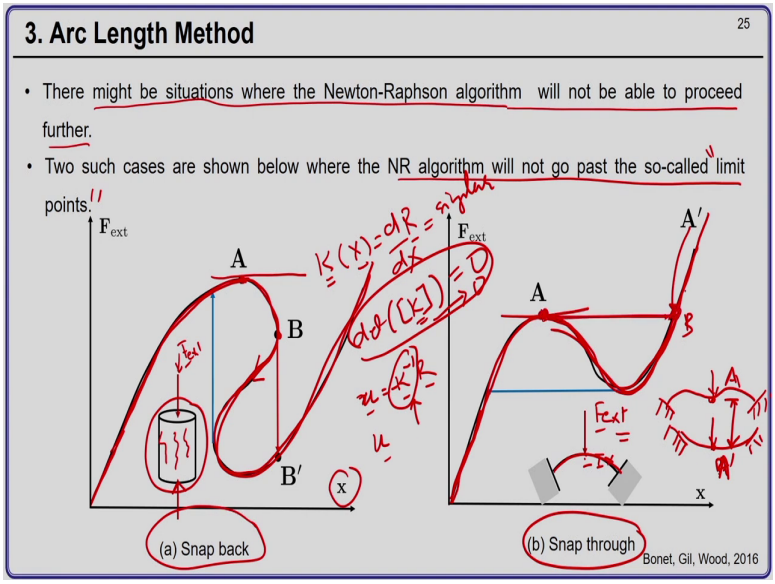
Computational Continuum Mechanics
Dr. Sachin Singh Gautam
Department of Mechanical Engineering
Indian Institute of Technology, Guwahati

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

(Refer Slide Time: 00:42)



So, next we are going to see the method of Arc length. So, there might be situations where the Newton-Raphson algorithm will not be able to proceed further ok. So, what this means is that it is not the convergence problem it is just that the determinant of the tangent matrix will approach 0 which means you are approaching a point with the tangent to the curve it becomes horizontal.

So, they are two cases which are shown below where the Newton-Raphson algorithm will know will not go past the so called limit points ok. So, these are 2 graphs for force external force versus the current position and these are ofcourse hypothetical but they might be present in certain cases.

For example, the one which is shown here is called the snap back behaviour. Here, if you can see the load displacement curve increases up to a certain point and then it reverses its position and finally, it again starts to increase ok. So, this called the snap back because from point B this curves comes backward ok.

The second curve shows the snap through behaviour ok. In this, once the force versus current position graph reaches the maximum value it comes down and then starts to increase again and goes something like this. So, the first case you can you will encounter when you have a thin cylindrical shell for example, and it is compressed by 2 forces on the opposite faces.

And when you it start increasing the forces what will happen the force external force required to cause the deflection of this cylinder will increase till a certain point you will have wrinkling or the local buckling nodes which will come into the picture and then you will have this snap back behaviour.

The snap through behaviour for example, can be seen when you have say a arch and you are applying an external force. So, initially as you start increasing this force the external force versus the displacement say of this point here ok. So, this point displacement let us say it is x ok. So, the force versus current position graph will increase till point A and at that position the arch may look something like this and at this point.

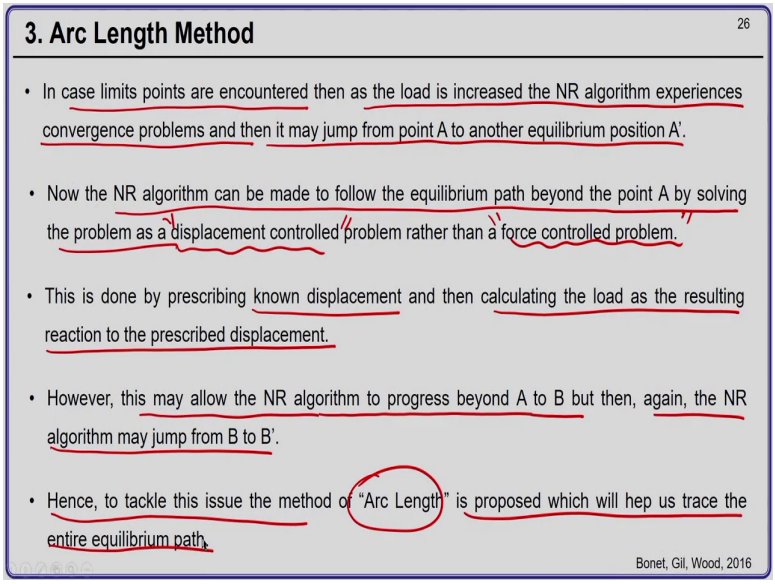
So, this might be your point A and at this point what will happen the force displacement curve goes down which means the arch will occupy ok. So, after this point the next position the arch will occupy will be something like this and this will be this point P here and then ok. So, this point is A dash. So, this point is A dash and in between this part of the curve that you

see will be the force which goes from which takes this arch from position A to position A dash ok.

So, in these 2 graphs you can see that at point A the tangent to the curve becomes horizontal which means the tangent matrix K which is a function of x which is nothing, but del by del x of the residual will become singular or close to singular which means the determinant of matrix formed by the components of the tangent matrix will be equal to 0 or it will approach 0.

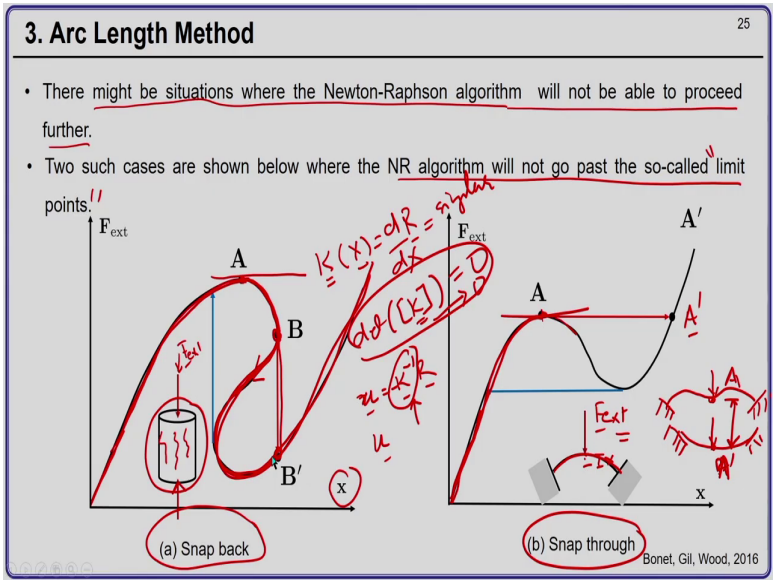
So, therefore, if you are solving for u which is K inverse R minus of K inverse R and if K has a nearly 0 determinant that you cannot take the inverse of K and therefore, you cannot get the solution u ok.

