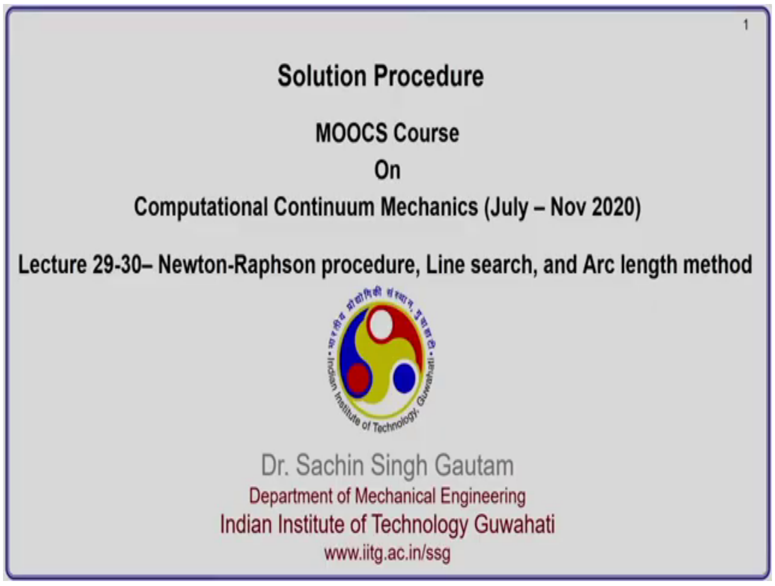
Computational Continuum Mechanics

Dr. Sachin Singh Gautam
Department of Mechanical Engineering
Indian Institute of Technology, Guwahati

Solution Procedure
Lecture - 29-30
Newton-Raphson procedure, Line search and Arc length method

(Refer Slide Time: 00:33)



Solution Procedure

MOOCS Course

On

Computational Continuum Mechanics (July – Nov 2020)

Lecture 29-30– Newton-Raphson procedure, Line search, and Arc length method

Dr. Sachin Singh Gautam

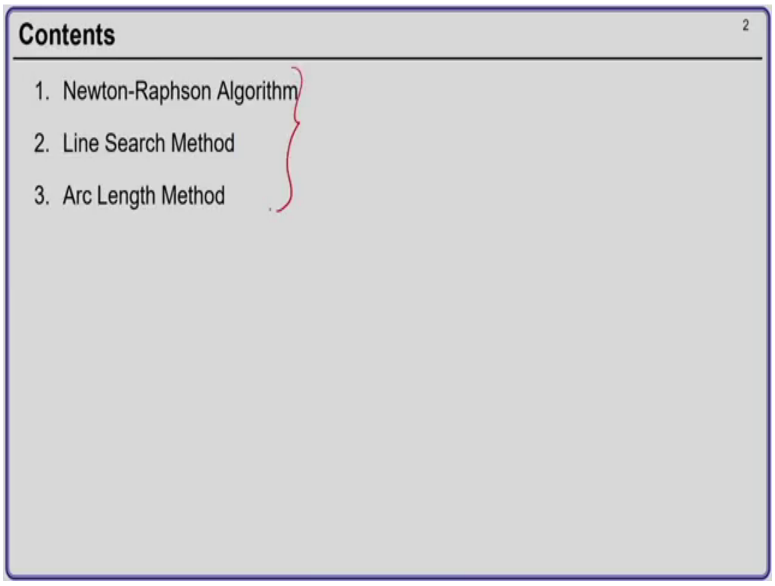
Department of Mechanical Engineering

Indian Institute of Technology Guwahati

www.iitg.ac.in/ssg

So, in this module we will see the different solution procedures, which are used to solve the non-linear system of discretize equation that we have obtained in our previous lectures, ok. In particular we will discuss about the Newton-Raphson solution algorithm and we will also discuss two of the strategies like Line search, and Arc length method, which are used to handle divergence of Newton-Raphson method or for a special situations where the Newton-Raphson method may fail.

(Refer Slide Time: 01:30)



Contents

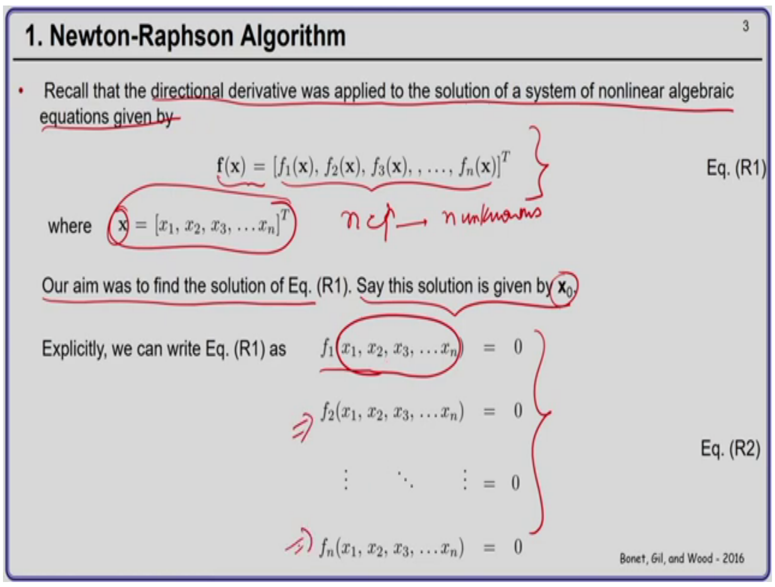
2

1. Newton-Raphson Algorithm
2. Line Search Method
3. Arc Length Method

So, in this module we will be discussing the following three topics. First, we will have a quick revision of the Newton-Raphson algorithm that we discussed in our mathematical essentials that we covered in the initial part of this course followed by the Newton-Raphson algorithm for the discretized equilibrium equations and the discretized linearized virtual work expression that we obtained in the previous lectures, ok.

And finally, we will discuss two methods that is line search method and arc length method which are used when Newton-Raphson method faces convergence issue or it faces, it cannot solve a certain problem ok, because of that peculiar nature in which the load deflection curve behaves, ok.

(Refer Slide Time: 02:26)



1. Newton-Raphson Algorithm

3

- Recall that the directional derivative was applied to the solution of a system of nonlinear algebraic equations given by

$$\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x}), , \ldots, f_n(\mathbf{x})]^T \quad \text{Eq. (R1)}$$

where $\mathbf{x} = [x_1, x_2, x_3, \ldots x_n]^T$ n eq $\rightarrow$ n unknowns

Our aim was to find the solution of Eq. (R1). Say this solution is given by $\mathbf{x}_0$.

