Introduction to Finite Element Method
Dr. R. Krishnakumar
Department of Mechanical Engineering
Indian Institute of Technology, Madras

Lecture - 31

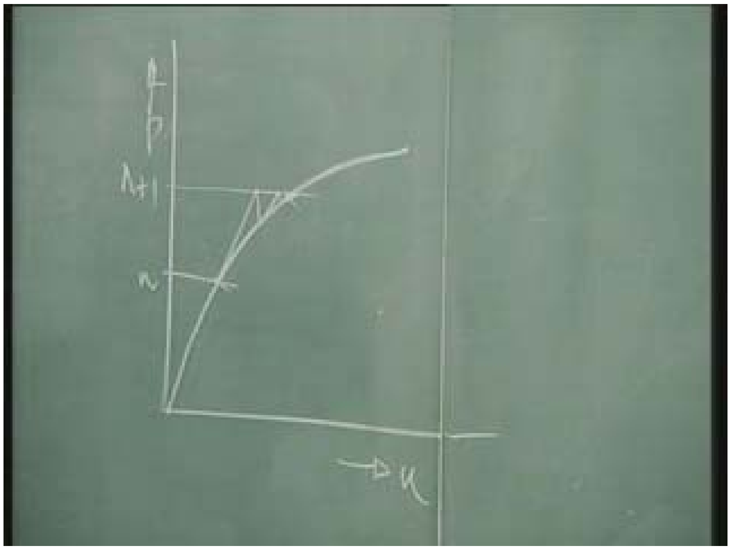
We were looking at non-linear problems in the last class and we said that we will continue with the non-linear analysis.

(Refer Slide Time: 1:06)

We identified the type of problems which could be called as non-linear and we said that geometric material non-linearities due to varying boundary conditions like contact would induce a problem to become non-linear. That is where we stopped and we have to go further with non-linearities. As all of you know that many of the manufacturing problems especially, are problems which invoke non-linear analysis or in other words you have to carry forward this analysis in order to simulate, say for example, the manufacturing process. We have already seen examples of it. We saw a welding example; right in the beginning, we also saw forging and so on, though without going into details, because that topic of simulation, complete understanding of the simulation of manufacturing process is a very big affair, because you have to know so much about plasticity especially finite deformation, large deformation plasticity and so on. So, I am not going to go into the details of all these aspects but at least I want to tell you, given a package what are the things that you should look for or what are the things that you have to know in order to, for example, run a non-linear problem.

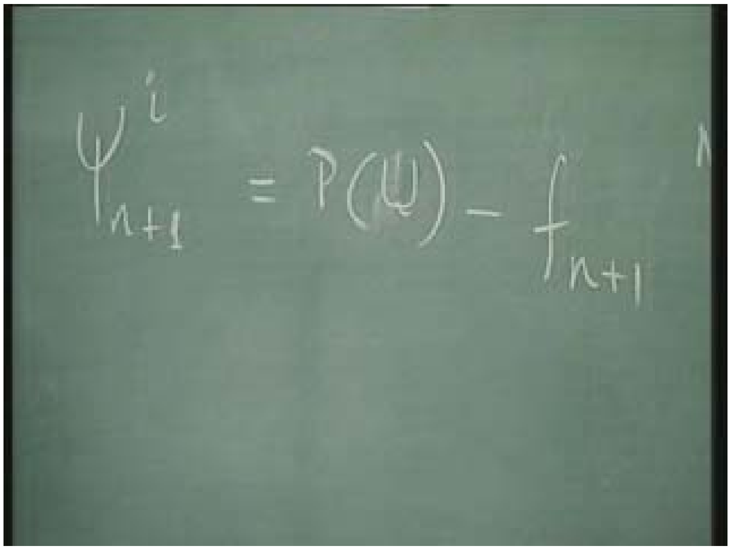
So, my emphasis in this course is going to be only from that angle, but not from the angle of development of a finite element code itself as far as non-linear analysis is concerned, basically because of the continuum mechanics background that is required in order to do that is very high. But nevertheless, without losing rigour, without losing what is to be known in order to run it, we will carry forward our study on non-linear analysis. Now we identified, from our physical arguments we identified lot of things.

(Refer Slide Time: 3:10)



You remember that we identified, when we looked at say load deflection curve that the curve is non-linear. This we identified. We identified that if you want to go from here to here to this position, if you want to go from one position to the other we identified that is this is the load that I want to achieve for a particular step at particular increment, then, I said that from the previous increment at n when I go to n plus 1, I do an iterative procedure in order that we reach that point until convergence takes place.

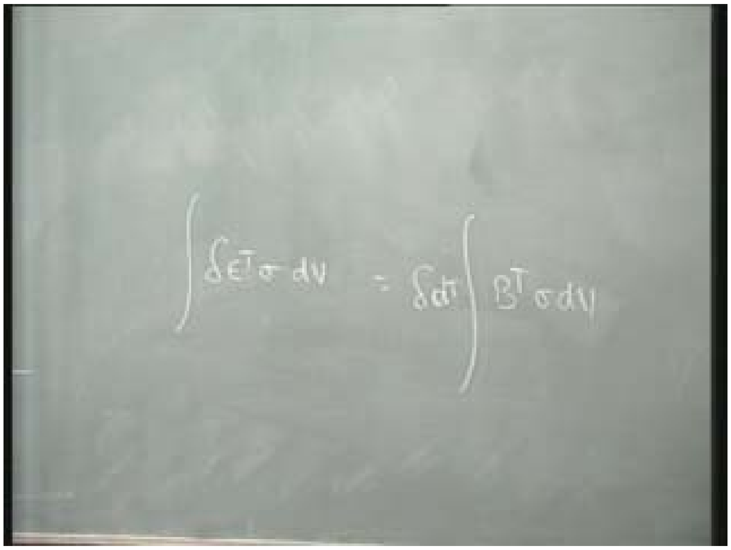
(Refer Slide Time: 4:06)



$$\psi_{n+1}^{i} = P(U) - f_{n+1}$$

Putting it mathematically, I can say that the error psi, error psi say at n plus 1, at say an iteration i is equal to some internal loads which is, what is the matrix that we used? Capital U; for the displacements capital U. Yeah, long ago we used the vector for the complete degrees of freedom U. So, you can say that the error is say the internal forces minus the externally applied forces. Externally applied forces can be a function of displacement or need not be a function of displacements. Let us say that for the time being we say that it is not a function of displacement and say that f, say, n plus 1. Is that clear? So, what is this? This is the internal forces; reason for it is the presence of stresses, is the presence of stresses. It is very simple to see what it is. How do you see that? Even from our virtual work principle you can see that, say you remember what is the equation we wrote?