(Refer Slide Time: 05:48)



So, in case of limit points are encountered then as the load is increased the Newton-Raphson algorithm experiences convergence problem and then it may jump from point A to another equilibrium position A dash ok. So, as you can see here the force displacement curve may directly jump from point A to point A dash which is shown here ok. So, this is your point A dash ok.

(Refer Slide Time: 06:16)



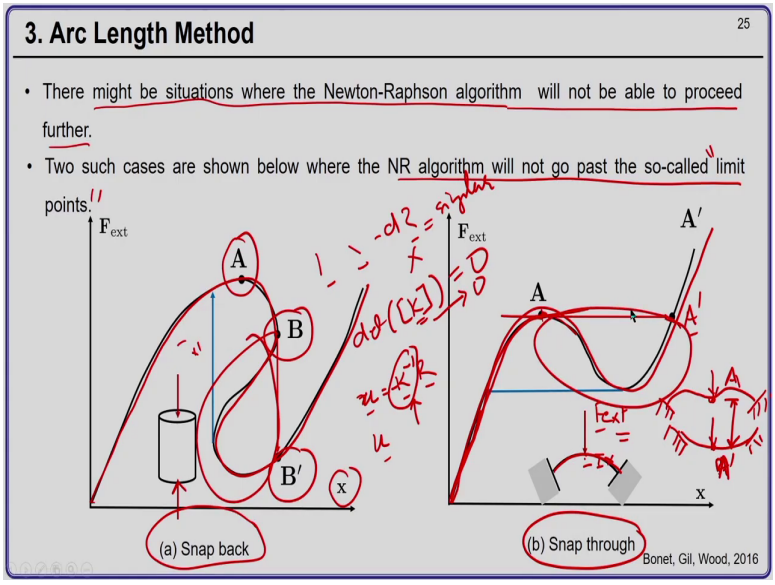
So, in numerical setting you may directly jump from this point to this point ok. Now, the Newton-Raphson algorithm can be made to follow the equilibrium path beyond point A by solving the problem as a displacement controlled problem rather than a force controlled problem ok.

So, rather than applying force you will apply a known displacement in a numerical simulation and then you can go beyond point A and this is done by prescribing known displacement and then calculating the load at the resulting reaction to the prescribed displacement.

However, this may allow the Newton-Raphson algorithm to progress beyond A to B, but then again the Newton-Raphson algorithm may jump from B to B dash ok. So, if you see in this part of the curve.

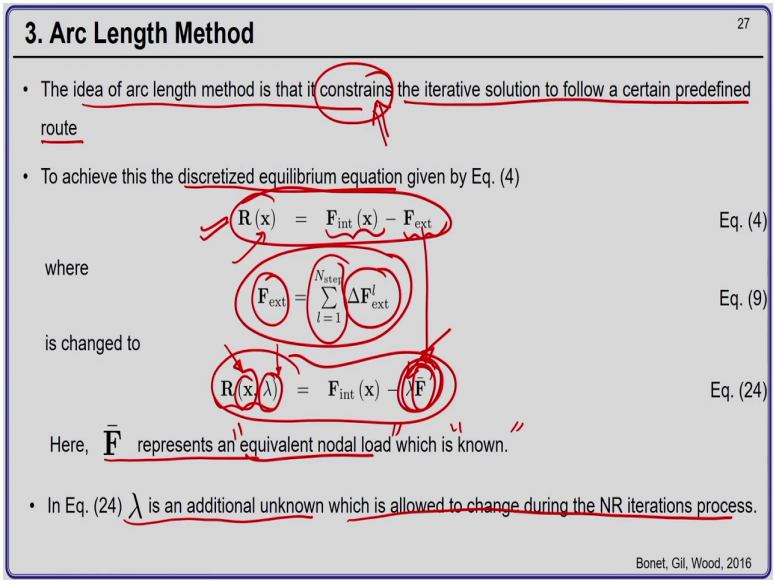
So, if you apply displacement control you maybe you might be able to reach till point B, but then again at point B the curve may jump from B to B dash and you will not be able to trace this part of the force displacement curve ok. So, hence to tackle this issue the method of arc length is proposed which will help us trace the entire equilibrium path. So, if you want to trace this entire equilibrium path this is the equilibrium path.

(Refer Slide Time: 07:55)



So, if you want to trace the entire equilibrium path like this in snap back or snap through then you have to use a technique which is called the arc length method because the traditional Newton-Raphson algorithm will not help you go beyond point A or in the best case will go only till point B, but then you will go at point B dash. So, you will not be able to trace this part of the curve for snap back or this part of the curve for snap through.

(Refer Slide Time: 08:31)



So, the idea of arc length method is very simple. It is that it constraints the iterative solution to follow a certain pre defined route. So, to do this what we do is we take the discretized equilibrium equation which is the form given by equation number 4.

So, you have the residual as a function of a current position equal to the difference of the internal forces minus the external forces and now what we do is we change equation 4 by appending it with a scalar value lambda and then the external load vector which is here becomes lambda times F bar.