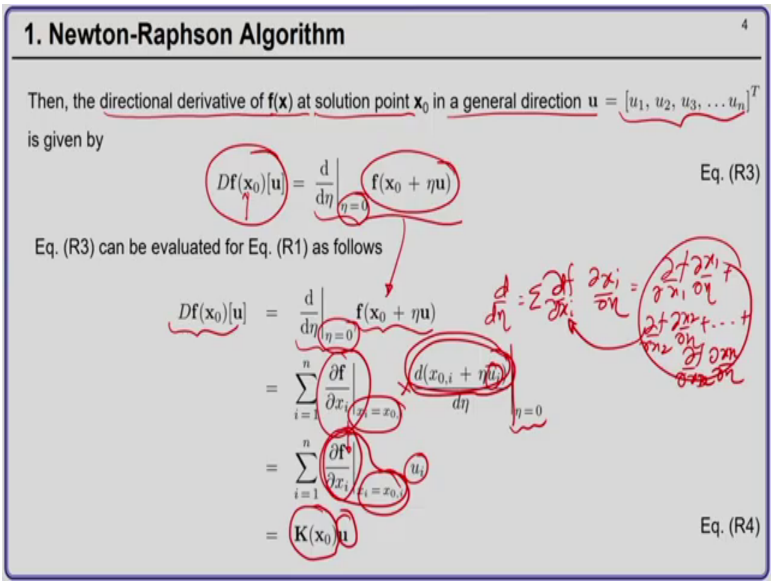
Explicitly, we can write Eq. (R1) as

$$\left.\begin{array}{rcl} f_1(x_1, x_2, x_3, \ldots x_n) & = & 0 \\ f_2(x_1, x_2, x_3, \ldots x_n) & = & 0 \\ \vdots \quad \ddots \quad \vdots & = & 0 \\ f_n(x_1, x_2, x_3, \ldots x_n) & = & 0 \end{array}\right\} \quad \text{Eq. (R2)}$$

Bonet, Gil, and Wood - 2016

So, now recall that, the directional derivative was applied to the solution of system of non-linear algebraic equations ok, as this ok. So, we had this system of non-linear equations which were composed of n such equations. We had n such equations and the unknowns over n unknowns. So, X was the vector of unknowns which was x 1, x 2, x 3 all the way up to x n. So, we had n equations and we have n unknowns. Once, we have this, our aim now is to find the solution of our system of non-linear algebraic equation.

(Refer Slide Time: 03:44)



1. Newton-Raphson Algorithm

Then, the directional derivative of $\mathbf{f}(\mathbf{x})$ at solution point $\mathbf{x}_0$ in a general direction $\mathbf{u} = [u_1, u_2, u_3, \dots u_n]^T$ is given by

$$D\mathbf{f}(\mathbf{x}_0)[\mathbf{u}] = \left.\frac{\mathrm{d}}{\mathrm{d}\eta}\right|_{\eta=0} \mathbf{f}(\mathbf{x}_0 + \eta\mathbf{u}) \qquad \text{Eq. (R3)}$$

Eq. (R3) can be evaluated for Eq. (R1) as follows

$$\begin{aligned} D\mathbf{f}(\mathbf{x}_0)[\mathbf{u}] &= \left.\frac{\mathrm{d}}{\mathrm{d}\eta}\right|_{\eta=0} \mathbf{f}(\mathbf{x}_0 + \eta\mathbf{u}) \\ &= \sum_{i=1}^{n} \left.\frac{\partial \mathbf{f}}{\partial x_i}\right|_{x_i = x_{0,i}} \left.\frac{d(x_{0,i} + \eta u_i)}{d\eta}\right|_{\eta=0} \\ &= \sum_{i=1}^{n} \left.\frac{\partial \mathbf{f}}{\partial x_i}\right|_{x_i = x_{0,i}} u_i \\ &= \mathbf{K}(\mathbf{x}_0)\mathbf{u} \end{aligned} \qquad \text{Eq. (R4)}$$

$$\frac{d}{d\eta} = \sum \frac{\partial f}{\partial x_i}\frac{\partial x_i}{\partial \eta} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial \eta} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial \eta} + \dots + \frac{\partial f}{\partial x_n}\frac{\partial x_n}{\partial \eta}$$

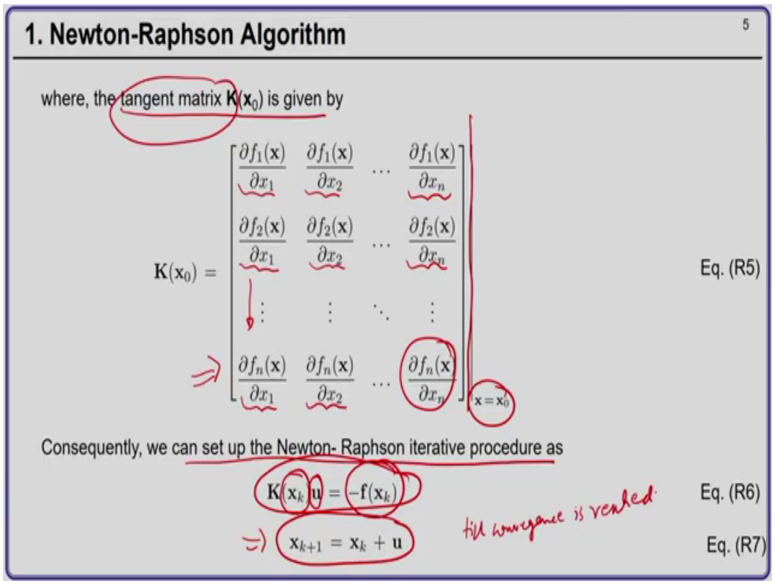
Let us say this solution is X 0, then we can explicitly write equation R 1 as f 1, function of all the unknowns equal to 0, f 2 equal to 0 all the way up to f n equal to 0. Then, we can take the directional derivative of the system of non-linear algebraic equation at the solution point X 0 in a direction u ok.

So, u is a direction in which we proceed; so, that X 0 plus u will take us closer to the solution of this non-linear set of algebraic equation. So, the change in the value of the function ok, evaluated at X 0 in the direction u is nothing but the, directional derivative of the function evaluated at X 0 plus eta u with respect to eta and then substituting eta equal to 0.

So, now when we solve equation R 3 ok; so, this is del by del eta of f of X 0 plus eta u evaluated at eta equal to 0, and using chain rule I can write d by d eta as d by or del by del xi and del xi by del eta summation, ok. This is like del f by del x 1 del x 1 by del eta plus del f by del x 2 del x 2 by del eta plus all the way up to del f by del x 3 sorry, del f by del x n del x n by del eta. So, this as a summation is here.

So, that is what we have here we have del f by del x i evaluated at the solution point X 0 into del by del eta of X 0 plus eta u, evaluated at eta equal to 0, ok. Once, we differentiate this with respect to eta we get our coefficient of eta here u i and this term over here is nothing, but the gradient of the function f with respect to x evaluated at solution point X 0 ok. So, the gradient of f with respect to X can be written as a matrix K evaluated at X 0 into increment of the general direction u, ok.

(Refer Slide Time: 06:35)



1. Newton-Raphson Algorithm

5

where, the tangent matrix $\mathbf{K}(\mathbf{x}_0)$ is given by

$$\mathbf{K}(\mathbf{x}_0) = \left.\begin{bmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \frac{\partial f_1(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \frac{\partial f_2(\mathbf{x})}{\partial x_1} & \frac{\partial f_2(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f_2(\mathbf{x})}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n(\mathbf{x})}{\partial x_1} & \frac{\partial f_n(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f_n(\mathbf{x})}{\partial x_n} \end{bmatrix}\right|_{\mathbf{x}=\mathbf{x}_0} \qquad \text{Eq. (R5)}$$

Consequently, we can set up the Newton- Raphson iterative procedure as

$$\mathbf{K}(\mathbf{x}_k)\mathbf{u} = -\mathbf{f}(\mathbf{x}_k) \qquad \text{Eq. (R6)}$$

