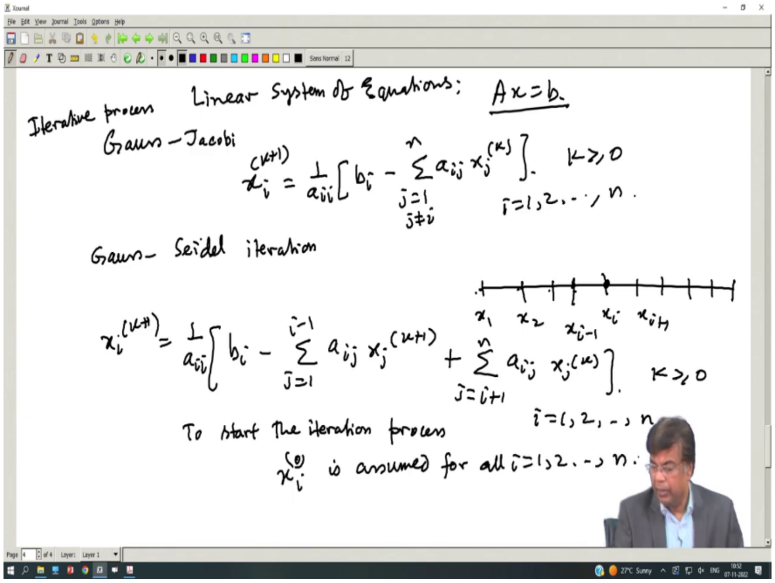
Advanced Computational Techniques
Professor Somnath Bhattacharyya
Department of Mathematics
Indian Institute of Technology, Kharagpur
Lecture 10
Linear System of Equations (Contd.)

(Refer Slide Time: 00:34)


Xournal
File Edit View Journal Tools Options Help
Sans Normal 12
Page 4 of 4 Layer: Layer 1
x_1
x_2
x_{i-1}
x_i
x_{i+1}

Linear System of Equations: $Ax = b.$

Iterative process

Gauss – Jacobi

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{\substack{j=1\\ j\neq i}}^{n} a_{ij}x_j^{(k)}\right]. \quad k \geq 0$$

$i = 1, 2, \ldots, n.$

Gauss – Seidel iteration

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} + \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right]. \quad k \geq 0$$

$i = 1, 2, \ldots, n$

To start the iteration process $x_i^{(0)}$ is assumed for all $i = 1, 2, \ldots, n.$

So, we will talk about some iterative methods we already discussed for solving the linear system of equation Ax= b. If it is a large system. we talked about Gauss-Jacobi iterative process, we are talking about Gauss-Jacobi, there we do is every equation we write as $X_i^{(k+1)} = \frac{1}{a_{ii}}\left\{b_i - \sum_{a_{ij}}^{n} x_j^{(k)}\right\}$ j=1, k≥0, i=1, 2,. . . .n j≠I , this is the Gauss-Jacobi iteration that means the other part, so except the of course, this what they need is a_{ii} to be nonzero. So, if it is a diagonally dominant that guarantees convergence. Then there is an improvement that is what is the Gauss-Seidel iteration, so what we do there is we now when say this is the iteration, this is the points or the grid points or what you call the variables.

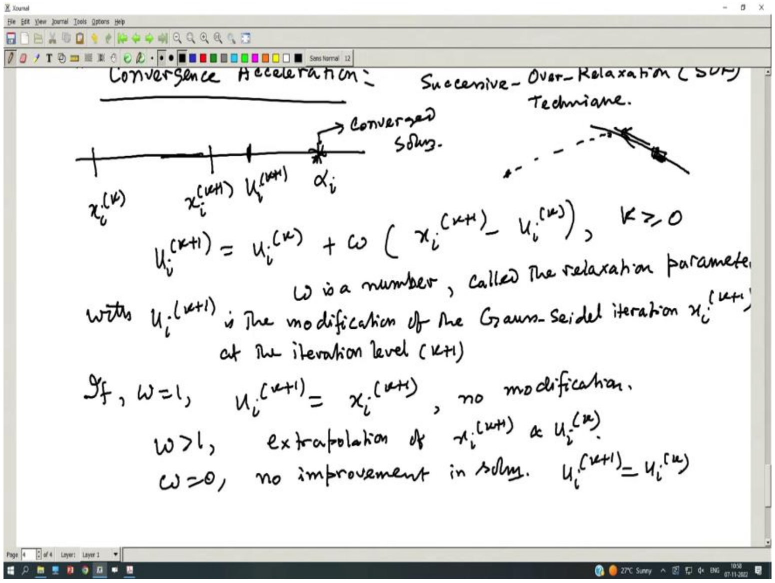
$$x_1 - - - - - x_2 - - - - - x_{i-1} - - - - x_i - - - - - x_{ih} - - - - -$$

this is say x_1, $x_2$etcetera, x_i, x_{i+1}, so I can call this as xi minus 1. So, when we come here x_i so at a particular x_{th} iteration say. So, that means from k we are starting from 0, so at k plus 1 iteration k, maybe the k equal to 0 is a static one. So, add the k plus 1 iteration when we come to x_i,. So, that means, already we have obtained the solution from $x_1, x_2, \ldots, x_{i-1}$up to that. So, Gauss-Seidel iteration make use of this.