Remember the external forces as we discussed in the previous lectures is always applied in a certain number of steps, you do not apply the external forces in one go rather you apply in

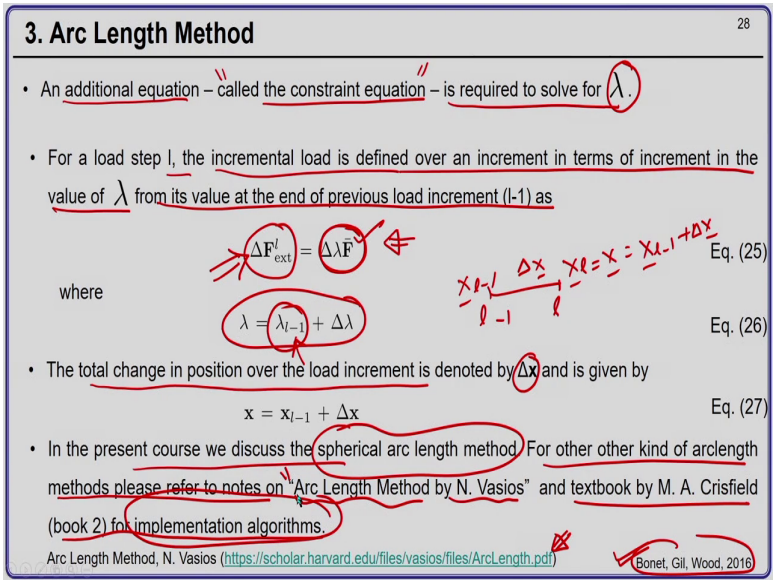
small steps of delta F l. Now, what we do is we replace this external force by a scalar times a known equivalent nodal load vector ok.

So, this F bar is nothing, but an equivalent nodal load vector which is known. So, at the start of the simulation you will give this nodal load vector ok. This might be 1 at a particular position 0 at all other position and then that is what will become your equivalent nodal load vector.

So, now what we have done is we have these unknowns x and now we have introduced another unknown lambda which will help us to trace the complete equilibrium path. Now, the idea of arc length method is we have to place certain constraint on the increment in the value of x and the increment in the value of lambda so that we are able to trace the complete path ok.

So, lambda here is an additional unknown which is allowed to change during the Newton-Raphson iteration process ok. So, lambda keep continuously keeps on changing during the Newton-Raphson iteration so that we are able to achieve the equilibrium after every load increment and then with help of this we are able to trace the complete equilibrium path.

(Refer Slide Time: 11:20)



So, now we have n equations, but now we have increased the number of unknowns from n to n plus 1, where n is the total number of degrees of freedom which is number of node times the degree of freedom per node ok.

Now, we need an additional equation to solve n plus 1 unknown using n equations is not possible. Therefore, we need one more equation and this equation we get by what is called the constraint equation. So, we define a constraint equation which is required to be solved so that we can get the value of lambda ok.

So, for a particular load step l the incremental load is defined over an increment in terms of increment in the value of lambda from its value at the end of previous load increment l minus 1 as delta F l external is equal to delta F delta lambda F bar F bar is known. So, the

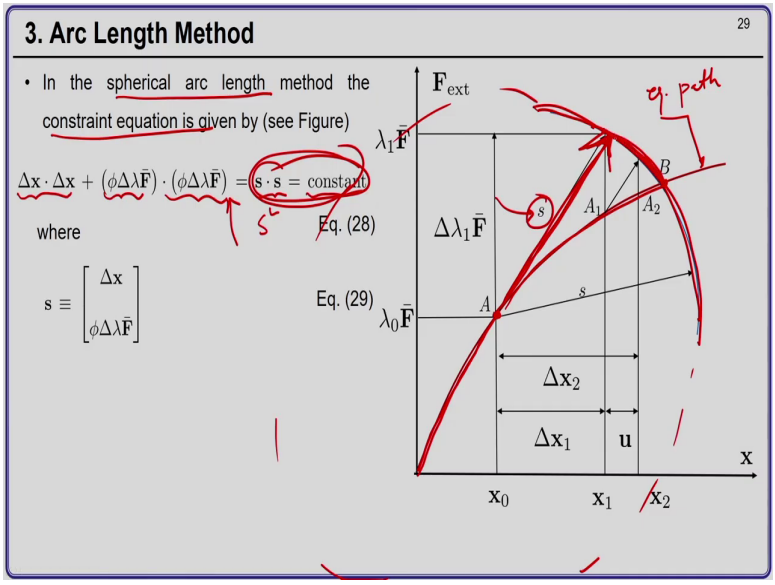
incremental load which is applied from going from load step l minus 1 to l is nothing, but delta lambda times our equivalent nodal load vector F bar ok.

So, F bar is known. So, we have this equation number 25 giving as the incremental load where lambda is nothing but lambda l minus 1 plus delta lambda and the lambda l minus 1 is nothing but the value of the scalar variable lambda at load step l minus 1 ok.

Now, the total change in position over the load increment is denoted by delta x. So, from l to l l minus 1 to l the total change in the position let say is delta x and the position here is x l minus 1. Therefore, the position here x l or written as x will be nothing but x l minus 1 plus delta x.

So, in the present course we discuss what is called the spherical arc length method which is given in this book by Bonet and Wood. For any other kind of arc length method I request you to refer to this notes on arc length method by Vasios the link for which I am giving here or you can see the textbook by Crisfield which is Volume 2 for the implementation algorithms for the arc length method.

(Refer Slide Time: 14:06)



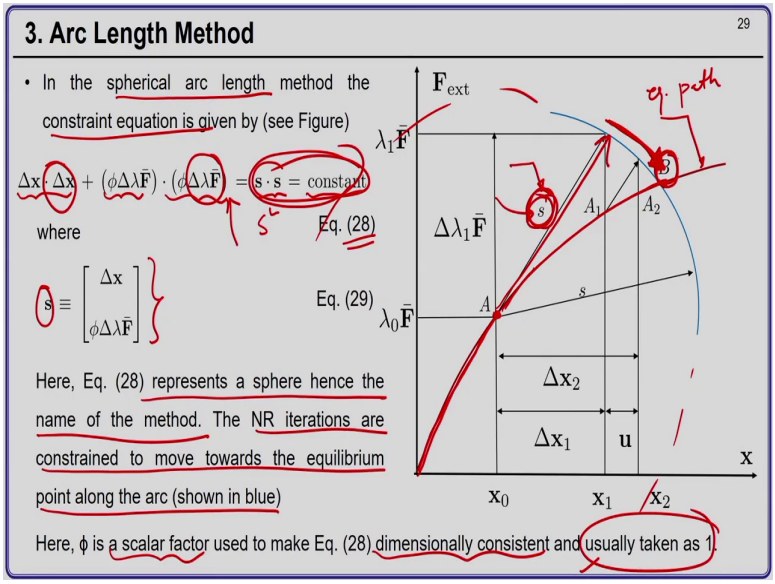
Now, the idea of spherical arc length method is that in this arc length method the constraint equation is given by delta x dot delta x plus a scalar constant phi delta lambda F bar dot delta lambda F bar equal to s dot s which is equal to a constant ok. So, if you see your figure here.