(Refer Slide Time: 5:36)

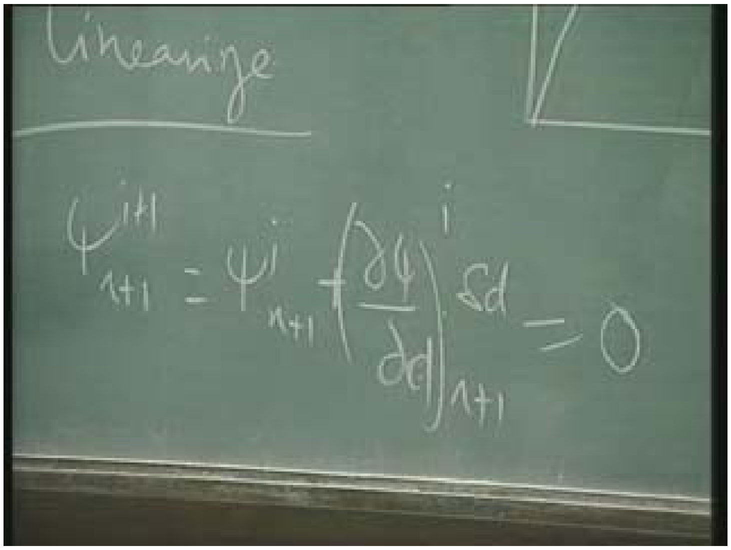


$$\int \delta \epsilon^T \sigma \, dV = \delta d^T \int B^T \sigma \, dV$$

Integral delta epsilon transpose sigma dV. We started something like that in our previous case, remember that. That is how internal virtual work and external virtual work when we did that we had a term delta epsilon transpose sigma and delta epsilon transpose can be written as delta u transpose B transpose sigma dV or delta d transposes rather, B transpose sigma dV in finite element terminology. Go back and refer to this and so you see that delta d transpose B transpose sigma dV. Now look at this term. This looks like this is a force and that is the displacement. This is the virtual displacement and this is the force. So, you can easily identify that B transpose sigma dV would be an internal force. Actually from here only we get our famous stiffness matrix sigma substituted in terms of what? Simple; d B. d into epsilon, epsilon is B u. That is all we get. So, this is the force term. What is this? This is nothing but this P which I have put here. B transpose sigma dV minus f. What is f? f is the load that is applied. So this is f_n, f_{n+1}.

Now what I am going to do is I am going to linearize. Note the word.

(Refer Slide Time: 7:22)

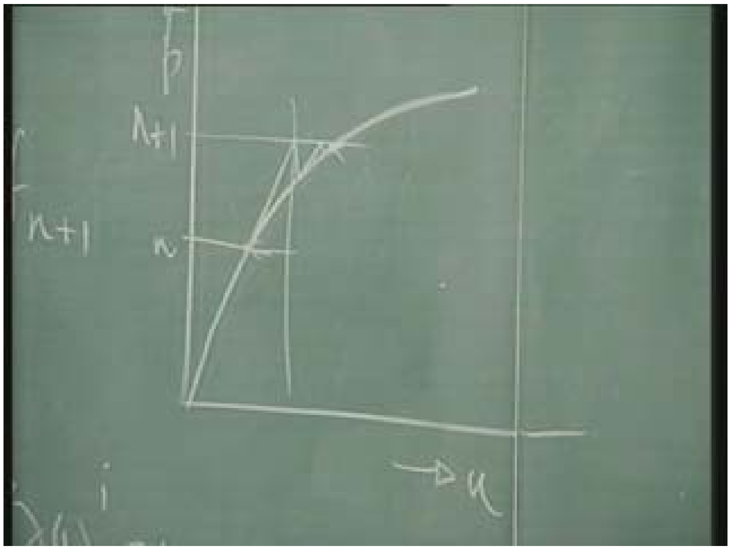

Linearize
$\psi^{i+1}_{n+1} = \psi^{i}_{n+1} + \left(\frac{\partial\psi}{\partial d}\right)^{i}_{n+1} \delta d = 0$

It is one of the most famous words in non-linear finite element, linearization. Linearize about this point say I; for the first step about say n where i is equal to zero. What do I mean by linearization? Linearization simply means that I am going to write down a Taylor series approximation about that point, about this point say for example or about this point, I am going to Taylor series approximation I am going to write down, stopping it after the first term. Taylor series approximation you can keep on writing, higher order terms; d squared y by dx squared 1 by 2 d squared y by dx squared and so on. So, I am going to, what I am going to do is I am going to linearize, which means that I am going to stop with the first term. What does it mean?

It means that if I have to go to say psi i plus 1 n plus 1, this is equal to psi i at n plus 1 plus what is it? Dow psi by dow, say, let me limit first to the to one element, though it is U, just because we did the similar thing. So d is what is the corresponding degrees of freedom for one element. Dow d Dow psi by dow d, displacements. You can write it for dow U, but then it will be for a complete system; dow d into say delta d, where dow psi by dow d is taken at the point i. This means iteration and this means increment. That is the Taylor series approximation about psi plus 1 n plus 1. That is in other words, I obtain, sorry, the error at i plus 1 through the error at i or from the error at i. So, this is equated to zero. This you call linearization. It will not be zero. The error will not disappear, because I have only taken the first term. If I keep calculating all terms 1 by 2 psi squared psi, I mean dow squared psi and so on, it may go to zero.