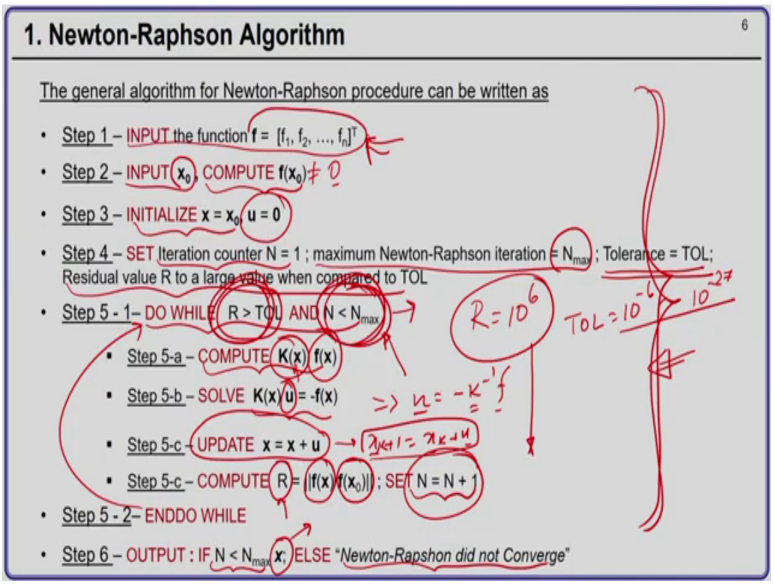
$$\Rightarrow \mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{u} \qquad \text{Eq. (R7)}$$

till convergence is reached

Now, the explicit form of this matrix also called the tangent matrix and evaluated at X 0 is given by del f 1 by del x 1 del f 1 by del x 2 all the way up to del f n del f 1 by del x n. Second row is del f 2 by del x 1 del f 2 by del x 2 all the way up to del f 2 by del x n hence so on, and finally, the last row is del f n by del x 1 del f n by del x 2 all the way up to del f n by del x n and this matrix has to be now evaluated at the solution point X equal to X 0, ok.

And now, we can set up the Newton-Raphson iterative procedure used as following equation, ok. So, the tangent matrix evaluated at a particular Newton-Raphson step K into the direction u equal to minus of f X k and then the variable X is updated as X k plus 1 equal to X k plus u and this we keep till convergence is reached.

(Refer Slide Time: 07:56)



1. Newton-Raphson Algorithm

6

The general algorithm for Newton-Raphson procedure can be written as

- Step 1 – INPUT the function $\mathbf{f} = [f_1, f_2, \ldots, f_n]^T$
- Step 2 – INPUT x_0, COMPUTE $f(x_0) \neq 0$
- Step 3 – INITIALIZE $x = x_0$, $u = 0$
- Step 4 – SET Iteration counter N = 1 ; maximum Newton-Raphson iteration = N_{max} ; Tolerance = TOL; Residual value R to a large value when compared to TOL
- Step 5 - 1– DO WHILE $R > TOL$ AND $N < N_{max}$
    - Step 5-a – COMPUTE $K(x)$ $f(x)$
    - Step 5-b – SOLVE $K(x) u = -f(x)$ $\Rightarrow u = -K^{-1} f$
    - Step 5-c – UPDATE $x = x + u$ $\rightarrow$ $x_{k+1} = x_k + u$
    - Step 5-c – COMPUTE $R = \|f(x)/f(x_0)\|$; SET $N = N + 1$
- Step 5 - 2– ENDDO WHILE
- Step 6 – OUTPUT : IF $N < N_{max}$ x; ELSE "*Newton-Rapshon did not Converge*"

$R = 10^6$

$TOL = 10^{-6}$

10^{-27}

Remember, we discussed this and we also discussed this algorithm for Newton-Raphson method for solving a system of non-linear algebraic equation. So, just to summarize what we had discussed at that point I have reproduced this algorithm here, ok. So, first you have to input the function and then you have to also input your initial guess or the solution point and then you evaluate your function at this solution point.

If you are lucky, your solution point will be your solution, but it may not be that case. So, what will happen f of X 0 will not be equal to 0 vector or then what to do then we initialize X equal to X 0 and we initialize u equal to 0, and then we set the iteration counter the Newton-Raphson iteration counter n equal to 0 n equal to 1. We also set the maximum number of Newton-Raphson iteration. We set the tolerance for the convergence. And, the residual R is set to, there is residual value R which is set to a very large value when compare to tolerance, ok.

So, I said that time R maybe set to 10 raise to power 6 or something like this which is a very high value, and tolerance we can set to a low value like 10 raise to power minus 6 and sometime it goes all the way up to 10 raise to power minus 27 like this ok. So, we can set to a really low value and you can set residual R to a really large value. And, then we run a do loop while our residual is more than the tolerance and our number of Newton-Raphson iterations are less than the maximum number of Newton-Raphson iterations, ok.

So, this is there so that the loop does not go infinitely. Next, what we do we compute our gradient K that is a tangent matrix at the solution point X. So, for N equal to 1, this X will be equal to X 0 and also we calculate the function f at X 0.

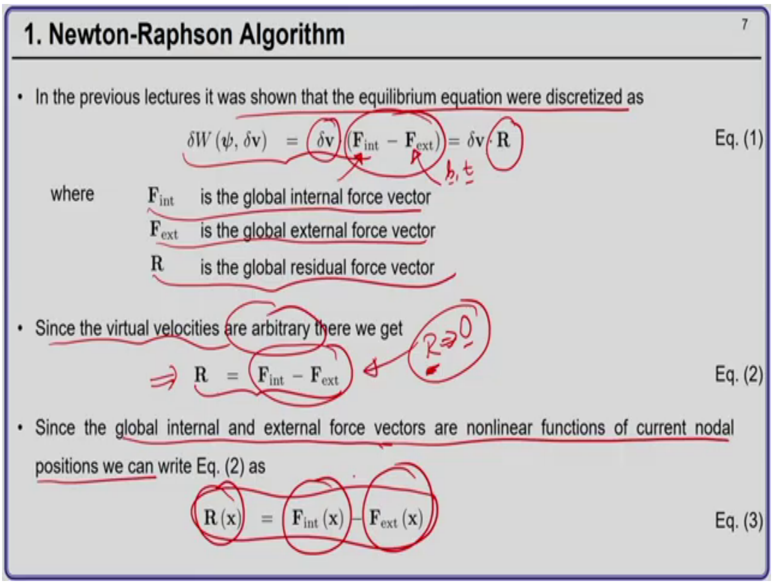
And, then we solve for the general direction u using K u equal to minus f therefore u equal to minus of K inverse f. Once, we have solved we update the value. So, this is like xk plus 1 is xk plus u and then what we do is, we again find out the function at new solution point xk plus 1 and take its norm with respect to the value of the function at the start.

And, we then call this as residual R, and then we increment the total number of Newton-Raphson iterations by 1 and then we go back and then we check whether our R that we just now computed is more than tolerance or less than tolerance. If it is more than the tolerance value then we repeat the Newton Raphson steps or our N is less than N max ok, then we repeat the Newton-Raphson step, ok.

If either of these condition is not valid then we come out of the loop, ok. So, once say we have achieved the convergence within the prescribed number of Newton Raphson iteration steps that is if N is less than N max our solution is the value X else, if N is more than N max we say that our Newton Raphson iteration did not converge, ok. Now, the point to see here is that we are interested in this algorithm in finding the root ok, X 0 is the final solution X is the root of the system of non-linear algebraic equation.

However, in our previous lectures what we have derived, we had external loads and these external loads maybe applied in certain steps. So, this Newton Raphson algorithm that we have discussed has to be now changed a little bit the whole idea remains same, but there will be some additional steps that may come in between.