So, that means, what we do is $x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} x_j^{(k+1)} \quad + \sum_{j=i+1}^{n} a_{ij} x_j^{(k)}\right)$

So, k≥0 and i=1,2, …n. So, to start the iteration process $x_i^{(0)}$ is assumed for all i=1,2, …n. So, this is already assumed, so this is the Gauss-Seidel iteration or Gauss-Jacobi iteration that already we have discussed and if it is diagonally dominant matrix, we have a convergence guaranteed but without failure of diagonally dominant also it can convert.

Now, as can be seen that this is a linear process, so that means the iteration converge but converge at a slower rate unlike the new translation technique which is quadratically converge. So, that means, every step the error is reducing in a quadratic fashion. But here the error is going in a linear fashion.


Xournal
File Edit View Journal Tools Options Help
Sans Normal 12
converged soln.
$x_i^{(k)}$
$x_i^{(k+1)}$
$u_i^{(k+1)}$
α_i
Page 4 of 4 Layer: Layer 1
27°C Sunny ENG 10:58 07-11-2022

Convergence Acceleration:- Successive-Over-Relaxation (SOR) Technique.

$u_i^{(k+1)} = u_i^{(k)} + \omega\left(x_i^{(k+1)} - u_i^{(k)}\right), \quad k \geq 0$

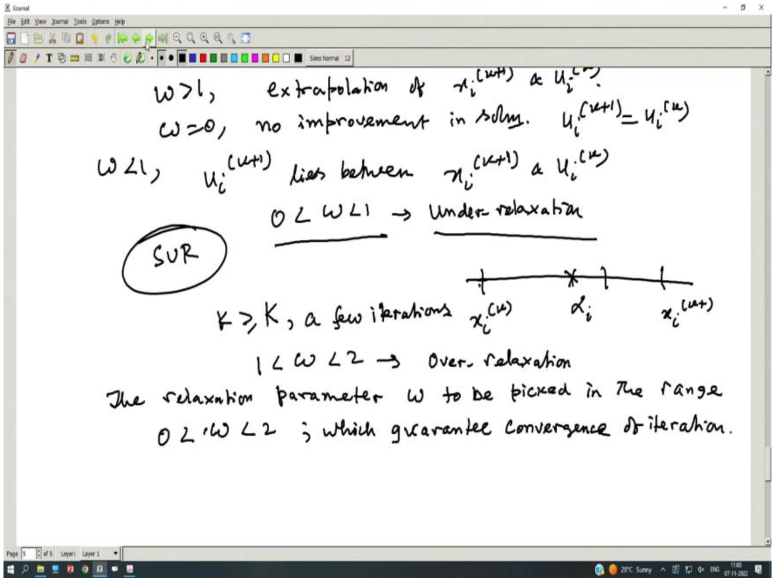
ω is a number, called the relaxation parameter

with $u_i^{(k+1)}$ is the modification of the Gauss-Seidel iteration $x_i^{(k+1)}$ at the iteration level $(k+1)$

If, $\omega = 1$, $u_i^{(k+1)} = x_i^{(k+1)}$, no modification.

$\omega > 1$, extrapolation of $x_i^{(k+1)}$ & $u_i^{(k)}$.

$\omega = 0$, no improvement in soln. $u_i^{(k+1)} = u_i^{(k)}$


Xournal
File Edit View Journal Tools Options Help
Sans Normal 12
Page 5 of 5 Layer: Layer 1

$\omega > 1$, extrapolation of $x_i^{(k+1)}$ & $u_i^{(k)}$.

$\omega = 0$, no improvement in soln. $u_i^{(k+1)} = u_i^{(k)}$

$\omega < 1$, $u_i^{(k+1)}$ lies between $x_i^{(k+1)}$ & $u_i^{(k)}$

$\underline{0 < \omega < 1}$ → Under-relaxation

SUR

$k \geq K$, a few iterations $x_i^{(k)}$ α_i $x_i^{(k+1)}$

$1 < \omega < 2$ → Over-relaxation

The relaxation parameter ω to be picked in the range $0 < \omega < 2$; which guarantee convergence of iteration.

So, in order to accelerate the convergence, convergence acceleration, so there are one very easy technique is called the successive over relaxation technique, SOR in the short form. Now, this is something like that, suppose, you want to shoot a or hit a flying bird. Now, what we do that we assume when we are targeting a position where we want to hit the bird we assume or we position little ahead of the bird where we are thinking about at the time of shooting.

So, what I mean is that suppose you are here and there is a flying bird is moving. So, when you want to shoot this flying bird say something like this bird. So, what you do is you target this position this point and we assume that the time by which the stone or the whatever we are throwing so will hit the object the object will move to this position.

So, this is the technique for SOR. So, that means, what we know that say this is the converge solution alpha for $x_i^{(k)}$--------$x_i^{(k+1)}$ ------ α_i → (converge solution), this is the converge solution. we know that we are moving in the right direction.

So, that is why what we do is we take a extrapolation between these two values and we call that at the k+1 iteration our value is a little ahead of these Gauss-Seidel value, whatever the iteration values, so we call this as $u_i^{(k+1)}$, So, that means $u_i^{(k+1)}$ is written as $u_i^{(k+1)} = u_i^{(k)}$+ω $\left(x_i^{(k+1)} - u_i^{(k)}\right)$, k≥0.

And ω is a real some number called the relaxation parameter. So, one thing is that if ω=1 so $u_i^{(k+1)}$ this modification we have not defined what is $u_i^{(k+1)}$is. So, before that let us call this with $u_i^{(k+1)}$ is the modification of the Gauss-Seidel iteration $u_i^{(k+1)}$ at the iteration level (k +1).