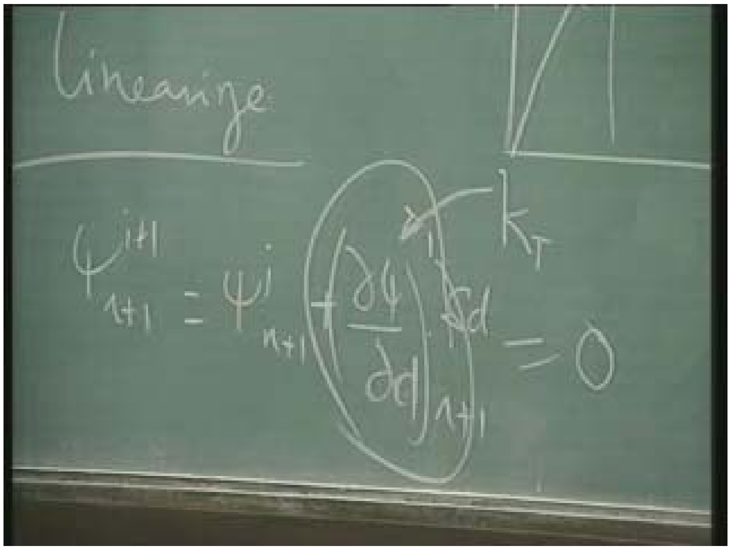
But since I have stopped with the first term, it will not go to zero. Is that clear? This will not go to zero. In other words, when I go from i to i plus 1, there is a possible chance, there is a good chance of an error. But the error may disappear as my i keeps increasing. Is it clear? As my i keeps increasing, the error will come down. In other words, what does this step physically mean?

(Refer Slide Time: 11:18)



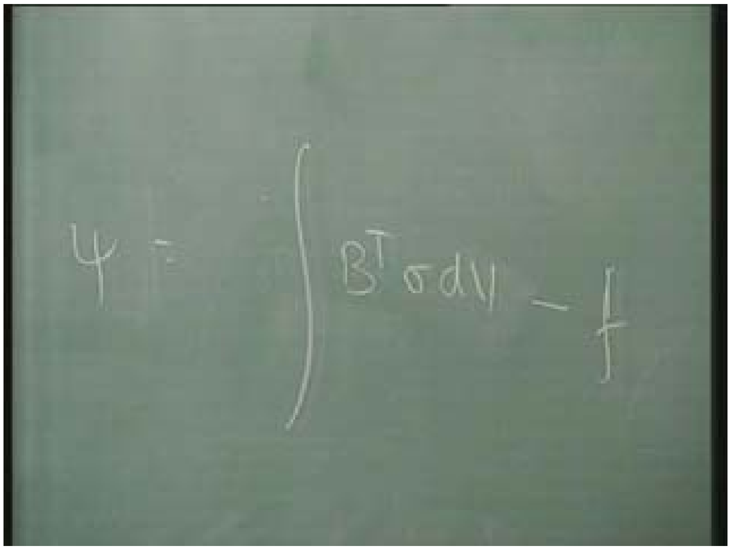
Physically simply it means that get the slope at that point, because what is dow psi by dow d, from here? It is simply dow P by dow d. What is dow P by dow d at i? It is nothing but the slope; P versus d. Dow P by dow d is nothing but the slope. Is it clear? So, I get the slope at this point and call this slope as tangent stiffness. This term here is called as tangent stiffness or K_T.

(Refer Slide Time: 12:02)


linearize
$\psi^{i+1}_{n+1} = \psi^{i}_{n+1} + \left(\frac{\partial\psi}{\partial d}\right)^{i}_{n+1} \delta d = 0$
K_T

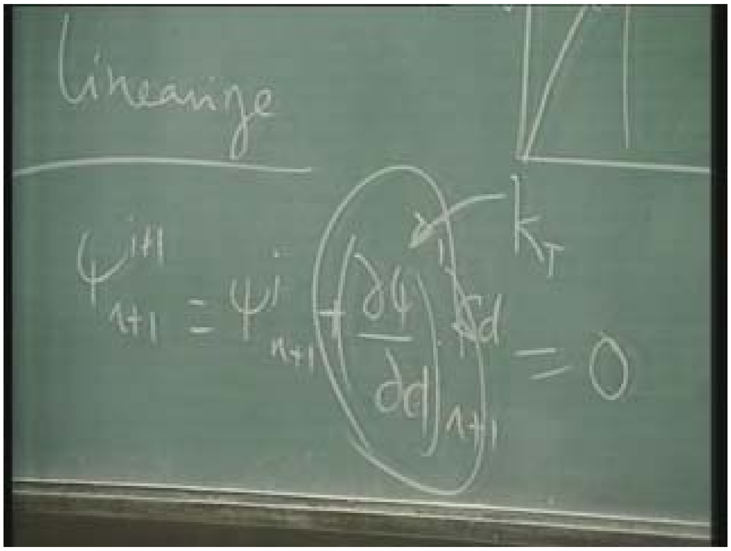
Is it clear? We call this as tangent stiffness or K_T. Is it clear? How do I get here in this term? How do I get this? This is very simple. For 1D I can very easily understand what is happening. It is the slope at that point. What is it here?

(Refer Slide Time: 12:40)



Say, suppose I say that B transpose dV that is the error term minus say f, what would be my stiffness matrix? What would be my stiffness matrix? So, I have to say that dow psi by dow d or dow epsilon is equal to into delta epsilon or delta d is equal to zero. No! One minute let me extend this. Let me come back here. If you do not understand, I will come back here and then go to that problem. Just a second, I have skipped one step. Let me make this step clear, so that if there is a problem you can easily understand. Let me remove this for a minute, let me extend it to get the, from physical to mathematical picture. What does this equation signify?