(Refer Slide Time: 12:44)



1. Newton-Raphson Algorithm

7

- In the previous lectures it was shown that the equilibrium equation were discretized as

$$\delta W(\psi, \delta \mathbf{v}) = \delta \mathbf{v} \cdot (\mathbf{F}_{\text{int}} - \mathbf{F}_{\text{ext}}) = \delta \mathbf{v} \cdot \mathbf{R} \qquad \text{Eq. (1)}$$

where $\mathbf{F}_{\text{int}}$ is the global internal force vector
$\mathbf{F}_{\text{ext}}$ is the global external force vector
$\mathbf{R}$ is the global residual force vector

- Since the virtual velocities are arbitrary there we get

$$\Rightarrow \mathbf{R} = \mathbf{F}_{\text{int}} - \mathbf{F}_{\text{ext}} \qquad \text{Eq. (2)}$$

- Since the global internal and external force vectors are nonlinear functions of current nodal positions we can write Eq. (2) as

$$\mathbf{R}(\mathbf{x}) = \mathbf{F}_{\text{int}}(\mathbf{x}) - \mathbf{F}_{\text{ext}}(\mathbf{x}) \qquad \text{Eq. (3)}$$

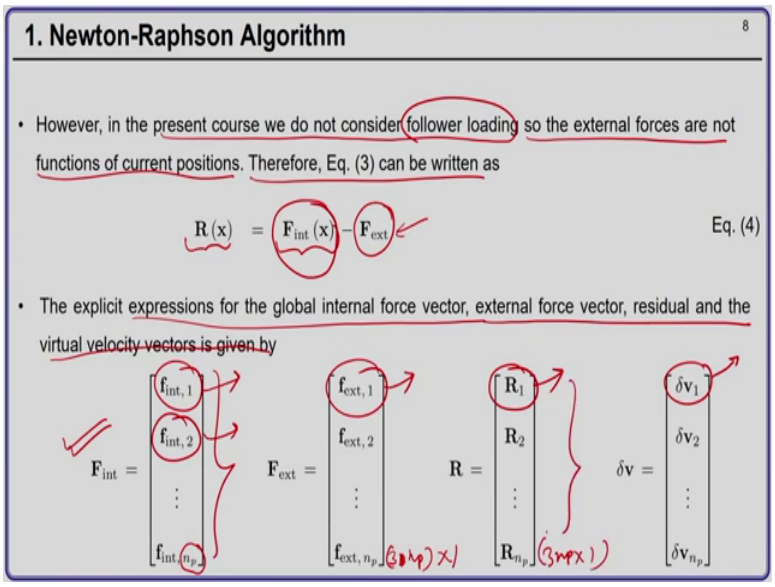
So, in the previous lectures we had shown that the equilibrium equation can be descretized as del v which is the virtual velocity dot with the difference of the internal and the external forces, ok. So, the internal forces come because of the stresses generated inside the body and the external forces come because of the body forces and surface applied surface tractions, ok. And, then the difference of the internal external forces if the equilibrium is not achieved will be equal to some residual which is we denote by R, ok

So, F internal is the global internal force vector, F external is the global external force vector and R is the global residual vector ok. Now, because the virtual velocities in equation 1, they are arbitrary therefore, the residual has to be equal to the difference of the internal and the external forces, ok.

Now, as you approach equilibrium this residual ok. So, this residual vector will go to 0 vector, ok. So, in general it becomes very small, and now, our objective of the Newton-Raphson method is to get this residual go to 0. Now, these global internal and external force vectors which we have here, in equation 2 are non-linear functions of current nodal position, ok.

Therefore, equation 2 can be explicitly written as the residual R which is a function of current nodal position equal to the internal forces, global internal forces which are function of current nodal position minus the external forces which are the function of current nodal positions.

(Refer Slide Time: 14:47)



1. Newton-Raphson Algorithm

- However, in the present course we do not consider follower loading so the external forces are not functions of current positions. Therefore, Eq. (3) can be written as

$$R(x) = F_{int}(x) - F_{ext} \qquad \text{Eq. (4)}$$

- The explicit expressions for the global internal force vector, external force vector, residual and the virtual velocity vectors is given by

$$F_{int} = \begin{bmatrix} f_{int,1} \\ f_{int,2} \\ \vdots \\ f_{int,n_p} \end{bmatrix} \quad F_{ext} = \begin{bmatrix} f_{ext,1} \\ f_{ext,2} \\ \vdots \\ f_{ext,n_p} \end{bmatrix}_{(3n_p)\times 1} \quad R = \begin{bmatrix} R_1 \\ R_2 \\ \vdots \\ R_{n_p} \end{bmatrix}_{(3n_p \times 1)} \quad \delta v = \begin{bmatrix} \delta v_1 \\ \delta v_2 \\ \vdots \\ \delta v_{n_p} \end{bmatrix}$$

However, in the present course we have not considered any follower loads or any deformation dependent loads ok, no pressure loading, no follower loads, ok. So, our external forces are not functions of current positions.

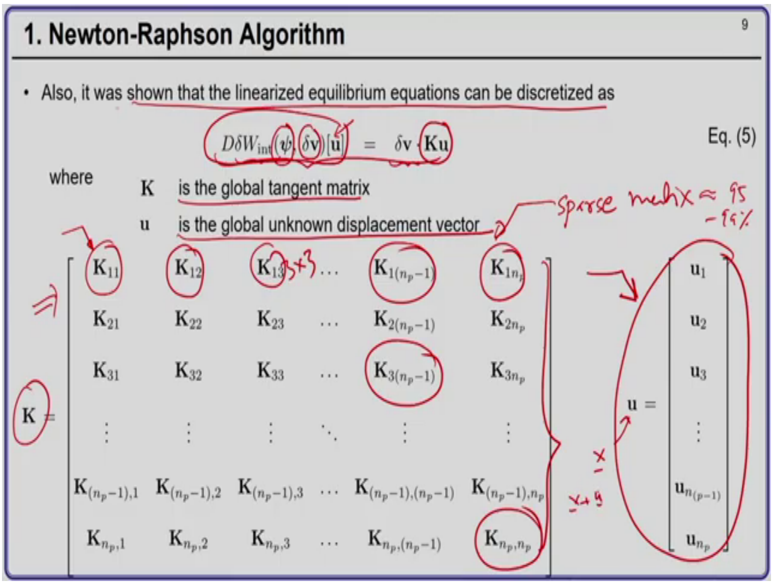
Therefore, equation three can be written as the residual which is a function of current nodal position equal to the internal forces as a function of current nodal position minus the external force and now see external forces are not the function of the current nodal position.

So, nonlinearity comes now it is only because of the internal force vector being non-linearly depending on the current nodal position. Now, the explicit expressions for the global internal force vector, global external force vector, the residual and the virtual velocity vectors is given by following array form, ok.

So, where each of these values correspond to the say the internal forces corresponding to node 1, global node 1, internal forces corresponding to global node 2, this corresponds to the external forces corresponding to global node 1 this is the residual corresponding to global node 1 and del v 1 is the virtual velocity vector corresponding to global node 1, ok.

And, remember we have total of np number of finite element nodes in our mesh ok. So, these matrices if we consider 3 degrees of freedom per node will be 3 cross np ok, 3 np cross 1, ok. So, this is 3 np cross 1 ok. So, if you have 1000 nodes, these will be vectors of 1000 cross 1, ok.