So, this is point A on the so this red curve here is your equilibrium path and you have already solved your equilibrium path or you have traced your equilibrium path till point A. Now, you want to trace or equilibrium path from point A to point B.

Now, what this equation tells us so this is equation of a sphere general sphere ok. So, this blue curve here shows that sphere and now this constant that is s dot s is nothing but this radius ok. So, s dot s is s square. Therefore, this s is root over s dot s and this is the radius of the sphere.

So, there is a big sphere and then your Newton-Raphson method is constrained to move along this. So, rather than constraining your Newton-Raphson method to move along this red path we constrained this equation our Newton-Raphson algorithm to move along this blue path ok.

(Refer Slide Time: 15:53)



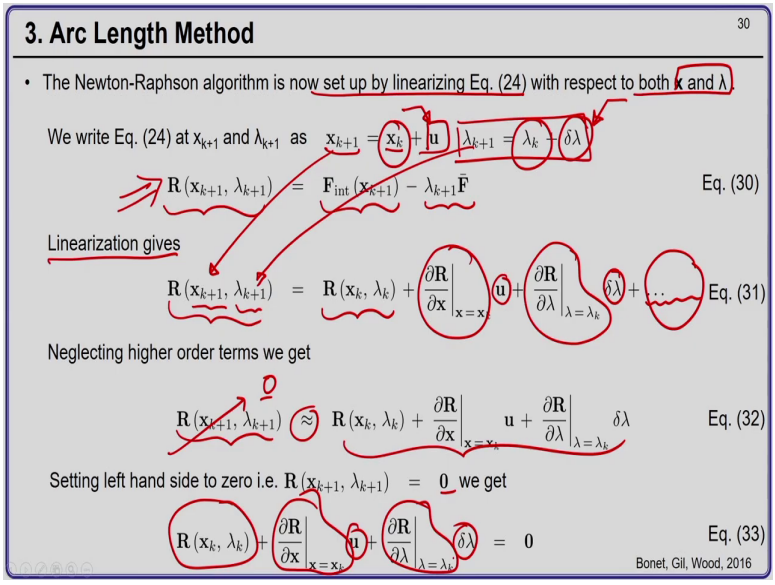
So, Newton-Raphson algorithm has to move along this blue path and then whether you have curved line, whether you have something like this you will be able to trace depending on the length of this vector I mean the length of this arrow which is s. So, you can continuously keep changing this s to trace your equilibrium path where this vector s can be thought of as a vector of delta x and phi delta lambda F bar.

Now, here as I said equation 28 represents a sphere hence that is the name of the method spherical arc length method ok. Now, the Newton-Raphson iterations are constrained to move towards the equilibrium point along this arc ok.

So, that is what I said this is the next equilibrium point and then you have constrained the Newton-Raphson algorithm to move along this blue path rather than constraining it to move along this red path it is constrained now to move along this blue path so that you can reach point B ok.

And now you have control over the say the radius of this sphere which is nothing, but small s and phi here is called a scalar factor which will make equation 28 dimensionally consistent dimensionally consistent means you have a force here and you have a position ok. So, you have 2 different physical quantities. So, phi will make this dimensionally consistent and it is usually taken as 1.

(Refer Slide Time: 17:35)



Now, we can again setup the Newton-Raphson algorithm by linearizing equation number 24 with respect to both current position and the unknown scalar lambda. So, just like previous cases we have to now write our discretize equilibrium equation at k plus 1 th Newton-Raphson iteration where the position at x k plus 1 is position at x k plus the increment from k to k plus 1 and lambda at k plus 1 is lambda k plus delta lambda.

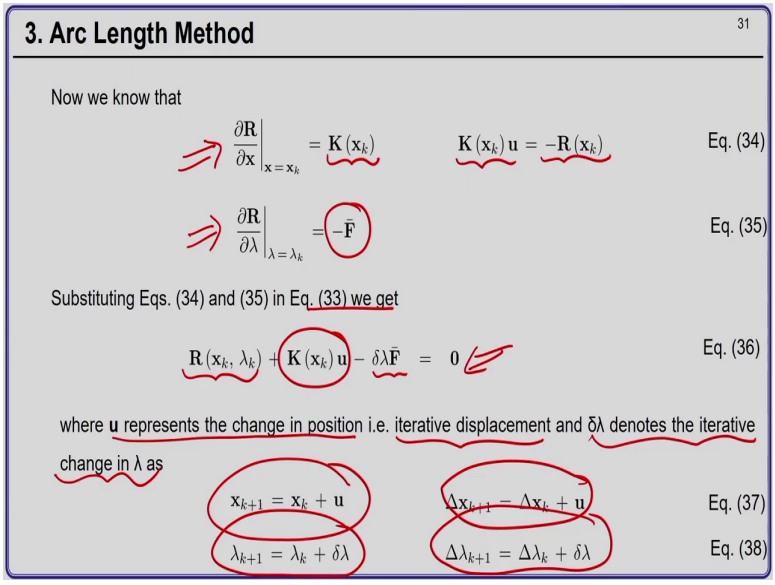
Now, our objective is to find out this u and this delta lambda and we assume that we have solved the previous Newton-Raphson step so that we already have in hand x k and lambda k ok.

So, now the equilibrium equation return at x k plus 1 and lambda x k plus 1 is the internal forces evaluated at x k plus 1 and that is the position at Newton-Raphson step k plus 1 minus lambda k plus 1 F bar.

Now, I can do linearization with respect to both k plus 1 and lambda k plus 1 I can substitute x k plus 1 is x k plus u and lambda k plus 1 as lambda k plus delta lambda and I do the Taylor series expansion. I can write the residual as R x k lambda k plus del R by del x evaluated at x k into u plus del R by del lambda evaluated at lambda k into delta lambda plus we will have higher order terms.