(Refer Slide Time: 13:48)

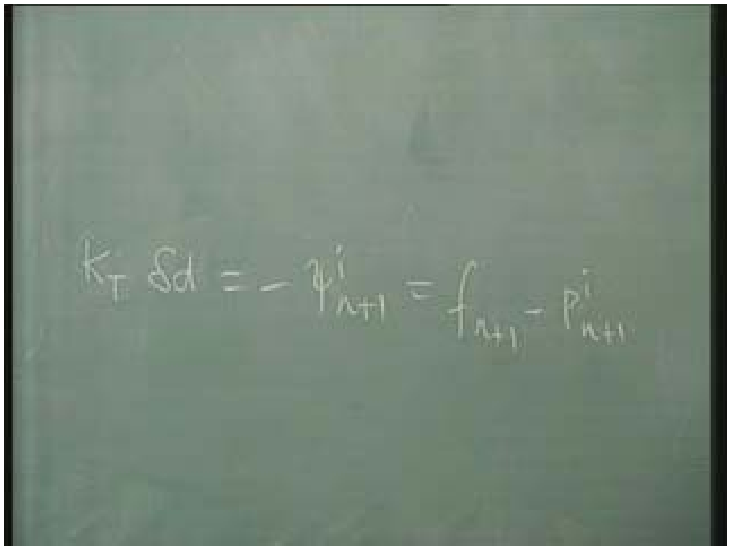


linearize

$$\psi^{i+1}_{n+1} = \psi^{i}_{n+1} + \underbrace{\left(\frac{\partial \psi}{\partial d}\right)^{i}_{n+1}}_{K_T} \delta d = 0$$

It signifies that K_T delta d is equal to minus psi i n plus 1, is equal to minus psi i n plus 1.

(Refer Slide Time: 14:09)

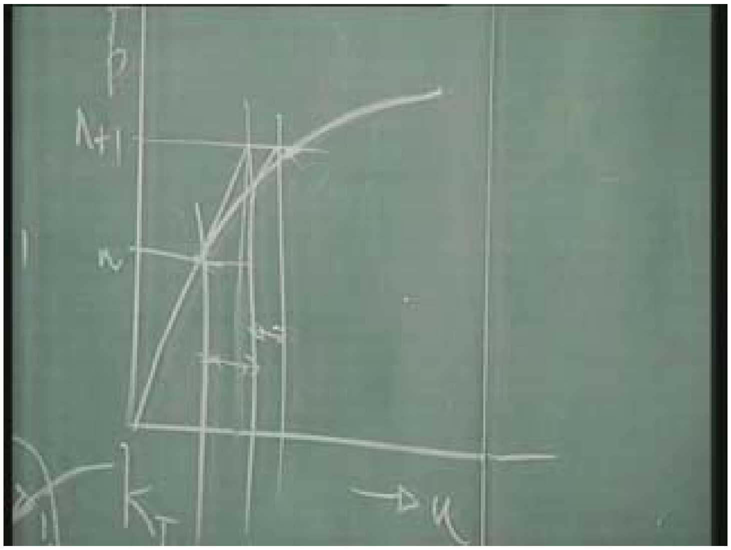


$$K_T\, \delta d = -\psi^{i}_{n+1} = f_{n+1} - P^{i}_{n+1}$$

Is it not? What is psi i n plus 1? Get it from this equation here, first equation, but minus of that, so f_{n+1} minus p. This is equal to f_{n+1}minus P. Let me call this as i, say n plus 1. That is the error term. In other words, what is that you get out of it? I get delta d by solving it, because I know the right hand side, I know K_T. K_T is right now I know it, because for 1D, K is just the slope of the stress strain curve. So, K_T I know. K_T into delta d is equal to f_{n+1} minus p i n plus 1. Is it clear, any question?

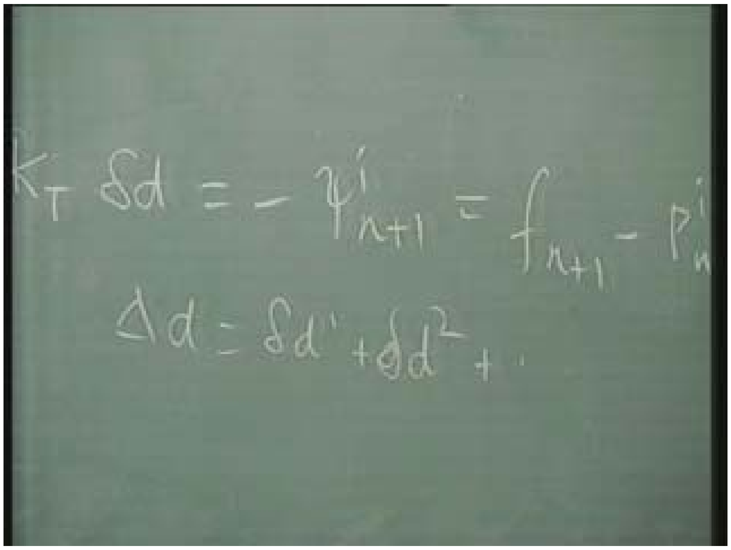
Let me say that I start from say i is equal to zero. That means I am here. 1516I calculate it and calculate K_T, at that point calculate delta d, find out the error at that point after solving. My error keeps decreasing and that we have discussed in the last class itself. When my error is very low or in other words when I say that it has converged, I would stop calculating further delta d's or in other words in one increment, in one increment my delta d, what is delta d here?

(Refer Slide Time: 15:51)



Say for example, this is the first delta d, this is my second delta d and so on. That is first, that is the second one and so on. So, I keep sigma, keep summing up this delta d's, so that at the end of my iterative process, in other words at the end of my convergence, I would be able to get the solution for the force at n plus 1.