Now, at the iteration level (k +1). So, obviously if ω=1, so that means $u_i^{(k+1)}$ no modification equal to $u_i^{(k+1)}$no modification and what we find that if ω>1, so it is an extrapolation of $u_i^{(k+1)}$ and previous iteration $u_i^{(k)}$can be the $x_i^{(k)}$ or $x_i^{(k+1)}$and another thing is that if ω=0, so no improvement. So, normally omega cannot be zero so no improvement in solution.

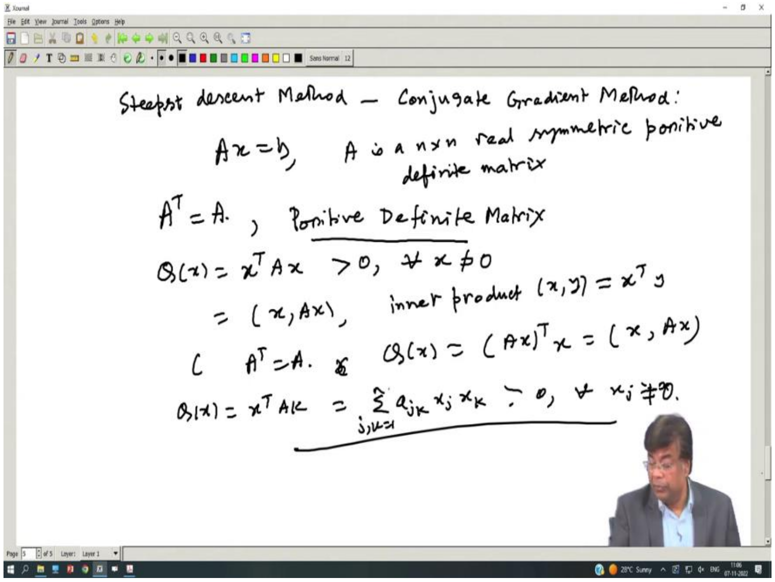
So, that means $u_i^{(k+1)} = u_i^{(k)}$and another important thing is that if ω<0, in that case $u_i^{(k+1)}$ is lying between $u_i^{(k+1)}$and $u_i^{(k)}$. So, in this case this is called the under relaxation. So, $0 < ω < 1$ is called the under relaxation, so where some cases what we find that the say under relaxation SUR is also called SUR.

Now, $u_i^{(k)}$--------α_i $---------x_i^{(k+1)}, \alpha_i$is the converse solution and what we find that this is a $x_i^{(k)}$ or $u_i^{(k)}$and this is $x_i^{(k+1)}$ So, next step $x_i^{(k+2)}$. So, that means there is a either one time it is overshooting another time is under shooting like that way. So, that means it is exceeding the converged value. So, what we find that error is changing the sign. We do not know we may not have the idea of converged solution.

So, what we find that error is or some case is bigger some case is smaller values so if that is the case then we apply that under relaxation technique. So, in between values $x_i^{(k+1)}$So, normally what we do is this either successive over relaxation or under relaxation we applied when after few steps. So, that means for certain k >K finite number of iteration after few iteration level few iterations we apply this technique and then at that point we know that whether the error is successively reducing or it is oscillating. So, that means, either it is going up going positive going negative, so in that case we which kind of technique we will use. Now, whether $0 < ω < 1$ or $1 < ω < 2$ that called the over relaxation. Now, one thing is that it can be proved easily that the relaxation parameter omega must rely on ω to be picked in the range $0 < ω < 2$ if one which guarantees the convergence.

But failure of that does not imply that it is not convergent, convergence of iteration. So, this is how the improvement of convergence is meant. Now, so there it is normally Gauss-Seidel iteration so $x_i^{(k+1)}$ we use by Gauss-Seidel iteration and then we come to the stage, the stage $x_i^{(k+1)}$ by some Gauss-Seidel iteration apply here, get a modified value $u_i^{(k+1)}$ using the previous iterated values and that is called the SUR technique. ω is some value is prescribed and then we repeat the process. So, this is how the convergence acceleration can meet.

(Refer Slide Time: 15:13)



Steepest descent Method — Conjugate Gradient Method:

$Ax = b$, A is a n×n real symmetric positive definite matrix

$A^T = A.$, Positive Definite Matrix

$Q(x) = x^T A x > 0, \ \forall x \neq 0$

$= (x, Ax)$, inner product $(x, y) = x^T y$

($A^T = A.$ & $Q(x) = (Ax)^T x = (x, Ax)$

$Q(x) = x^T A x = \sum_{j,k=1}^{n} a_{jk} x_j x_k > 0, \ \forall x_j \neq 0.$

Now, there are several other iterative process for solving like we are not going to talk about much descent method, class of method and then conjugate gradient method, so we are not discuss on much about this only thing I just want to give a overview for this so any book if you find I mean interested you can learn this conjugate gradient method a iterative process a class of iterative process which is refers the steepest descent method should there if you have a system $Ax = b$ where A is a n×n real symmetric positive definite matrix.