(Refer Slide Time: 16:48)



1. Newton-Raphson Algorithm

9

- Also, it was shown that the linearized equilibrium equations can be discretized as

$$D\delta W_{\text{int}}(\psi, \delta \mathbf{v})[\mathbf{u}] = \delta \mathbf{v} \cdot \mathbf{K}\mathbf{u} \qquad \text{Eq. (5)}$$

where K is the global tangent matrix

u is the global unknown displacement vector

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} & \mathbf{K}_{13} & \cdots & \mathbf{K}_{1(n_p-1)} & \mathbf{K}_{1n_p} \\ \mathbf{K}_{21} & \mathbf{K}_{22} & \mathbf{K}_{23} & \cdots & \mathbf{K}_{2(n_p-1)} & \mathbf{K}_{2n_p} \\ \mathbf{K}_{31} & \mathbf{K}_{32} & \mathbf{K}_{33} & \cdots & \mathbf{K}_{3(n_p-1)} & \mathbf{K}_{3n_p} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{K}_{(n_p-1),1} & \mathbf{K}_{(n_p-1),2} & \mathbf{K}_{(n_p-1),3} & \cdots & \mathbf{K}_{(n_p-1),(n_p-1)} & \mathbf{K}_{(n_p-1),n_p} \\ \mathbf{K}_{n_p,1} & \mathbf{K}_{n_p,2} & \mathbf{K}_{n_p,3} & \cdots & \mathbf{K}_{n_p,(n_p-1)} & \mathbf{K}_{n_p,n_p} \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \\ \vdots \\ \mathbf{u}_{n(p-1)} \\ \mathbf{u}_{n_p} \end{bmatrix}$$

Now, we had also shown that the linearized equilibrium equation can be discretized as follows, ok. So, the discretized form of the linearized virtual work, internal virtual work expression as a function of deformation mapping and virtual velocities in a general direction u is nothing but, the virtual velocity dotted with the tangent matrix times the global unknown vector which in our case is the displacement vector.

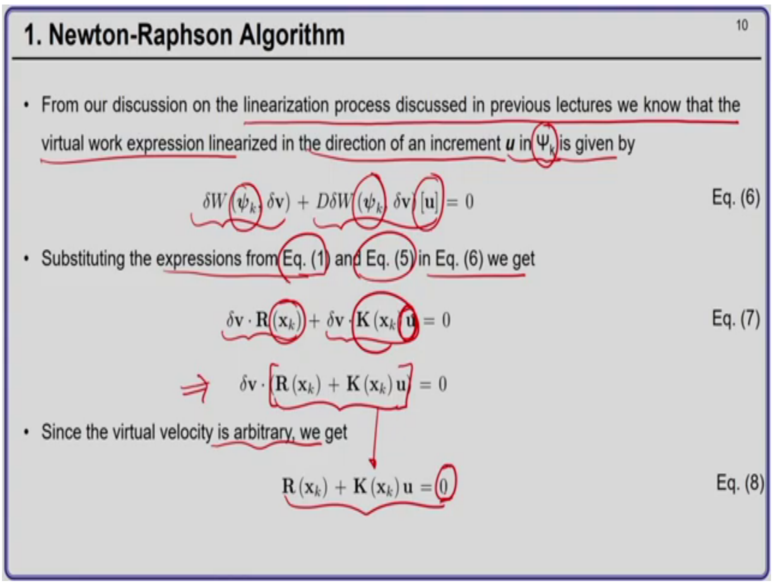
Therefore, K is the global tangent matrix and u is a global unknown displacement vector and the form of this global tangent matrix it is shown in this matrix form, ok. So, where K 11 corresponds to the change. So, stiffness matrix is a change, it represents change. So, K 11 shows the change in the stiffness matrix, change in the global tangent matrix at node 1, because of changes at the node 1 itself, ok.

So, K 12 denotes the changes in the tangent matrix at node 1, because of the changes in the node 2, ok, like this. Similarly, K 3 n p minus 1 will denote for example, the change in the stiffness matrix or the tangent matrix and node 3, because of the change at node np minus 1 like this we can compute the tangent matrix.

So, many of these the property of tangent matrix is that many of these sum matrices, ok. So, all the sum matrices are 3 by 3 and many of these metarises will be equal to 0 matrices, because of the connectivity matrix only few number of nodes are connected to a particular node therefore, all of the nodes if there is some change, they will not affect the particular node.

So, there are lot number of 0's in this tangent matrix and this is also a very, I mean this is a sparse matrix. Sparse matrix means close to 95 to 99 percentage of the entries will be 0, and this is the global displacement vector that we need to find out. This is a change ok. So, this is a change from the current position X that will lead us to in the direction of equilibrium and when we add these two current position X to get a new position X plus u, we hope that we will go closer to the equilibrium, ok.

(Refer Slide Time: 19:43)



1. Newton-Raphson Algorithm

10

- From our discussion on the linearization process discussed in previous lectures we know that the virtual work expression linearized in the direction of an increment ***u*** in Ψ_k is given by

$$\delta W(\psi_k, \delta \mathbf{v}) + D\delta W(\psi_k, \delta \mathbf{v})[\mathbf{u}] = 0 \qquad \text{Eq. (6)}$$

- Substituting the expressions from Eq. (1) and Eq. (5) in Eq. (6) we get

$$\delta \mathbf{v} \cdot \mathbf{R}(\mathbf{x}_k) + \delta \mathbf{v} \cdot \mathbf{K}(\mathbf{x}_k)\mathbf{u} = 0 \qquad \text{Eq. (7)}$$

$$\Rightarrow \quad \delta \mathbf{v} \cdot [\mathbf{R}(\mathbf{x}_k) + \mathbf{K}(\mathbf{x}_k)\mathbf{u}] = 0$$

- Since the virtual velocity is arbitrary, we get

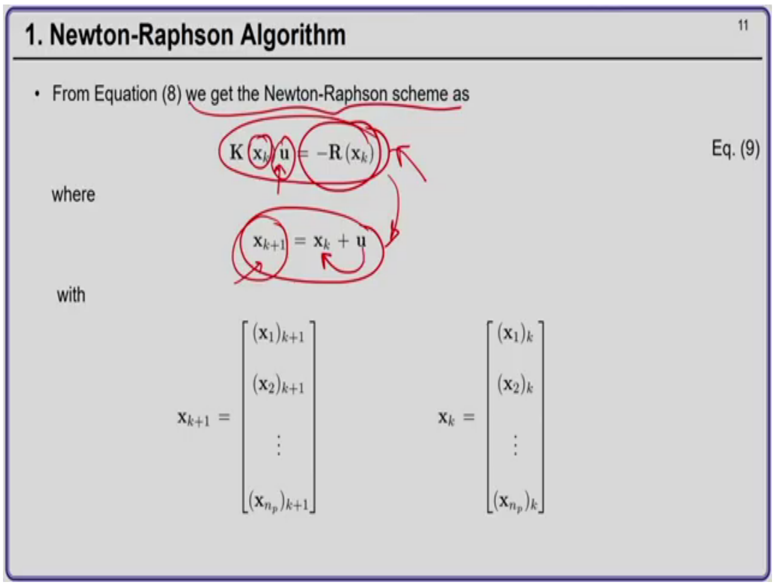
$$\mathbf{R}(\mathbf{x}_k) + \mathbf{K}(\mathbf{x}_k)\mathbf{u} = 0 \qquad \text{Eq. (8)}$$

So, from our discussion on the linearization process discussed in the previous lectures, we know that the virtual work expression linearized in the direction of an increment u in psi k, where k denotes the Newton-Raphson iteration step, and this was given by the internal virtual work evaluated at current Newton-Raphson step k plus the directional derivative of the virtual work evaluated at current Newton-Raphson iteration step psi k and taken in the direction u.

So, once we substitute the expressions from equation 1 and equation 5 ok, in equation 6 we get del v dot R evaluated at current Newton Raphson step k plus del v dot K that is the tangent matrix evaluated at current step k into u and u will is the change that is we have to add to X k to get a new position, so that we can get closer to the equilibrium position, ok.

So, now, taking the virtual velocities outside the bracket, what we get is this term inside the bracket dotted with del v and now, because the virtual velocities are arbitrary therefore, the term inside the bracket has to be equal to the 0 vector. So, this has to be a 0 vector.