(Refer Slide Time: 16:33)

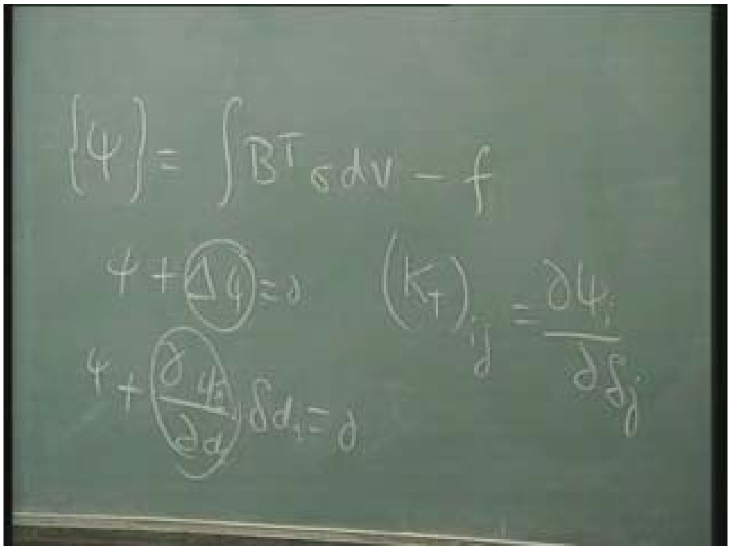


$$K_T \delta d = -\psi^i_{n+1} = f_{n+1} - P^i_n$$

$$\Delta d = \delta d^1 + \delta d^2 + \cdot$$

The total delta d is equal to delta d 1 plus delta d 2 and so on. This is for a particular step. We have not come to stress. I will come to that in a minute. Before we go there because 1D is always easy to understand you may feel that we do not understand multi axial case. Because it is very simple, so, I want to just jump right ahead to a multi axial situation and see how the same stiffness matrix is generated for multi axial case. Any question? No question.

(Refer Slide Time: 17:27)



$$\{\psi\} = \int B^T \sigma \, dV - f$$

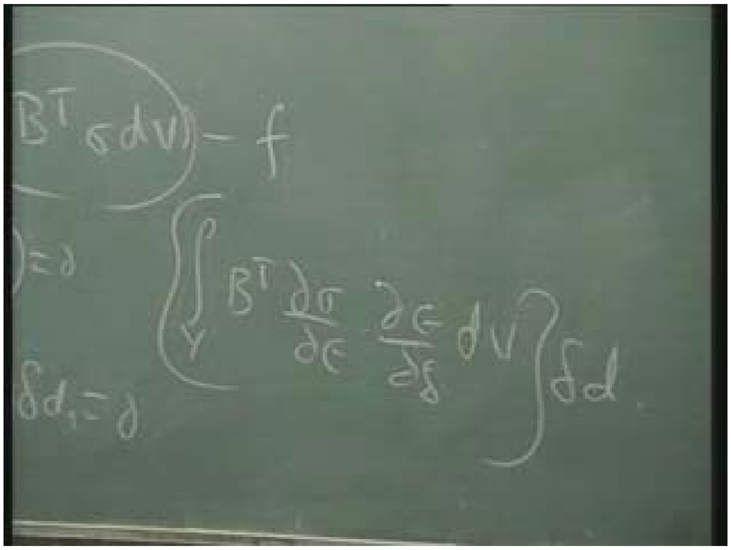
$$\psi + \Delta\psi = 0 \qquad (K_T)_{ij} = \frac{\partial \psi_i}{\partial \delta_j}$$

$$\psi + \frac{\partial \psi_i}{\partial d} \delta d_1 = 0$$

I can write down in that case, psi is equal to integral B transpose sigma dV, now, minus say f. All of them are vectors. Please note that all these guys are vectors. Psi is a vector, if you want add, add put all everywhere, but I think by this time you should be quite clear. Now, what is it that I do now? I do a linearization, I do a linearization. So, I say that psi plus say delta psi is equal to zero where delta psi is that is obtained by my first linearization. So, that is delta dow of psi by dow d into delta d is equal to zero. That is psi plus dow psi by dow d into delta d is equal to zero. Please note that this will be a matrix dow psi 1 by dow d 1 dow psi 1 by what is it, you know, if you expand this? It is a complete expansion. So dow 1 by dow d 1 into delta d 1 plus 1 2 delta d 2 and so on. Then dow here 2 1 and so on. In other words, both of them being vectors, I get a when I differentiate this I get a matrix here and that is the K_T matrix. There in 1D it was one value because delta d was only one value. From this figure the delta d was only one value. Now delta d is, even for an element it is not one value and so, it is a vector. This is a vector, so this becomes a matrix, where K_T ij can be said as something thing like that. Is that clear? I mean, it is nothing but a Taylor series with a number of variables d 1 d 2 d 3 etc. Yeah, any question?

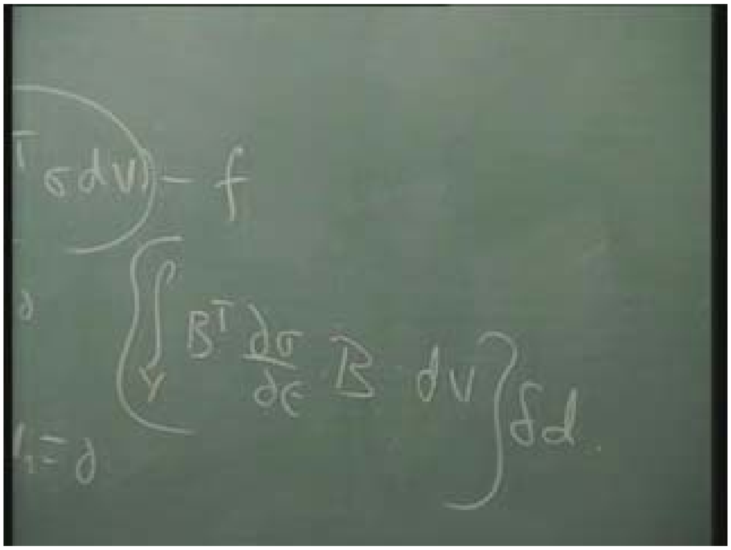
How do I do this trick here? How do I do that trick there or how do I get now the K_T matrix knowing this? That is all. Yeah, any clues? Simple; dow sigma by dow epsilon into dow epsilon by dow d into delta d, because now you can look at this whole thing as a function of sigma.