Now, if we neglect these higher order terms we will have the left hand side approximately equal to this right hand side. And now if I make the left hand side go to 0. So, if I make left hand side go to 0, then my right hand side that is residual evaluated at x k lambda x k lambda k plus del R by del x evaluated at x k into u plus del R by del lambda evaluated at lambda k del lambda should be equal to 0.

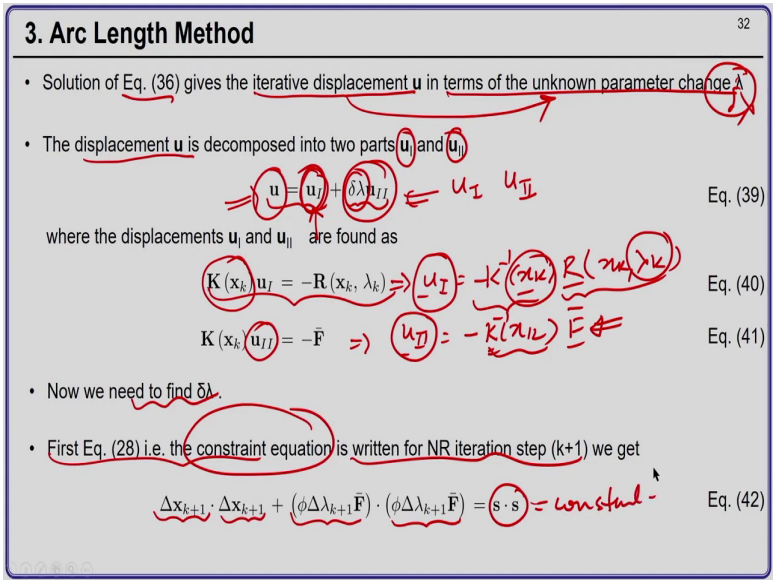
(Refer Slide Time: 19:55)



And now I know that del R by del x is nothing but my tangent matrix at position x k and the tangent matrix times u equal to minus of the residual evaluated at x k and del R by del lambda is nothing but minus F bar. So, now if I substitute 34 and 35 in equation 33 I will get my residual at x k comma lambda k plus k x k u minus del lambda F bar equal to 0.

Now, here u represents the change in position that is it is the iterative displacement and delta lambda denotes the iterative change in lambda and they are given by x k plus 1 equal to x k plus u and delta x k plus 1 is delta x k plus u lambda k plus 1 is lambda k plus delta lambda and delta lambda k plus 1 is delta lambda k plus delta lambda.

(Refer Slide Time: 21:00)



Now, the solution of equation number 36 that is this one. So, the solution of equation 36 gives you the iterative displacement u in terms of an unknown parameter lambda ok. Now, we decompose the displacement u into 2 parts u 1 and u 2 and we write u as composed of one part u 1 plus delta lambda into u 2.

Remember our iterative displacement comes in terms of delta lambda here ok. So, I can decompose my iterative displacement as composed of 2 parts one which is independent of delta lambda and one which is linearly dependent on delta lambda.

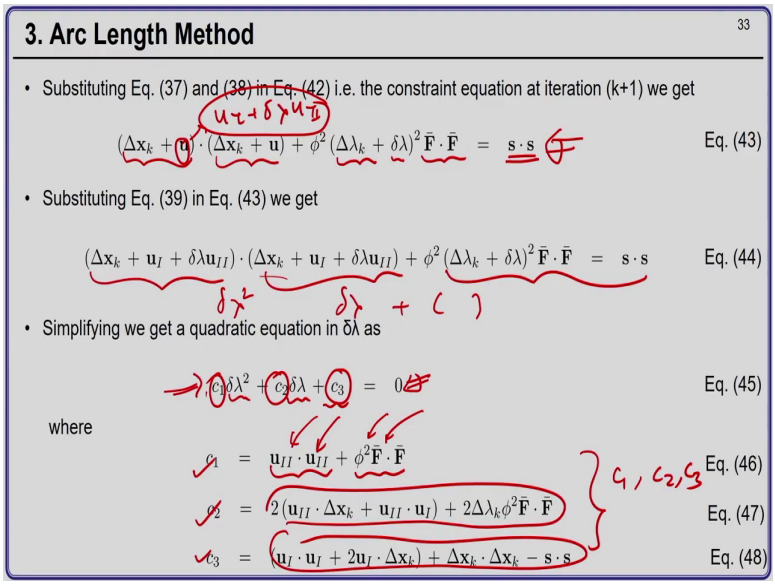
Now, here u 1 is found out from the following equation ok. So, u 1 is k inverse evaluated at ok. So, this is u 1 which is nothing but k inverse evaluated at x k into R x k lambda k. So, I already know my position and lambdas from the previous Newton-Raphson iteration. So, I

can find out the tangent matrix. I can find out the residual and solve to get u I ok. So, u I will be known and now u 2 is nothing but minus of k x inverse F bar.

So, now F bar is initialize at the start of the simulation and k is known. Therefore, I can get u 2 ok. So, from equation 39 I know u 1 and I also know u 2 the only thing left to determined is this term delta lambda. So, how to determine this term delta lambda that is what we are going to see next.

So, we first what we do is we write the constraint equation which is given by equation 28 and we write this at Newton-Raphson step k plus 1 ok. So, we have delta x k x k plus 1 dot delta x k plus 1 plus phi delta lambda k plus 1 F bar dot phi lambda k plus 1 F bar equal to s dot s which will be nothing but equal to a constant. This constant is specified at the start of the simulation ok.