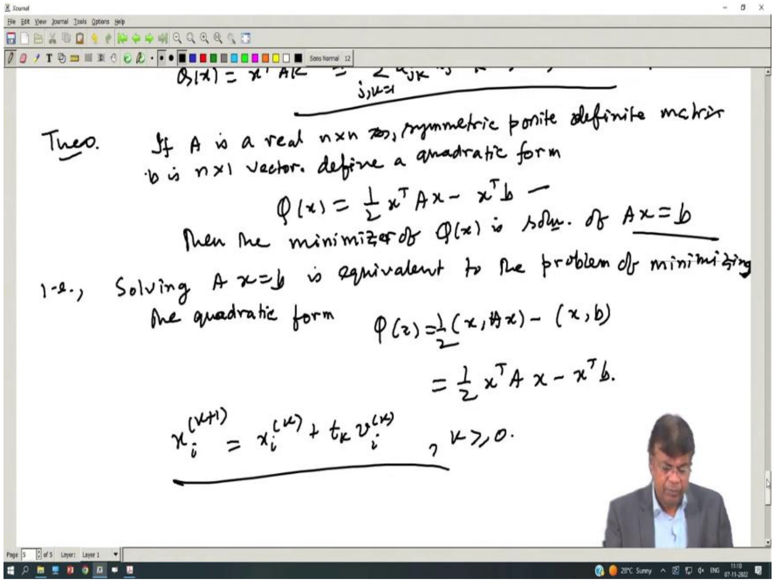
Now, what is symmetric means $A^T = A$ because it is a real. And what is positive definite means, positive definite. Now, if I call a quadratic form so any quadratic form say $= Q(x) = x^T Ax$. So, if this $Q(x) > 0$ for all choice of x ≠0 then it is a positive definite matrix. So, a matrix is such that the quadratic form associated with this is positive always then it is the one which is referred as the positive definite.

Now, this also can be written as x transpose Ax, so if I define (x, Ax) if I define the inner product between two vector (x y) = x^T y. So, this is the inner product. Now, we have also show that this can be written as since $A^T = A$, so this we can write as this Q(x)=$(Ax)^T x =$ (x, Ax) inner product.

So, if such a matrix are there, which are positive definite, so that means

Q(x)=x^TAx = $\sum a_{jk}\ x_j x_k$ so if this kind of form if I write then it is a positive definite. So, this quadratic form is always ≥0 a material of whatever the value is. So, for all x, oh this is positive for all x_j≠0, Now, j,k 1 to n.

(Refer Slide Time: 19:06)



$Q(x) = x^T Ax = \sum_{j,k=1} a_{jk} x_j x_k$

Theo. If A is a real n×n ~~ro~~, symmetric posite definite matrix b is n×1 vector. define a quadratic form

$$\phi(x) = \frac{1}{2} x^T Ax - x^T b$$

Then the minimizer of $\phi(x)$ is soln. of $\underline{Ax = b}$

i.e., Solving $Ax = b$ is equivalent to the problem of minimizing the quadratic form $\phi(x) = \frac{1}{2}(x, Ax) - (x, b)$

$$= \frac{1}{2} x^T A x - x^T b.$$

$$\underline{x_i^{(k+1)} = x_i^{(k)} + t_k v_i^{(k)}} \ , k \geq 0.$$

There is a theorem. Based on this theorem all these steepest descent methods conjugate gradient methods are all derived is if A is a real n×n symmetric positive definite matrix and b is a n×1 vector.

Define a quadratic form as

$$\emptyset(x) = \frac{1}{2} x^T \text{Ax-} x^T b$$

then the minimizer of $\emptyset(x)$ is the solution of Ax = b.

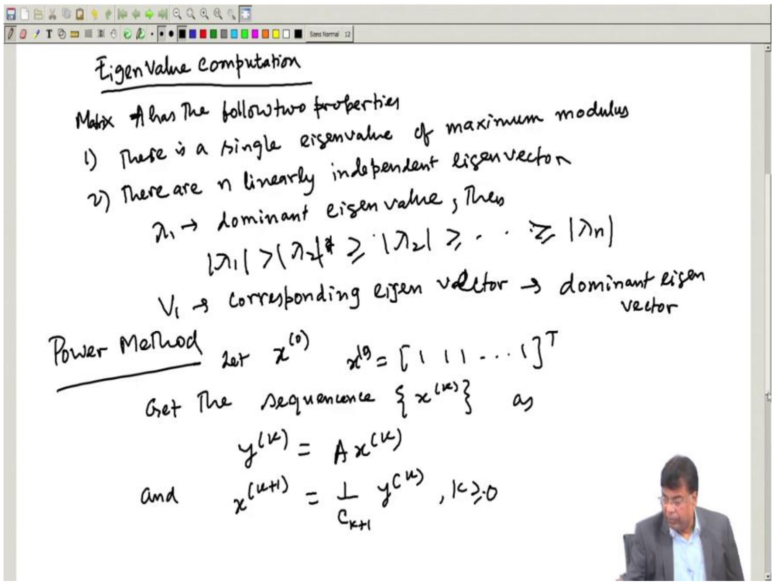
In other words, this quadratic form has minimum value if Ax = b is satisfied that is those x which are the solution of the linear system. In other words, solving that is solving a system Ax = b is equivalent to the problem of minimizing the quadratic form $\emptyset(x)$= $\frac{1}{2}$ (x, Ax)-(x,b)= $\frac{1}{2}x^T$ Ax $x^T b$. So, anyway, I am not going into details of this.