(Refer Slide Time: 21:04)


$B^T \sigma dV) - f$
$)=0$
$\left\{\int_V B^T \frac{\partial\sigma}{\partial\epsilon}\frac{\partial\epsilon}{\partial\delta} dV\right\}\delta d$
$\delta d_1 = 0$

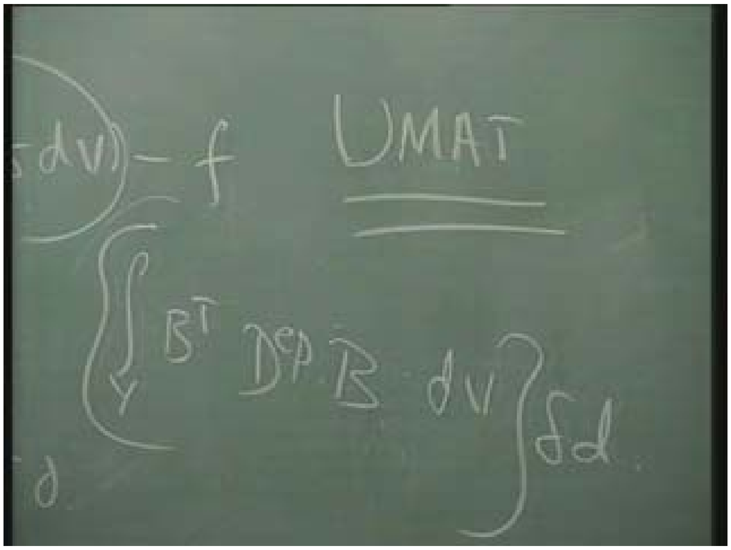
This term here, this term here when I apply to this term becomes integral v B transpose dow sigma by dow epsilon, because sigma is a function of epsilon. We know it. That is the constitutive equation dow sigma by dow epsilon into dow epsilon by dow d. This also we know, because strain is a function of displacement into delta d? Is it clear? No, exactly that is the K_T matrix. This is, what is inside here is, the K_T matrix. What is dow epsilon by dow d? What is this? What is that from finite element?

(Refer Slide Time: 22:12)


T σ dV) − f
∂
$\left\{\int_V B^T \frac{\partial\sigma}{\partial\epsilon} B \, dV\right\} \delta d.$
₁ = ∂

Yeah, B matrix epsilon is B times T. Yeah, any question? B matrix; so, B transpose dow sigma by dow epsilon B dV, period; very simple. Is that clear? Now, the whole important thing here is what is this? This is nothing but this is what my K matrix is, K_T matrix. The whole important thing here is that recognize what introduces non-linearity. Look at that equation. What is that? What is that which induces non-linearity here? It is dow sigma by dow epsilon.

(Refer Slide Time: 23:07)



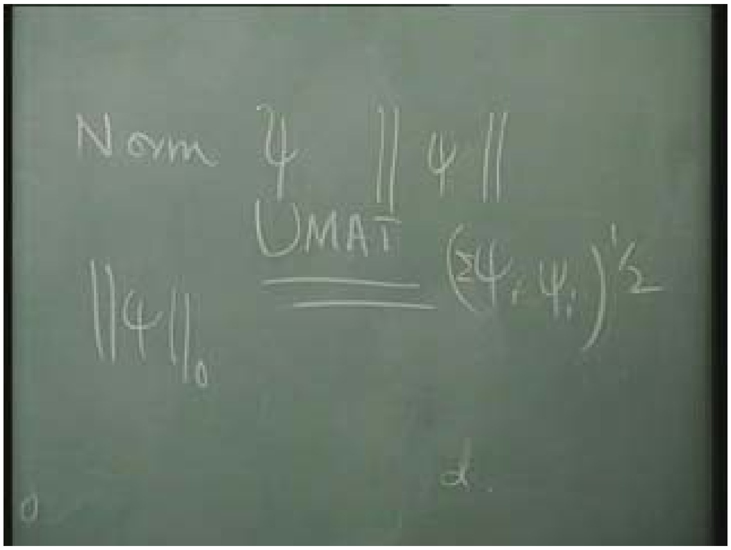
For example if you are doing an elastoplastic problem, this is called as D ep, elastoplastic. If you are doing a linear problem, you can still solve it as a non-linear problem, nothing wrong. That dow sigma by dow epsilon, what will that be? d; that is all, very good. So, it becomes just d and so B transpose db it will become and that is the known equation for us. In this case it becomes B transpose D ep B dV delta d. Is that clear, any question?

The two important issues in the whole of non-linear analysis, of course apart from convergence whatever I have taught from now on, one is the relationship between sigma and epsilon or in other words dow sigma by dow epsilon, what is it? In fact people call this with various names. One of the names by which people call this is Jacobian; they say material Jacobian or Jacobian, dow sigma by dow epsilon. I do not know how that name came about, but still people sometimes call this as Jacobian. In other words, that is the whole advantage of non-linear finite element and that is why packages like Abaqus has say a routine called UMAT.

The first issue is K_T formation. Please note that from one type of non-linearity, non-linear material behavior to another type of non-linear material behavior, the fellow who changes is this chap. Is that clear? He is the chap who changes. If you know how to calculate him, you can easily handle the material behavior. Is it clear? We will come back and look at an elastoplastic analysis later, but that is the first thing. There is one more issue, important issue. Before we go to that issue, what we do is we calculate tangent stiffness matrix using this. We calculate delta d each time, keep adding delta d, look at the error. Now, the error psi as I told you is a vector. So there is no one value. Here in this case, 1D case it was so beautiful, because there was only one value. You said that when that value goes to very small value, stop that program. That is what you said. But in this case, in this case the error term is not one value because it is a vector. For one element, there may be eight terms if it is a four nodded, say plane element. But if you look at the complete problem, then that is equal to the degrees of freedom for the problem. It is equal to the degrees of freedom for the problem. So, we should have a method of saying that convergence has been achieved.