(Refer Slide Time: 21:34)



1. Newton-Raphson Algorithm

11

- From Equation (8) we get the Newton-Raphson scheme as

$$K(\mathbf{x}_k)\,\mathbf{u} = -\mathbf{R}(\mathbf{x}_k) \qquad \text{Eq. (9)}$$

where

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{u}$$

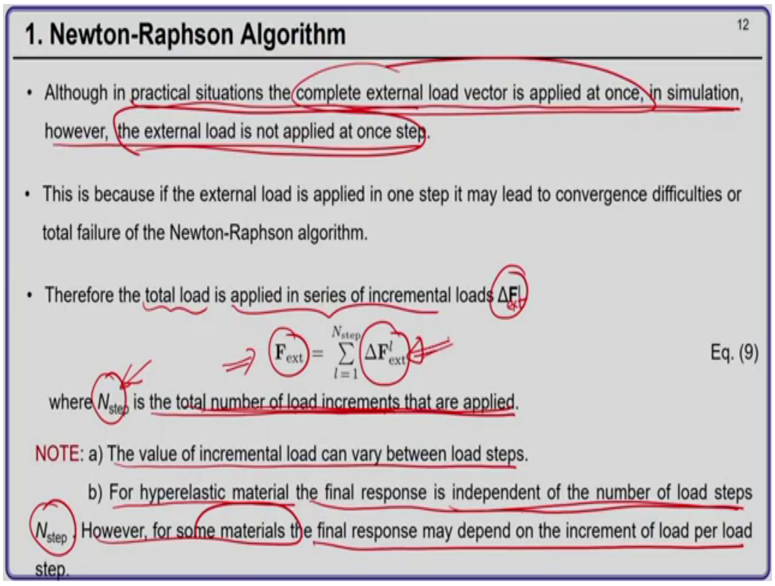
with

$$\mathbf{x}_{k+1} = \begin{bmatrix} (\mathbf{x}_1)_{k+1} \\ (\mathbf{x}_2)_{k+1} \\ \vdots \\ (\mathbf{x}_{n_p})_{k+1} \end{bmatrix} \qquad \mathbf{x}_k = \begin{bmatrix} (\mathbf{x}_1)_k \\ (\mathbf{x}_2)_k \\ \vdots \\ (\mathbf{x}_{n_p})_k \end{bmatrix}$$

And then, we can set up the Newton-Raphson iteration scheme as the tangent matrix evaluated at X k times the change u equal to minus of the residual R evaluated at X k and once we have found out u the new approximation for the equilibrium position can be obtained by adding u to the previous approximation for equilibrium that was X k.

And then, we hope that this X k plus 1 will be the equilibrium position, if it is not then we will again solve equation 9 with X k plus 1, ok. Instead of X k now, we will solve with X k plus 1 and then we will get a new value of u and this will be done till the convergences achieved.

(Refer Slide Time: 22:29)



1. Newton-Raphson Algorithm

12

- Although in practical situations the complete external load vector is applied at once, in simulation, however, the external load is not applied at once step.
- This is because if the external load is applied in one step it may lead to convergence difficulties or total failure of the Newton-Raphson algorithm.
- Therefore the total load is applied in series of incremental loads ΔF

$$\mathbf{F}_{ext} = \sum_{l=1}^{N_{step}} \Delta \mathbf{F}^{l}_{ext} \qquad \text{Eq. (9)}$$

where N_{step} is the total number of load increments that are applied.

NOTE: a) The value of incremental load can vary between load steps.

b) For hyperelastic material the final response is independent of the number of load steps N_{step}. However, for some materials the final response may depend on the increment of load per load step.

So, a important point to now note is that in practical situations the complete external load vector is not applied at once. So, when you see an actual case going on the complete external load will be applied in at once; however, in simulations you cannot apply this total external load at once, because you may face convergence issue or the worst of all you may not even get a solution.

So, what we do is; we apply the total load, ok. So, this total load is applied in a series of increment that is small increments delta F l and the total load would be sum over all the increments of this external force l ok, and this N step is the total number of load increments that are applied.

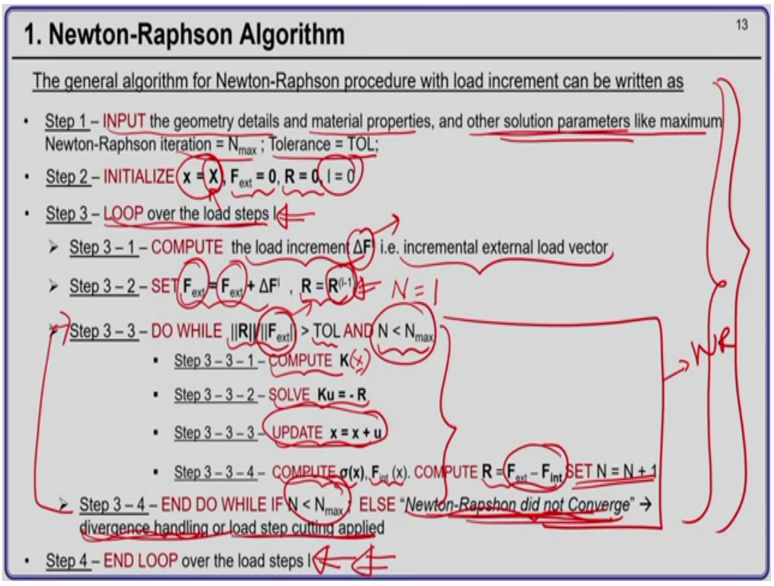
So, the smaller the value of N step probably, you will have more convergence difficulties the more the number of increments, total number of load increments you take, the more will be the faster will be the convergence of Newton-Raphson method, but there is a trade of, because if you are using full Newton-Raphson method that will be discussed next, you will have more computational effort going into each load step, ok.

So, there has to be a balance between the total amount of load increment that you require and the time it takes to compute the tangent matrix in each load step, ok. So, the value of the incremental load can vary between the load step. So, you have to note that this increment in the load step ok, can vary between different load step, it need not be same ok.

So, if you are say if you are finding that your convergence is very good it may increase the load step, if your convergence is becoming bad then you can decrease the load step. So, that is what practically many of the software's do. For hyper-elastic materials that we have in this course, the final response is independent of the number of load steps that you take, because the response for hyper-elastic material only depends on the initial and the final point and it does not depend on the path you take, ok.

So, your response, final response will be independent of the number of load step that you take. However, for some materials the final response may depend on the increment of load, per load step, ok. So, there are certain material for which your response will be different if you take different load steps that you need to remember.

(Refer Slide Time: 25:22)



1. Newton-Raphson Algorithm

13

The general algorithm for Newton-Raphson procedure with load increment can be written as

- Step 1 – INPUT the geometry details and material properties, and other solution parameters like maximum Newton-Raphson iteration = N_{max} ; Tolerance = TOL;
- Step 2 – INITIALIZE $x = X$, $F_{ext} = 0$, $R = 0$, $I = 0$
- Step 3 – LOOP over the load steps I
  - Step 3 – 1 – COMPUTE the load increment ΔF^I i.e. incremental external load vector
  - Step 3 – 2 – SET $F_{ext} = F_{ext} + \Delta F^I$, $R = R^{(I-1)}$
  - Step 3 – 3 – DO WHILE $||R|| / ||F_{ext}|| > TOL$ AND $N < N_{max}$
    - Step 3 – 3 – 1 – COMPUTE $K(x)$
    - Step 3 – 3 – 2 – SOLVE $Ku = -R$
    - Step 3 – 3 – 3 – UPDATE $x = x + u$
    - Step 3 – 3 – 4 – COMPUTE $\sigma(x)$, $F_{int}(x)$. COMPUTE $R = F_{ext} - F_{int}$ SET $N = N + 1$
  - Step 3 – 4 – END DO WHILE IF $N < N_{max}$; ELSE "*Newton-Rapshon did not Converge*" → divergence handling or load step cutting applied
- Step 4 – END LOOP over the load steps I

So, now our Newton-Raphson algorithm for the linearize equilibrium equations that we had is now given here, ok. So, there is a more or less the idea remains same, but we add a loop over the load steps now. So, what you have to do first is you have to input the geometric details that is the geometric data of the body.