(Refer Slide Time: 23:43)

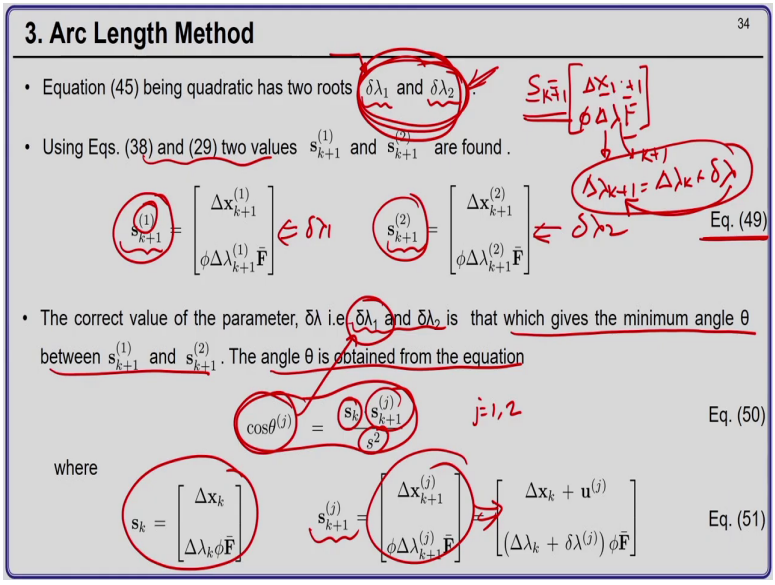


So, now here I substitute x delta x k plus 1 as delta x k plus u ok. So, now, I get delta x k plus u dot delta x k plus u plus phi square delta lambda k plus delta lambda the whole square F bar dot F bar equal to a constant ok. Now, u I can substitute as u 1 plus delta lambda u 2 ok. So, in this equation 43 the only unknown is u and now u is a unknown in delta lambda. So, I can substitute this and I get following expression.

Now, I can open up the brackets and I can simplify and I can collect the terms of delta lambda square delta lambda and constant. In doing so what I get is a quadratic equation in delta lambda square and c 1, c 2, c 3 are the coefficients of this quadratic equation where c 1 is u 2 dot u 2 plus phi square F bar dot F bar.

Now, remember phi is given to us, F bar is given to us, u 2 we have already found out ok. Similarly, here all the terms on the right hand side are given. So, c 2 is known similarly c 3 given by the following expression is also known ok. So, these constants c 1, c 2 are the coefficients of the quadratic equations in 45 c 1, c 2, c 3 are known. Therefore, I can solve for the 2 roots of this quadratic equation ok.

(Refer Slide Time: 25:40)



Let us say these 2 roots are delta lambda 1 and delta lambda 2 ok. So, there are 2 roots delta lambda 1 and delta lambda 2 and once I get this delta lambda 1 and delta lambda 2 what I can do is I know ok. So, I will right here I know s as nothing but delta x phi delta lambda F bar. Now, delta lambda ok.

So, at k plus 1 this will be x k plus 1 and this is delta lambda k plus 1 and delta lambda k plus 1 is delta lambda k plus del lambda. Now, I have 2 roots delta lambda 1 and delta lambda 2. So, if I substitute here delta lambda 1 and delta lambda 2 I will get 2 values of this vector s at Newton-Raphson iteration step k plus 1. So, one is s k superscript 1 and another is s k superscript 2.

So, the first one corresponds to the first root delta lambda 1 and the second one corresponds to the second root delta lambda 2. Now, I have to choose one value of delta lambda. I cannot choose both the values of delta lambda.

So, how to choose the correct value of delta lambda for this what you have to take is I have to take that value of delta lambda which will give me the minimum angle theta between the vectors s k plus 1 superscript 1 and s k plus 1 superscript 2. And this angle theta is obtained from this equation cos theta j equal to s k dot s k plus 1 j divided by s square. s square is a constant s k is known from the previous Newton-Raphson step and therefore, and this j is equal to 1 and 2.

So, from equation 49 I have these 2 terms and I can find 2 values of theta and now the value of theta which is minimum is what will be my solution delta lambda ok. So, now here s k is delta x k delta lambda k phi F bar and s k plus 1 j is given by following equation ok. So, this is your vector ok.

So, once you have found out the minimum value of theta and then you can recognize which one is the your correct solution delta lambda. You will go back and update your value of delta lambda and then you will check for the Newton-Raphson convergence criteria.

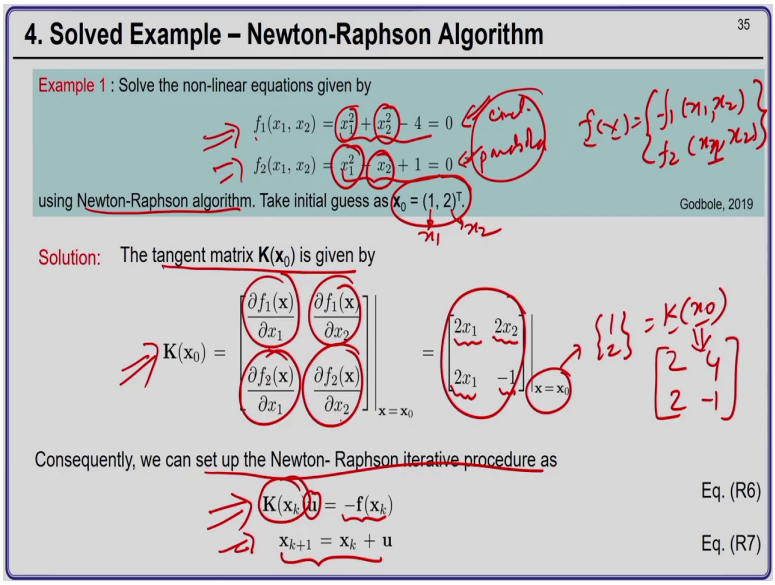
If your Newton-Raphson has not converge when you move to the next Newton-Raphson step and again you go on to find out the new values of delta lambda 1 and delta lambda 2 and this you keep on going till your point B as shown in the graph for the arc length method is reach to a certain degree of accuracy.

So, this completes our arc length method to deal with the failure of Newton-Raphson algorithm. We have already covered the line search method which deals with the slowness of the convergence rate of the Newton-Raphson algorithm.

So, we have dealt with 2 methods one which deal with the flow convergence of Newton-Raphson method that is line search method and we have covered arc length method

we deals with the failure of the Newton-Raphson method to get pass the limit points at the point at which the tangent becomes horizontal or it is becomes vertical as the case may be. So, now with this we next see 2 examples of Newton-Raphson method applied to system of equation.