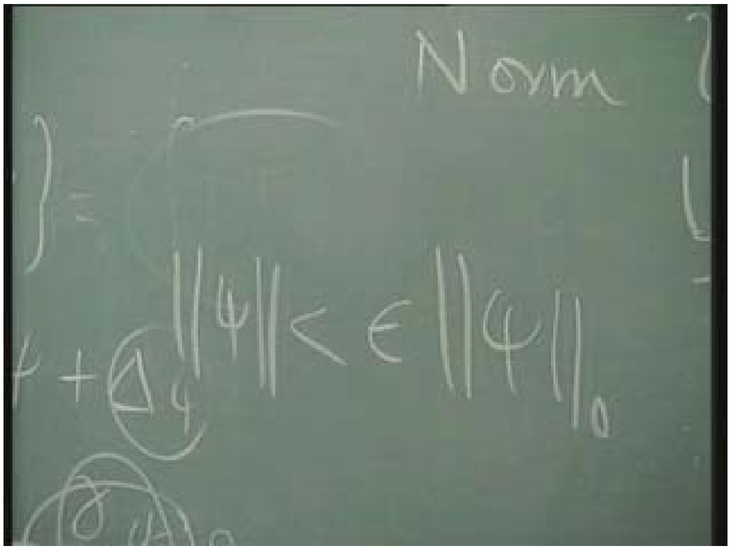
How do you do that? It is very simple.

(Refer Slide Time: 26:41)



You take what is called as the norm of psi, this vector. What is it? It is denoted like that; norm of psi is denoted by that two lines and that is nothing but root of psi i dot psi i. So root of that, that is sigma of psi i, of course you know that or sigma if you want to write, sigma of psi i or root of, that will give you the norm. What you do is, you compare, you calculate the norm in the first step or the first, rather the first iteration. Let me say that that norm is, let me say that that norm is zero. Very first step I calculate or very first, rather iteration I calculate. I keep on calculating this norm during every increment. I keep on calculating that and say that the convergence for this particular step has been achieved or first for this increment has been achieved when this norm multiplied by a tolerance value is less than the current norm.

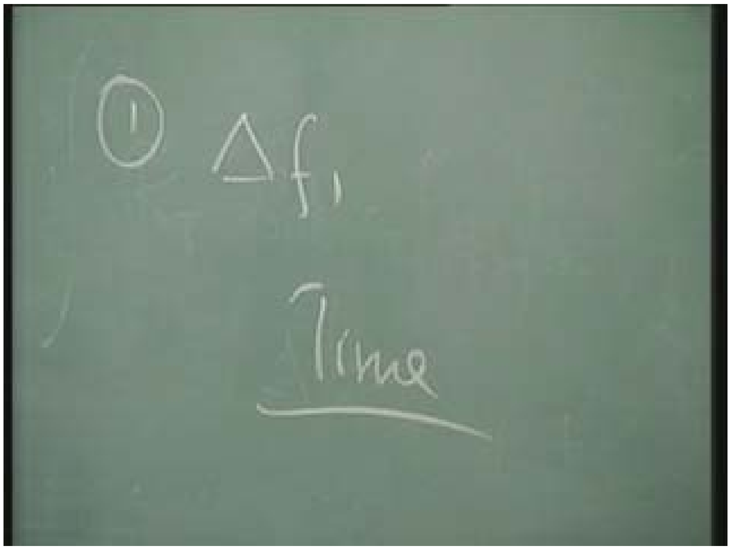
(Refer Slide Time: 28:45)



The current norm reduces to a value which is less than a tolerance times the initial first norm, then I say convergence has been achieved and this tolerance is usually 10 to the power of minus 6, because for Newton Raphson iteration 10 to the power of minus 6; it varies from problem to problem. If you want to have a tight convergence, then it is about 10 to the power of minus 6. If you want a loose convergence, you may get into problems, because the solution may drift, then may be 10 to the power of minus 3 is used. Usually, say in many packages, the default value is 10 to the power of minus 6 and you may not be or you do not usually change it unless you are an ace non-linear man or else you are not going to change …. that those things.

With this, let us summarize the steps. What are the steps? I will give you one minute to think about it and I will summarize all the steps. There I am going to hit upon another important issue. I am going to hit upon another important issue. Sorry, can you summarize? How do we start? We start the big loop, outside loop.

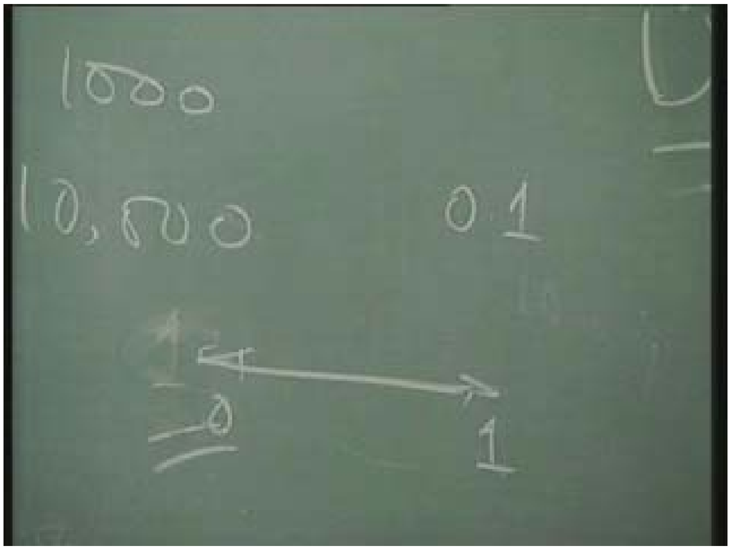
(Refer Slide Time: 30:33)



I start with the, what do I do? First step is what? Is to increment the load; so, I break the total load into a number of smaller loads, I said that. So, first I start with an increment of the total load or a fraction of the total load. I call this say as delta f_1. So, apply delta f_1. How do you now fraction this load? This becomes a head ache. See, if the total load is 1000, I may fraction. Say, for a problem say 1000 or 10,000 Newtons, I may be applying 200 as a first load. For a problem which may not withstand or for a load for a component which may not withstand that kind of load I may apply only 10. Say for example, for biological materials I apply some 10, in which case my first load may be 1 Newton. It is not very easy or there is no vehicle. There is no vehicle for you to talk to the program as to how much you have to apply the load.