So, based on that, now we have to find out iteratively $x_i^{(k+1)}$= $x_i^{(k)}$+t_k $v_i^{(k)}$

$v_i^{(k)}$ is the direction vector and t_k is step length which needs to be obtained so that this kind of iterative process, iteratively we use this to get a solution for this kind of system for which the quadratic form is positive definite matrix real and symmetric.

So, a class of method can be generated by this. So, I am not discussing details on that. So, any book can be followed to which you will talk in the tutorial sit so this quadratic form can be, I mean based on this quadratic form, we can derive those conjugate gradient method or others.

(Refer Slide Time: 22:55)



Eigen Value computation

Matrix ~~A~~ A has The follow two properties

1) There is a single eigenvalue of maximum modulus

2) There are n linearly independent eigenvector

$\lambda_1 \rightarrow$ dominant eigen value, Then

$|\lambda_1| > \cancel{|\lambda_2|} \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$

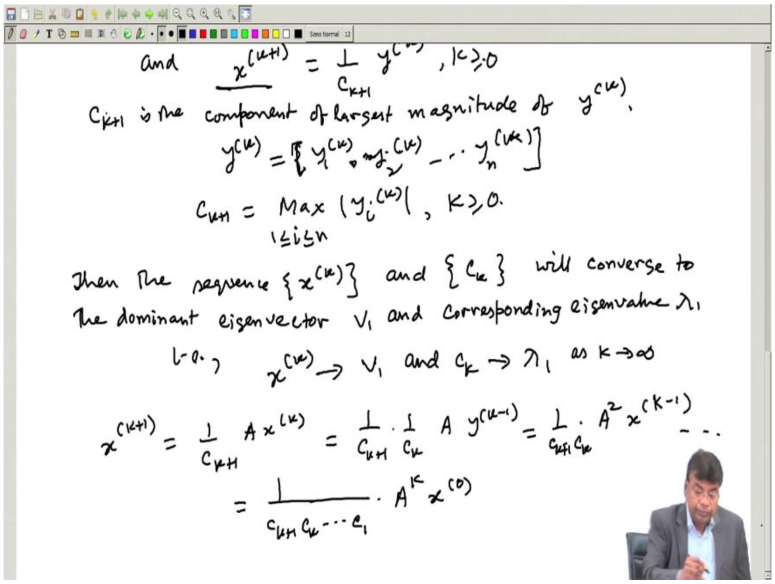
$V_1 \rightarrow$ corresponding eigen vector $\rightarrow$ dominant eigen vector

Power Method Let $x^{(0)}$ $x^{(0)} = [1\ 1\ 1 \cdots 1]^T$

Get The sequence $\{x^{(k)}\}$ as

$y^{(k)} = Ax^{(k)}$

and $x^{(k+1)} = \frac{1}{c_{k+1}} y^{(k)}, k \geq 0$



and $\underline{x^{(k+1)}} = \frac{1}{C_{k+1}} y^{(k)}, k \geq 0$

C_{k+1} is the component of largest magnitude of $y^{(k)}$.

$$y^{(k)} = [y_1^{(k)}, y_2^{(k)} \ldots y_n^{(k)}]$$

$$C_{k+1} = \max_{1 \leq i \leq n} |y_i^{(k)}|, \quad K \geq 0.$$

Then the sequence $\{x^{(k)}\}$ and $\{C_k\}$ will converse to the dominant eigenvector v_1 and corresponding eigenvalue λ_1

i.e., $x^{(k)} \to v_1$ and $C_k \to \lambda_1$ as $k \to \infty$

$$x^{(k+1)} = \frac{1}{C_{k+1}} A x^{(k)} = \frac{1}{C_{k+1}} \cdot \frac{1}{C_k} A y^{(k-1)} = \frac{1}{C_{k+1} C_k} \cdot A^2 x^{(k-1)} \ldots$$

$$= \frac{1}{C_{k+1} C_k \cdots C_1} \cdot A^k x^{(0)}$$

So, we will talk about a algorithm for computing the Eigen value or Eigen vector for a special type of matrix A, say a matrix A if A has a dominating Eigen vector and eigen value. So, let A has the following two properties or the matrix A as the following two properties. One is it has the largest eigen value. There is a single Eigen value of maximum modulus. And, two, there are n linearly independent Eigen vectors.

Now, λ_1 → dominant eigen value. So, then

$|\lambda_1| > |\lambda_2| \geq |\lambda_2| \geq |\lambda_3| \geq \ldots.. \geq |\lambda_n|$ etc. So, this largest Eigen value exists uniquely and $v_1 \rightarrow$ corresponding Eigen vector is called the dominant Eigen vector. So, dominant Eigen value and dominant eigen vector. This is the method called the power method.

So, it is basically a repeated, start with a trial Eigen vector say and then we repeatedly multiply this with, at every iteration we multiply with a given matrix A and then what will show that converges to the Eigen vector and the maximum modulus of that Eigen vector that approaches to the corresponding dominant Eigen value.

So, let $x^{(0)}$ be any vector and say let $x^{(0)}$ we consider as $x^{(0)}$ = $(1\ 1\ 1\ 1)^T$ a starting vector and then get the sequence $\{x^{(k)}\}$ sequence $y^{(k)} =$ A $x^{(k)}$ and then what we do $x^{(k+1)} = \frac{1}{C_{k+1}}\ y^{(k)}$

$y^{(k)} \geq 0$ where $C_{k+1}$is the component of largest magnitude of $y^{(k)}$.

