds and sizes.

(Refer Slide Time: 53:13)

Recommendation

Table: Eigenvalue problem: summary of methods:

Type	Size	Reduction	Algorithm	Post-processing
General	Small (up to 4)	Definition; Characteristic polynomial	Polynomial root finding; (eigenvalues)	Solution of linear systems (eigenvectors)
Symmetric	Intermediate (say, 4-12)	Jacobi sweeps	Selective Jacobi rotations	Inverse iteration (eigenvalue refinement and eigenvectors)
	Large	Tridiagonalization (Givens rotation or Householder method)	QR decomposition iteration	
Non-symmetric	Intermediate	Subranging, and then reduction to Hessenberg form	QR decomposition iteration (eigenvalues)	Inverse iteration (eigenvectors)
	Large	(Allow methods on Gaussian elimination)		
General	Very large (extensive requirements)		Power method, shift and deflation	

There are 3 parts, 3 columns here - one is reduction kind of preprocessing then the actual main algorithm and then the post processing. For general matrices is a size is extremely small up to 4 then you do not need sophisticated method, you can reduce it based on definition itself get a characteristic polynomial and algorithm is actually simply polynomial root finding and after that you solve the linear systems to get the eigenvectors.

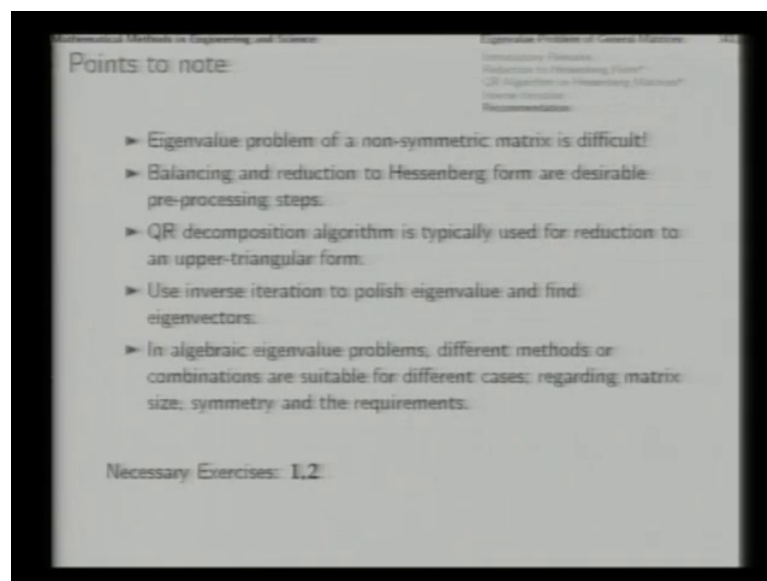
For intermediate size matrices say 4 to 12 you can either use Jacobi Sweeps to reduce the matrix to an extent and then rather than using further sweeps you can use selective Jacobi rotations to diagonalized it completely no post processing required. Alternatively for this range of files isometrics symmetric you can first (Refer Time: 54:36) organized by a given rotational (Refer Time: 54:37) methods and then you can use Sturm sequence property and bracket and bisection, using bracketing and bisection to find the rough eigenvalues. Once sort of eigenvalues are determined then you can use inverse iteration to refine the eigenvalues and determine the eigenvectors.

If you have symmetric matrices which are large say larger than (Refer Time: 55:02) size then reduction must be carried on for the sake of efficiency up to tridiagonalization, here

you use where you have use householder method because that is much more efficient compare to givens rotation called larger matrices. And then the main algorithm will be typically QR decomposition iterations called symmetric tridiagonal matrices. If you are working with non symmetric matrix with intermediate or large size then first you will reduce it through balancing and then through the Hessenberg pump either these methods or through Gauss elimination base methods and then you can use QR decomposition iterations to triangularize the matrix and get the eigenvalues and finally, you can use inverse iteration to refine the eigenvalues and determine the eigenvectors.

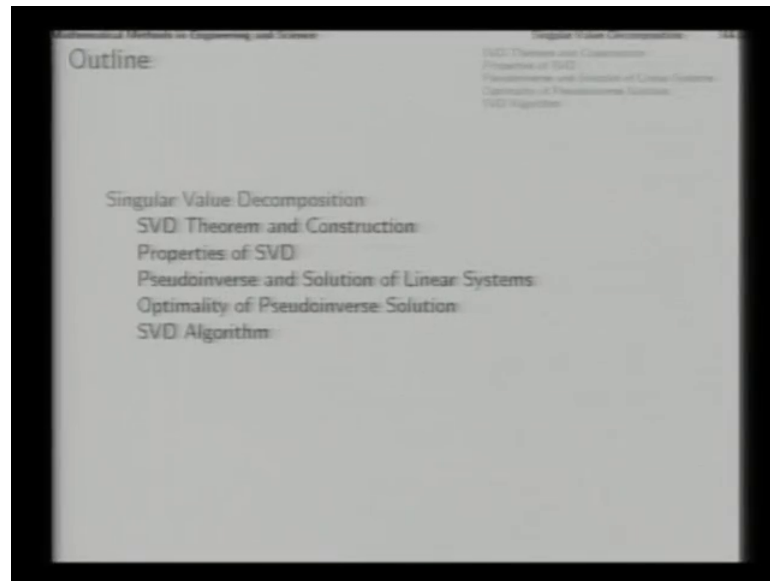
For general matrices symmetric as well as non symmetric if the matrix is very very large, but the requirement this only selective that is if you want to determine only the top and bottom eigenvalues or you want to determine top few eigenvalues in such situations you can use power method shift and deflation to pick up only those pieces of information which you need.

(Refer Slide Time: 56:27)



So, with this we complete this small sub module of the algebraic eigenvalue problem. Earlier we studied the first sub module of linear algebra that was systems of linear equations, now in the next lecture we will combine these two and see the connections between these two problems into the one of the most important lessons of this course which is Singular Value Decomposition that we will discuss in the next lecture.

(Refer Slide Time: 56:56)



Thank you.