(Refer Slide Time: 29:45)



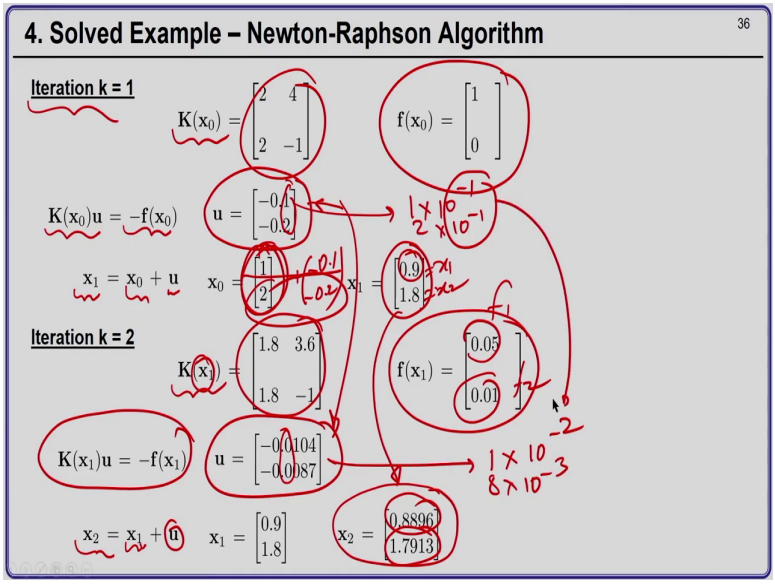
Now, the first equation that we consider is this given by f 1 equal to x 1 square plus x 2 square minus 4 and f 2 is x 1 square minus x 2 plus 1 equal to 0 ok. So, now, you have 1 if you look closely this is equation of a circle and this is equation of a parabola ok. So, what you need to find out is where does the circle and the parabola intersect ok. Now, but you have to find out using Newton-Raphson algorithm let us say your initial guess is given as x 0 equal to 1, 2 ok. So, this 1 correspond to x 1 and this 2 corresponds to x 2 ok.

So, now, first thing we need to find out is our tangent matrix. So, the tangent matrix evaluated at initial point x 0 will be nothing but del f 1 by del x 1 del f 1 by del x 2 del f 2 by del x 1 del f 2 by del x 2 and if you see f 1 here f 1 is x 1 square plus x 2 square minus 4. So, del f 1 by del x 1 will be nothing but twice of x 1 del f 1 by del x 2 is nothing but twice of x 2 del f 2 by del x 1 is nothing but twice of x 1 and del f 2 by del x 2 is nothing but minus of 1 ok.

So, that is your tangent matrix and then when is substitute x equal to x 0 that is 1, 2 you will get your tangent matrix at x 0 ok. Say your when you substitute it here for example, 1, 2 you will get 2 4 2 minus 1 as your tangent matrix. Now, to setup the Newton-Raphson algorithm what we need to do is the tangent matrix evaluated at the Newton-Raphson step x k times the increment in the vector x equal to minus of f and f here the vector f here is nothing but f 1 f 2 ok.

And then once you have found out u you update your vector x as x k plus u. Now, let us do few iterations of Newton-Raphson algorithm to see how we approach.

(Refer Slide Time: 32:24)

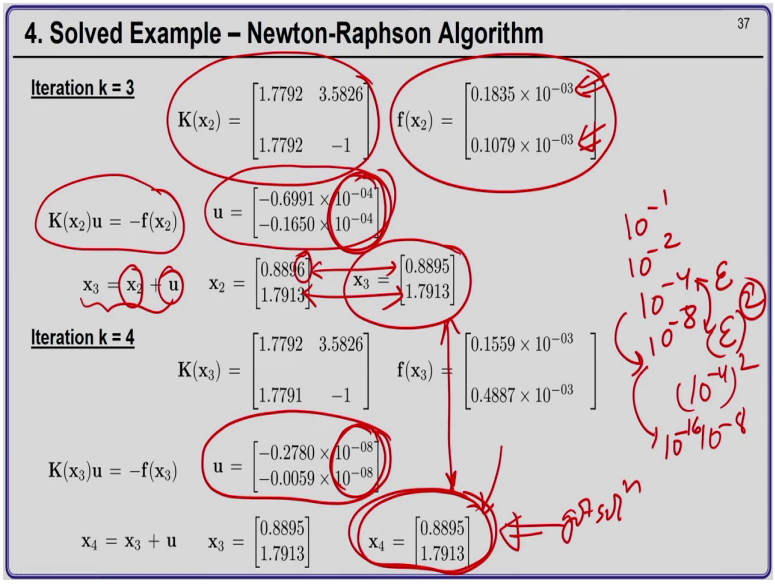


So, iteration 1 let us say k equal to 1. Your tangent at x 0 is 2 4 2 minus 1 in your function at x 0 is 1 0 ok. Now, you solve k u equal to minus f and we get the solution u as minus 0.1 point minus 0.2. So, my new estimate for the solution x 1 is x 0 plus u x 0 is 1 by 2. Therefore, if I tell if I add minus 0.1 and minus 0.2 I will get 1 minus 0.1 is 0.9 and 2 minus 0.2 I get 1.8. And then I do the next Newton-Raphson step.

In the next Newton-Raphson step, I recalculate my tangent at points x 1 ok. So, I will calculate my new tangent at point x 1 equal to 0.9 and x 2 equal to 1.8 and I get this as my tangent matrix and this is my external force vector or my given function values ok. This is the value of the function f 1 and this is value of the function f 2 at x 1 ok. Now, I solve for u again and I get my u as minus 0.0104 and minus 0.0087. Now, you can see clearly that u is decreasing already ok. Initially the 0s were after decimal place.

Now, there are zeros at the first decimal place after first decimal place ok. Now, I get my new estimate of the solution by adding the solution u to the previous estimate and my new estimate comes out to be 0.8896 1.7913 (Refer time: 34:29) you can compare here ok. So, we have drastically come down to 0.8896 1.79 with respect to initial solution, but with respect to the previous solution it has there is some marginal improvement ok.

(Refer Slide Time: 34:45)



So, now, I do the third iteration ok. So, I again find a tangent matrix at x 2 and my function at x 2. Now, it see the value of the function has become very small ok. It has started to approach a very small value which means we are starting to approach the solution ok. Now, again I solve for u and I get my u as this value.