So, that means, $y^{(k)}= \left\{y_1^{(k)}, y_2^{(k)} \ldots\ldots y_n^{(k)} \right\}$.

Then

$C_{k+1} = Max\ |y_i^{(k)}|$ is basically the maximum of $y_i^{(k)}$ $1 \leq i \leq n$, $k \geq 0$. So, the maximum modulus or magnitude is 1 and all are less than 1. So, then what we can say that, then the sequence $x^{(k)}$ and C_k will converge to the dominant Eigen vector.

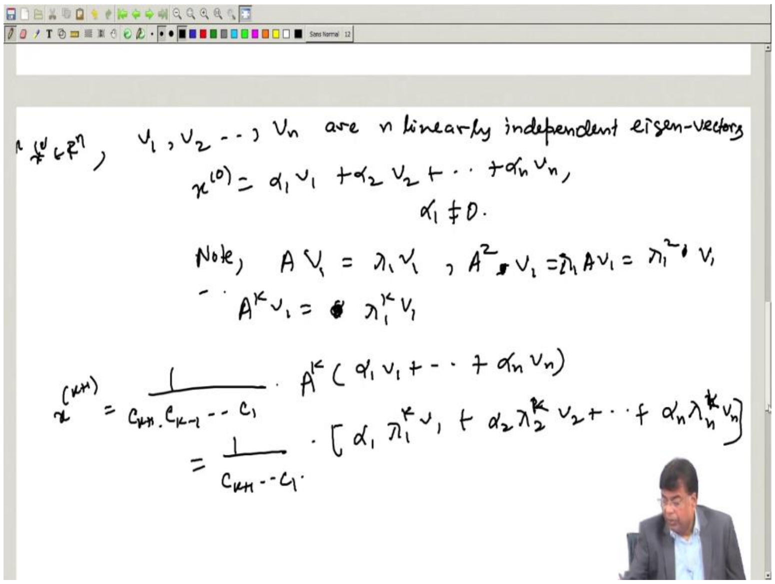
So, let us call it v_1 and Eigen value and corresponding Eigen value, so that means that is the dominant Eigen value, corresponding Eigen value λ_1 that is what we said is $x^{(k)} \rightarrow v_1$ and $C_k \rightarrow \lambda_1$ as $k \rightarrow \infty$. So, this is the algorithm which is referred to as power method. So, every iteration we are multiplying we start with a initial guess x_0, component as 1.

and then every iteration we are multiplying with the matrix A and getting the $x^{(k+1)}$ and rather $y^{(k)}$ then we are normalizing or scaling this is the scale this $x^{(k+1)}$ is the scale of $y^{(k)}$ which is having dividing by the maximum modulus component of maximum modulus C_{k+1}. So, at the when iteration becomes tending to infinity. So, at a large iteration step what we find that these $x^{(k)}$ tends to the corresponding dominant Eigen vector and C_k tends to the Eigen values.

So, now, how this happening, what is the, how to prove that I am not giving a details proof. So, I just giving an outline. So, $x^{(k+1)} = \frac{1}{c_{k+1}}$ A $x^{(k)}= \frac{1}{c_{k+1}} . \frac{1}{c_k}$ A $y^{(k-1)}= \frac{1}{c_{k+1}\, ck} A^2\ x^{(k-1)}. . . .= \frac{1}{c_{k+1}\, ck \ldots c_1} A^k\ x^{(0)}$

Now $x^{(0)}$ is any starting vectors.

(Refer Slide Time: 31:31)



$x^{(0)} \in \mathbb{R}^n$, $v_1, v_2, \ldots, v_n$ are n linearly independent eigen-vectors

$$x^{(0)} = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n,$$

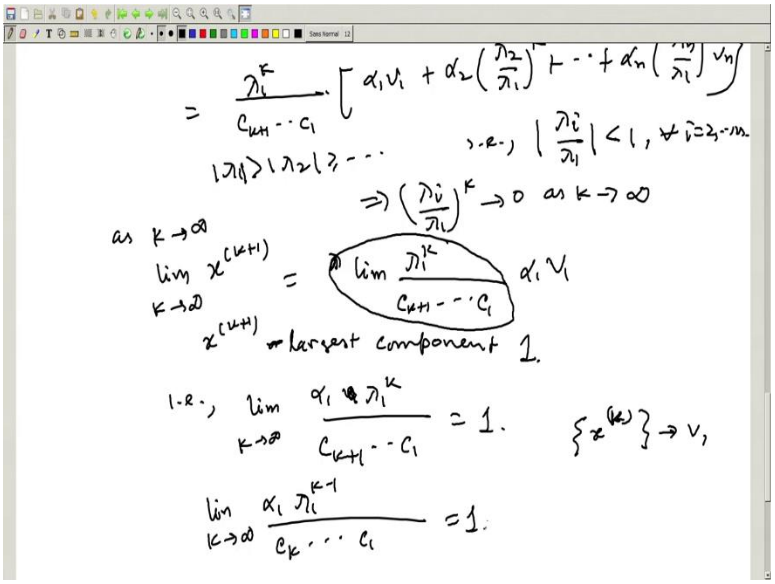
$$\alpha_1 \neq 0.$$

Note, $A v_1 = \lambda_1 v_1$, $A^2 v_1 = \lambda_1 A v_1 = \lambda_1^2 v_1$

$\ldots$ $A^k v_1 = \lambda_1^k v_1$

$$x^{(k+1)} = \frac{1}{c_{k+1} . c_{k-1} \cdots c_1} \cdot A^k (\alpha_1 v_1 + \cdots + \alpha_n v_n)$$

$$= \frac{1}{c_{k+1} \cdots c_1} \cdot \left[\alpha_1 \lambda_1^k v_1 + \alpha_2 \lambda_2^k v_2 + \cdots + \alpha_n \lambda_n^k v_n \right]$$



$$= \frac{\lambda_1^k}{c_{k+1} \cdots c_1} \left[\alpha_1 v_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right) + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right) v_n \right]$$