You have to input the material properties and you have to input the other solution parameters like maximum number of Newton-Raphson iterations, tolerance, you have to give the nodal, coordinates, initial nodal coordinates, connectivity, number of Gauss points that you need for integration and the boundary condition all these data have to be given.

Once, you have this you initialize your solution to the initial position of the body, the external loads are initialized to be 0, the residual is made 0 and the load step is made to be 0, ok. Now, you loop over load step, because you know how much of load you want to apply and you already would have decided that you want to go for these many load step, then according to your choice, you set the value of total number of load steps and then you loop over load steps, you compute first of all the load increment delta F l.

That is the incremental external load vector you calculate, then you add the incremental external load vector to your total external force vector to get the current value of the external force vector and then you set the residual for the current load step to be equal to the residual from the previous load step. You can do it, you can set it to some other value, but we can set it to the previous value of the residual and then we start our Newton Raphson solution algorithm.

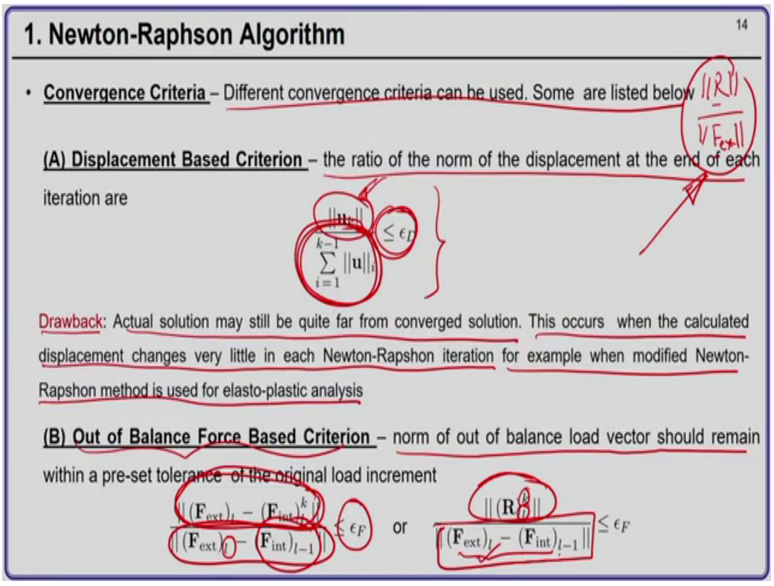
So, this is our Newton Raphson solution algorithm, ok. So, while the ratio of the norm of the residual to the current external force is more than the tolerance N, our number of Newton Raphson steps N are less than the maximum number of Newton Raphson steps per iteration, ok. So, we need to set N equal to 1 here and then what we do first, we compute K at the value X, and then we solve for the displacement u as K equal to minus of R and then we update our current position as X equal to X plus u.

Then, we what we do we compute the stresses, we compute the global internal force vector and then we compute the current value of the residual R as the difference of the external forces minus the internal forces and then we increment N by 1, ok. And, this we do till either we achieve the tolerance with convergence tolerance or we exceed the maximum number of Newton-Raphson steps, ok.

So, if we exceed the maximum number of Newton-Raphson convergence steps then what we do? We have to do some divergence handling, either the Newton-Raphson is not converging which means either it is diverging. So, then we use some divergence handling strategy or we have to cut our load step.

So, you may need to cut our total load delta F by say half or 1 by 4 whatever you choose and then we have to redo that particular load step to see whether we achieve the convergence. And finally, when we are exhausted with all the load step, our solution algorithm ends ok. So, basically the Newton Raphson algorithm is from step 3 3 to 3 4 ok. This is your Newton Raphson and this is the in general total procedure for solving the non-linear response of a material.

(Refer Slide Time: 30:07)



1. Newton-Raphson Algorithm

14

- **Convergence Criteria** – Different convergence criteria can be used. Some are listed below $\frac{||R||}{||F_{ext}||}$

  (A) Displacement Based Criterion – the ratio of the norm of the displacement at the end of each iteration are

  $$\frac{||\mathbf{u}_k||}{\sum_{i=1}^{k-1} ||\mathbf{u}||_i} \leq \epsilon_D$$

  Drawback: Actual solution may still be quite far from converged solution. This occurs when the calculated displacement changes very little in each Newton-Rapshon iteration for example when modified Newton-Rapshon method is used for elasto-plastic analysis

  (B) Out of Balance Force Based Criterion – norm of out of balance load vector should remain within a pre-set tolerance of the original load increment

  $$\frac{||(\mathbf{F}_{ext})_l - (\mathbf{F}_{int})_l^k||}{||(\mathbf{F}_{ext})_l - (\mathbf{F}_{int})_{l-1}||} \leq \epsilon_F \quad \text{or} \quad \frac{||(\mathbf{R})_l^k||}{||(\mathbf{F}_{ext})_l - (\mathbf{F}_{int})_{l-1}||} \leq \epsilon_F$$

Now, there are different kinds of convergence criteria's. In previous slide, we have seen, we have used the convergence criteria as the norm of the residual divided by the norm of the external load vector.

So, this is only one of the convergence criteria. There are some more convergence criteria that can be used and the first one is the displacement based criteria ok. In the displacement based criteria what we do, we check that the norm of the displacement obtained in the kth step divided by the norm of the displacement obtained from all the previous Newton-Raphson iterations is less than some tolerance epsilon D.

If this norm is less than epsilon D, then we say the convergence is achieved ok, but one of the problem with this strategy is that the actual solution may still be quite far away from the converse solution and this will occur when the calculated displacement changes very little in each Newton Raphson iteration, ok.

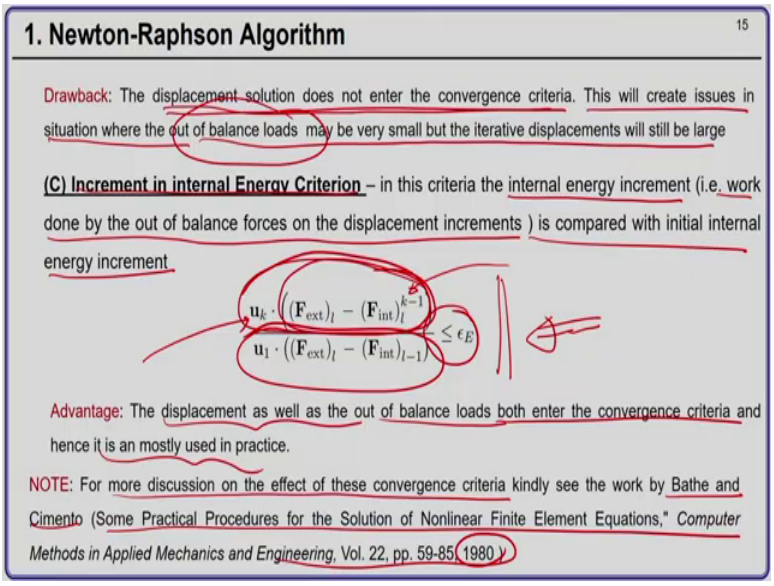
For example, when you are using the modified Newton Raphson strategy in elasto plastic analysis, then if you use the displacement based criteria you will have solution which is far away from the converse solution and eventually your, you will not get the final eventual solution.

Another criteria is the out of balance force based criteria and that is what partially we had in our algorithm that we will discuss in the previous slide. Here, what we do we compare the norm of out of balance load vector ok.