Now, I see now I am in the range of 10 is to power minus 4 this is very small value. Now, if I add this u to previous estimate of the solution. The new estimate of the solution will be

0.8895 0.7913. If you compare this with the previous solution you see there is a change only at the last decimal place while there is no change estimate for x 2 there is a change in the value of x 1 only at the fourth decimal place.

So, let us to one more iteration and after this I find that my solution u is now 10 is power minus 8 of the order of 10 is power minus 8. So, my new solution x 4 is 0.8895 1.7913 and if I compare till fourth decimal place my solution now does not seems to change.

Therefore, after 4 Newton-Raphson steps I have got my solution ok. So, if you fix your convergence tolerance to a much higher value you may need to do some more extra Newton-Raphson iteration steps ok. But, you can see already the error here ok.

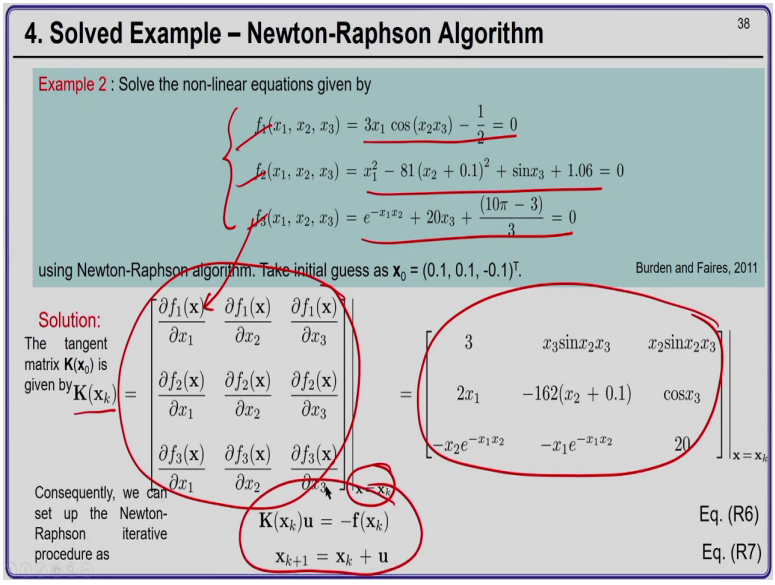
So, you can write this as 1 into 10 to power minus 1 and 2 into 10 is to power minus 1. So, it was 10 is to power minus 1 if you see you can write here is 1 into 10 is to power minus 2 and 8 into 10 is to power minus 3.

So, you have jumped from minus 1 to minus 2. Now, from minus 1 minus 2 you have jump to 10 is to power minus 4 and then you have jumps to 10 is to power minus 8. So, 10 is to power minus 1 10 is to power minus 2 10 is to power minus 4 10 is to power minus 8.

So, if you see here if the error here is was of the order of 10 is to power minus 4 in the next Newton-Raphson step when you have got your solution you get the order of the order of epsilon square ok. So, 10 is to power minus 4 square 10 is to power minus 8 ok. So, this is what we call the quadratic convergence ok.

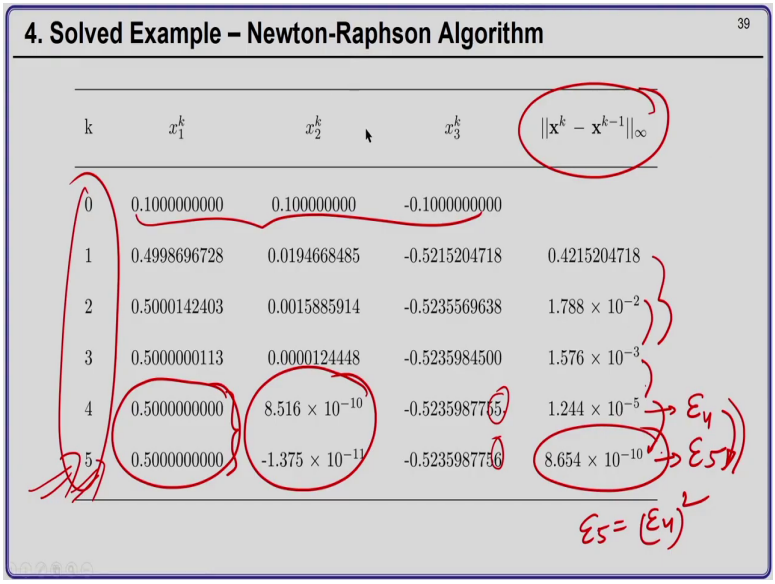
So, when your Newton-Raphson algorithm reaches the solution the error goes down quadratically ok. If you do one more step yours error would be of the order of 10 is to power minus 16 ok. So, you will already be near to the machine precision level ok.

(Refer Slide Time: 37:37)



Second example that we do is now we have instead of 2 functions we take 3 functions and these are these highly non-linear functions ok. So, instead of solving this I will just show you that the tangent matrix is given by this 3 by 3 matrix. So, if you take the derivative of these functions here you will get the following expression for the tangent matrix and this you have to evaluate for any given Newton-Raphson step.

(Refer Slide Time: 38:13)



So, again we set up the Newton-Raphson algorithm like this and when we carry out the Newton-Raphson iterations we carry out 5 Newton-Raphson iteration. My initial estimate of the solution is 0.1 0.1 minus 0.1 and my measure of the error is the infinity now ok.

Now, if I do this I see that after 5 Newton-Raphson iterations my error has gone down to 8.654 into 10 is to power minus 10 and you can see already at the first solution has converged till 9 th decimal place my second solution is converged till 10 decimal place and even my third solution has converged at least 12 9 th decimal place and only there is small change at the 10 decimal place.

You have 5 here and you have 6 here. For all practical purpose you have got the solution in the fifth Newton-Raphson step and you can already see as you went towards the solution your error. So, this is the error and approximately if the error here is epsilon 4 and here is epsilon

5. So, epsilon 5 is nothing but epsilon 4 square that is from step 4 to step 5 your error has gone down quadratically ok.

So, this is what we mean when we say that Newton-Raphson algorithm converges quadratically near the solution. So, near the solution you will have this kind of quadratics convergence, but away from the solution you will see it is more or less a linear convergence.

So, with this example we end this particular module ok.

Thank you.