$|\lambda_1| > |\lambda_2| \geq \cdots$ i.e., $\left| \frac{\lambda_i}{\lambda_1} \right| < 1, \forall i = 2, \ldots, n.$

$$\Rightarrow \left(\frac{\lambda_i}{\lambda_1}\right)^k \to 0 \text{ as } k \to \infty$$

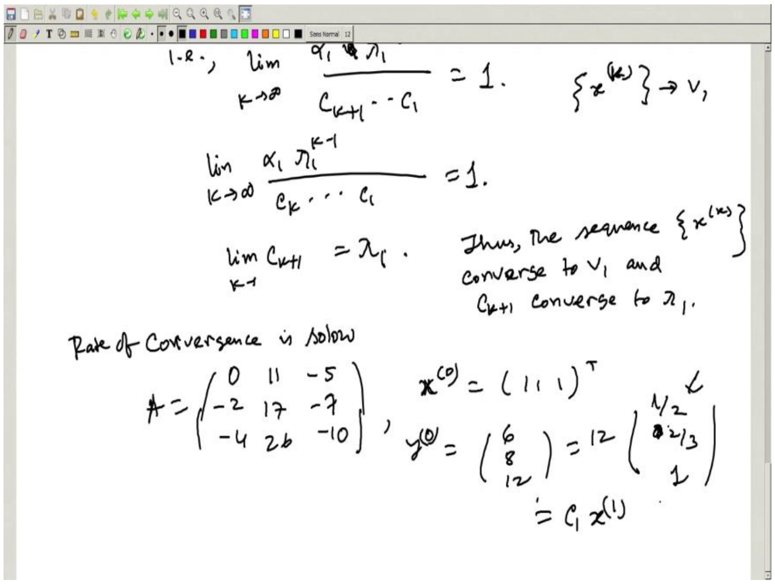
as $k \to \infty$

$$\lim_{k \to \infty} x^{(k+1)} = \lim \frac{\lambda_1^k}{c_{k+1} \cdots c_1} \alpha_1 v_1$$

$x^{(k+1)}$ largest component 1.

i.e., $\lim\limits_{k \to \infty} \frac{\alpha_1 \lambda_1^k}{c_{k+1} \cdots c_1} = 1.$ $\{x^{(k)}\} \to v_1$

$$\lim_{k \to \infty} \frac{\alpha_1 \lambda_1^{k-1}}{c_k \cdots c_1} = 1.$$



i.e., $\lim_{k\to\infty} \frac{\alpha_1 \lambda_1}{c_{k+1}\cdots c_1} = 1.$ $\{x^{(k)}\} \to v_1$

$\lim_{k\to\infty} \frac{\alpha_1 \lambda_1^{k-1}}{c_k\cdots c_1} = 1.$

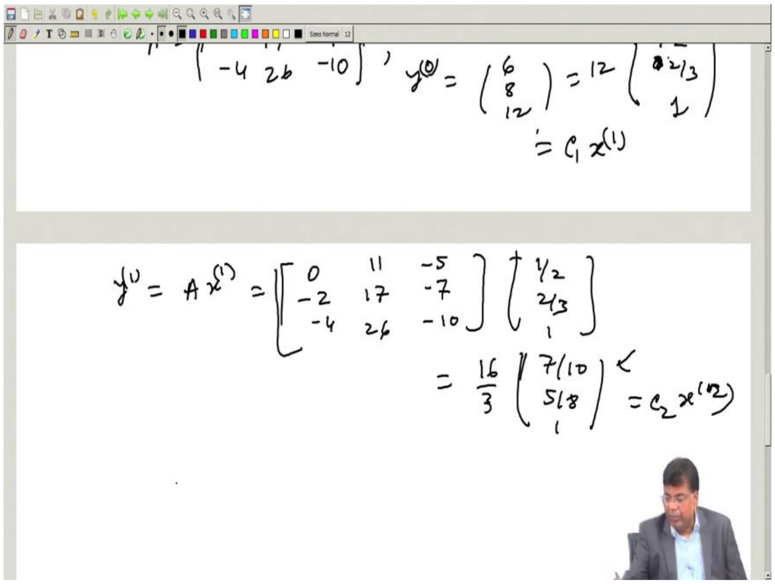
$\lim_{k\to 1} c_{k+1} = \lambda_1.$ Thus, the sequence $\{x^{(k)}\}$ converse to v_1 and c_{k+1} converse to λ_1.