So, this is the out of balance load vector, till the current Newton-Raphson iteration step and we check that the norm of this out of balance force vector normalized by the difference in the external load evaluated at load step l and the internal load evaluated at load l minus 1 should be less than epsilon F, where epsilon F is the convergence criteria ok.

So, in short this is the norm of the residual evaluated at Newton-Raphson step k for load step l and normalized by the norm of the difference of the external and internal forces at load step l and l minus 1.

(Refer Slide Time: 32:58)



1. Newton-Raphson Algorithm

15

Drawback: The displacement solution does not enter the convergence criteria. This will create issues in situation where the out of balance loads may be very small but the iterative displacements will still be large

(C) Increment in internal Energy Criterion – in this criteria the internal energy increment (i.e. work done by the out of balance forces on the displacement increments) is compared with initial internal energy increment

$$\frac{\mathbf{u}_k \cdot ((\mathbf{F}_{\text{ext}})_l - (\mathbf{F}_{\text{int}})_l^{k-1})}{\mathbf{u}_1 \cdot ((\mathbf{F}_{\text{ext}})_l - (\mathbf{F}_{\text{int}})_{l-1})} \leq \epsilon_E$$

Advantage: The displacement as well as the out of balance loads both enter the convergence criteria and hence it is an mostly used in practice.

NOTE: For more discussion on the effect of these convergence criteria kindly see the work by Bathe and Cimento (Some Practical Procedures for the Solution of Nonlinear Finite Element Equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 22, pp. 59-85, 1980)

Again, the problem here is the displacement solution here does not enter the convergence criteria and this will create issues in situation where the out of balance loads are very small, but the iterated displacements maybe very large. If the iterated displacements are very large this will mean that we have not yet achieved the convergence, but because of out of balance loads are very small, then we will say if we use this criteria we will say that our Newton-Raphson step has converged, ok.

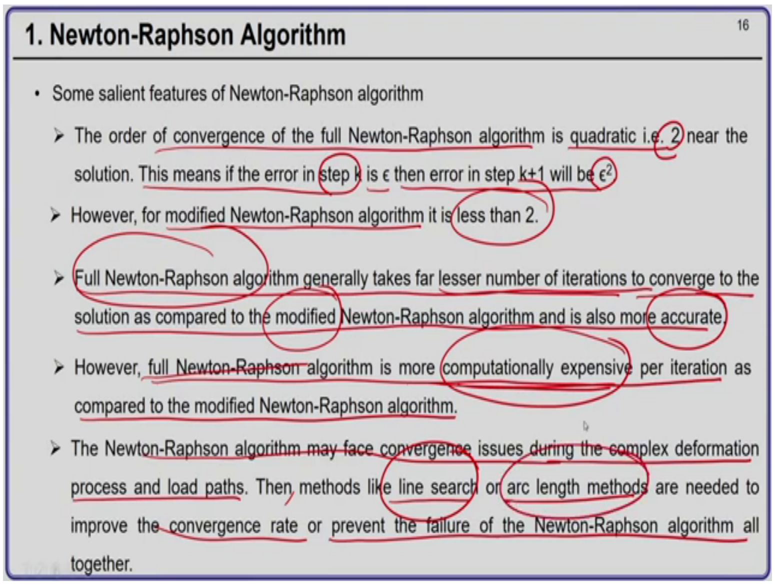
So, again we will have a solution which is far away from the converse solution or the actual solution. Finally, the third one is the increment in the internal energy criteria. So, in this criteria the internal energy increment that is the work done by the out of balance forces on the displacement increment is compared with initial internal energy increment ok.

So, this is the internal energy. So, uk is the solution for the displacement obtain in the kth Newton Raphson step and this dotted with the out of balance load that is the residual obtained till the previous Newton Raphson step divided by the a reference internal energy increment and if this is less than a tolerance epsilon E, then we say our Newton-Raphson has converged.

Now, you can see here, here both the displacement as well as the residual load vector enter into the convergence criteria. And therefore, since the displacement as well as the out of balance loads both enter the convergence criteria therefore, this kind of criteria is most commonly used in the finite element literature, ok.

So, you may well use this criteria along with the previous two criteria's to have a more robust convergence criteria. So, as a note more discussion on the effect of the convergence criteria can be found in the work by Bathe and Cimento in this particular journal paper, ok. So, you can see this journal paper and you can have more discussion on the convergence criteria's used in the Newton-Raphson algorithm and their effects.

(Refer Slide Time: 35:40)



1. Newton-Raphson Algorithm

16

- Some salient features of Newton-Raphson algorithm
  - The order of convergence of the full Newton-Raphson algorithm is quadratic i.e. 2 near the solution. This means if the error in step k is ϵ then error in step k+1 will be ϵ^2
  - However, for modified Newton-Raphson algorithm it is less than 2.
  - Full Newton-Raphson algorithm generally takes far lesser number of iterations to converge to the solution as compared to the modified Newton-Raphson algorithm and is also more accurate.
  - However, full Newton-Raphson algorithm is more computationally expensive per iteration as compared to the modified Newton-Raphson algorithm.
  - The Newton-Raphson algorithm may face convergence issues during the complex deformation process and load paths. Then, methods like line search or arc length methods are needed to improve the convergence rate or prevent the failure of the Newton-Raphson algorithm all together.

Now, we mention some of the salient features of Newton-Raphson algorithm. So, the order of convergence of the Newton full Newton-Raphson algorithm is quadratic that is it is 2 near the solution. What it means is, if the error at step k of the Newton-Raphson iteration is epsilon then the error in the next step will be epsilon square, ok. If you are near to the solution the error goes down quadratically that is of order 2; however, if you are using the modified Newton-Raphson algorithm, this will be less than 2 ok.

So, the whatever convergence form modified Newton Raphson method is less than 2. Then, full Newton-Raphson method generally takes far lesser number of iterations to converge to the solution as compared to the modified Newton-Raphson algorithm and also full Newton-Raphson method is more accurate than the modified Newton-Raphson algorithm.

So, in modified Newton Raphson algorithm what we do? We are, if you remember when we discuss this in our mathematical basics that in modified Newton Raphson method the tangent k is kept constant ok. So, evaluated once at the start of the Newton-Raphson iteration and we keep it fix for all the Newton Raphson iteration. In full Newton-Raphson method at each and every Newton-Raphson step we recalculate the tangent, ok.

Therefore, because we have a fresh tangent at every Newton-Raphson step we iterations needed to converge are lesser. However, full Newton Raphson algorithm is computationally more expensive per iteration as compare to the modified Newton Raphson algorithm. This is because in each step we have to re-compute the tangent matrix k in full Newton-Raphson method and computing the tangent matrix k is a computationally very expensive process, ok.

Therefore, full Newton-Raphson method is computationally expensive per iteration; however, note that we need fewer number of iterations per step for full Newton-Raphson method. So, there is always a tradeoff whether to use full Newton-Raphson or modified Newton-Raphson. So, there are strategies where once we find the convergence has slowed down, we can in between recompute the tangent matrix to accelerate the convergence, ok. So, they are called hybrid Newton-Raphson algorithms ok, which are in between full and modified Newton Raphson algorithm.

Now, the Newton-Raphson algorithm may face convergence issues during complex deformation process and load paths then what will happen? We will have to use methods like line search and arc length methods to improve the convergence rate or to prevent the failure of the Newton-Raphson algorithm all together, ok.

So, in actual practical situation you may have Newton Raphson method behaving erratically, like it will diverge, which means, the error instead of decreasing it will increase or, because of a certain type of the nature of the load displacement curve the Newton-Raphson method may actually fail all together. To prevent this we have to use what is called the line search method or the arc length methods. So, next we discuss, what is the line search method, ok.

Thank you.