Rate of Conversence is solow

$A = \begin{pmatrix} 0 & 11 & -5 \\ -2 & 17 & -7 \\ -4 & 26 & -10 \end{pmatrix}$, $x^{(0)} = (1\ 1\ 1)^T$

$y^{(0)} = \begin{pmatrix} 6 \\ 8 \\ 12 \end{pmatrix} = 12 \begin{pmatrix} 1/2 \\ 2/3 \\ 1 \end{pmatrix}$

$= c_1 x^{(1)}$



$\begin{pmatrix} -4 & 26 & -10 \end{pmatrix}$, $y^{(0)} = \begin{pmatrix} 6 \\ 8 \\ 12 \end{pmatrix} = 12 \begin{pmatrix} 2/3 \\ 1 \end{pmatrix}$

$= c_1 x^{(1)}$

$y^{(1)} = A x^{(1)} = \begin{bmatrix} 0 & 11 & -5 \\ -2 & 17 & -7 \\ -4 & 26 & -10 \end{bmatrix} \begin{bmatrix} 1/2 \\ 2/3 \\ 1 \end{bmatrix}$

$= \frac{16}{3} \begin{pmatrix} 7/10 \\ 5/8 \\ 1 \end{pmatrix} = c_2 x^{(2)}$

Now, what we can say this $x^{(0)}$ say $v_1, v_2 \ldots. v_n$ are n linearly independent Eigen vectors. This is already existed. So, this $x^{(0)} = \alpha_1 v_1 + \alpha_2 v_2 + \ldots . \alpha_n v_n, \alpha_1 \neq 0.$ So, any vector in R^n, $x^{(0)}$ is any vector in R^n, so that can be because these are linearly independent. So, we can write as a linear combination of that.

So, now, see this note A $v_1 = \lambda_1 v_1$, $A^2 v_1 = \lambda_1 A v_1 = {\lambda_1}^2 v_1$ and so in that way I can write that $A^k v_1 = {\lambda_1}^k v_1$. So, now, if I apply this property. So, now, what we have is what we have is $A^k x^{(0)}$. So, this $x^{(0)}$now, we substitute from here and apply this property.

So, what we get is $x^{(k+1)}$= $\frac{1}{c_{k+1.ck-1 \,....c_1}} A^k(\lambda_1 \, v_1 \, . \, . \, . \, . \, \lambda_n \, v_n)$ = $\frac{1}{c_{k+1.c_1}}(\alpha_1 \, \lambda_1{}^k v_1 \, + \, \alpha_2 \, \lambda_2{}^k v_2 +. \, . \, . \, \alpha_n \, \lambda_n{}^k v_n \,)$

$= \frac{\lambda_n{}^k}{c_{k+1..c_1}} \, (\alpha_1 \, v_1 \, + \alpha_2 \, (\frac{\lambda_2}{\lambda_1})^k$+……$\alpha_n \, (\frac{\lambda_n}{\lambda_1})^k \, v_n \,)$, k → ∞.

Now, what we know is $|\, \lambda_1|$>$|\, \lambda_2|\, |$> . . . .i.e., $|\frac{\lambda_i}{\lambda_1}| < 1$, i=2….n, So that means, if I have then, so this implies that =$(\frac{\lambda_i}{\lambda_1})^k$ → 0 as k → ∞.

So, that means, as k → ∞ I take the limit k → ∞ so $x^{(k+1)}$= Lim $\frac{\lambda_1{}^k}{c_{k+1..c_1}} \, \lambda_1 \, v_1$

all are tending to 0. Now, $x^{(k+1)}$ has the largest component as 1, but that is the way we have constructed.

So, what I can say from here that is lim $\frac{\alpha_1 \, \lambda_1{}^k}{c_{k+1..c_1}} = \, 1$ So, this is one thing. And now, what we can see that. So, in that case we call this limit as the $x^{(k+1)}$ So, if I take, so that means this sequence $x^{(k)}$ is converging to a sequence to the limit v_1 and if I the same way so, this is k → ∞.

So, same way if I take $\frac{\alpha_1 \, \lambda_1{}^{k-1}}{c_{k....c_1}} = 1$

So, from there if I take the ratio so what I find that lim k → ∞, C_{k+1}=λ_1

This is by definition $\{x^{(k)}\}$ converse to v_1 and the maximum modulus and c_k and corresponding c_{k+1}converts to λ_1. So, this is the thing.

So, this can be illustrated by an example say rate of convergence is slow, this is the one of the drawbacks. So, if you have A=$\begin{pmatrix} 0 & 11 & 5 \\ -2 & 17 & -7 \\ -4 & 26 & -10 \end{pmatrix}$

So, if I start with $x^0 \, = \, (111)^T$, what do you have to first find out this y^0=$\begin{pmatrix} 6 \\ 8 \\ 12 \end{pmatrix}$=12 $\begin{pmatrix} 1/2 \\ 2/3 \\ 1 \end{pmatrix}$ = $c_1 x_1$

which comes out to be 6 8 12. So, now if you scale, so that means the maximum modulus so half 3 by, 2 by 3 1. So, this is becoming the x1, so this is equal to c1 x1.

So, once I have that, so, then $c_1x_1$is there. So, then we got y_1= A x^1 = $\begin{pmatrix} 0 & 11 & 5 \\ -2 & 17 & -7 \\ -4 & 26 & -10 \end{pmatrix}\begin{pmatrix} 1/2 \\ 2/3 \\ 1 \end{pmatrix}$ = $\frac{16}{3}\begin{pmatrix} 7/10 \\ 5/8 \\ 1 \end{pmatrix}$= $C_2x^{(2)}$ so on, like that way the sequence are constructed. This is how the power method goes to compute the eigen value. It is simple, but its rate of convergence is slow. Thank you.