There is no vehicle. You cannot say every time say especially when you are using a software like Ansys or Abaqus, you cannot, you cannot keep on telling that. Today non-linear analysis is available in lot more packages like Ideas, Ansys and all these packages, but you cannot keep on telling that first step you apply this much, second step more than that and so on, because you are also restricting certain things, I will come to that in a minute. But before that, I need a vehicle to tell the software as to how much the load has to be applied for each of the increment. So, what I do is to normalize the loads by 1. I will say that whatever be, even if it is 10,000, the load is normalized by 1. That means that apply 0.1 when I say, it is 0.1 of 10,000, if it is 200, 0.1 of 200 and so on. This normalized value I give a name called time. So the pseudo, it is a pseudo time. This time is actually not a time, but a pseudo time.

(Refer Slide Time: 33:15)



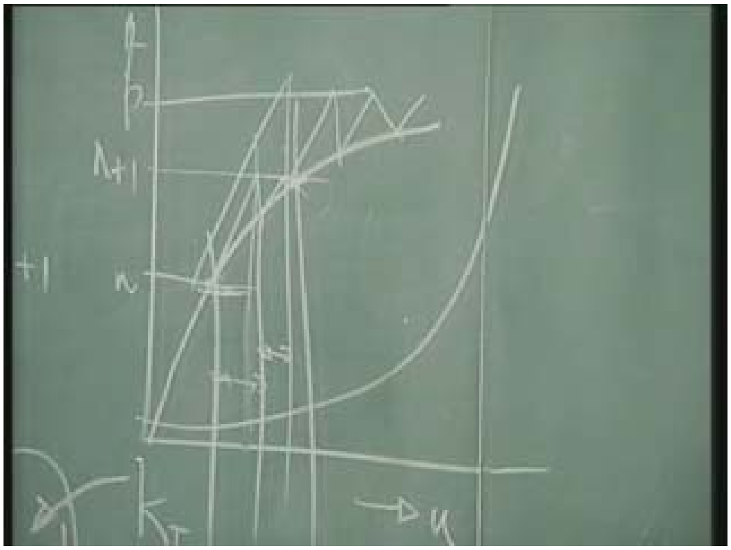
So, I say that I apply the total load of say 10,000 in one second. This has no meaning with an increment of 0.1 second. What do I mean? I mean that my increment of 0.1 second means that at the end of 0.1 or the first, say for example, first increment I apply 0.1 of 10,000. So, time is a vehicle or language to talk about the load application. Time has nothing to do with the actual problems; it is not a dynamic problem, it is a static problem. It is not a dynamic problem, but still we talk about time, basically because it is easier to say that and generalize it over a number of problems. So, if I say 1000 and say 0.1 second, 0.1 second is 0.1 of 1000. That is the load I applied. So, in both cases I will complete the load when I reach one second. Yeah, you do not understand?

Now what we do is, we normalize the application of load. Say for example, see for a problem there can be different amount of load. For a problem, there can be 1000, or say 10,000 or 20,000 or whatever it is. Now, I want to run a general purpose program. I cannot or I do not want to say for this problem apply first 100, later apply 200, 300 and so on and then for this problem, I do not want to say first apply 1000, then 2000, 3000 and so on. So, what I do is I normalize them in a scale of 1 and I call that scale as time. Note this carefully, I call that scale as time. So when I say time step, it actually means load step. In other words, in a time scale of 1 second, 0 to 1, 0 to 1, I apply for one problem 0 to 10,000; from here 0 to 10,000 and for another problem I apply from 0 to 5000, for another problem 0 to 1000 and so on, because I have normalized it to scale from 0 to 1.

What is the name I give to the scale? I call this scale as time scale. Is it clear? Why do I call like this, because it is anyway a static problem; slowly apply the load. There is a big difference between dynamic problem and static problem. Suppose I hit like this, I hit it, it is a dynamic problem. Whereas I just slowly apply a load here, it is a static problem. What we are doing here now is a static problem. So, I slowly apply the load in an incremental fashion and hence I give a pseudo time. Time has no meaning, just the scale is called the time. Now I say start with 0.1 second, 0.1. For every problem I can say start with 0.1. Then, for the first problem it will be 1000 if it is 10,000; for 5000 it will be 500 and so on. Is it clear? I will not talk about now the load step at all. I will talk about time step. The time step will automatically get converted into an appropriate load step depending upon this normalization.

Now, what is the advantage of this? The advantage is very simple. We have talked about convergence and it is very obvious looking at a figure like this, that if I take a very large step I would have difficulty of convergence. What does it mean? It means that when I move in smaller steps, when I move in smaller steps, then my linearization works better.

(Refer Slide Time: 38:02)

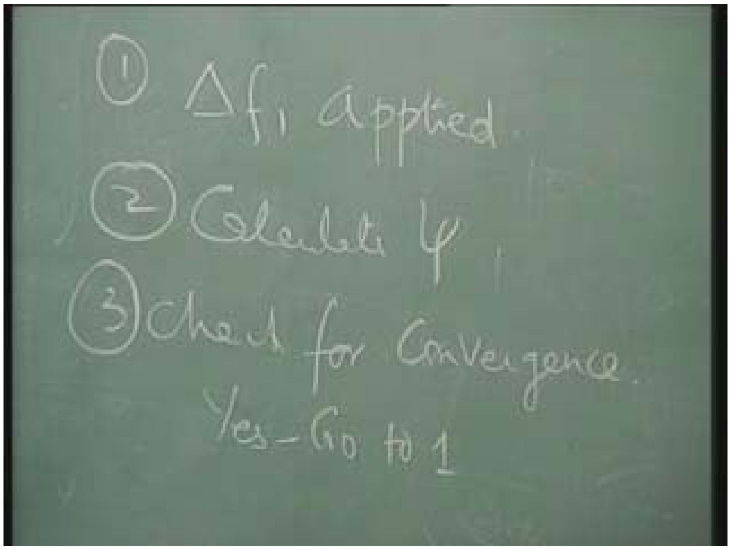


Suppose I take a big step from here to here, then I have to keep on working for a number of increments. I may or may not get convergence, there may be divergence also. Suppose the curve is like this, then there are problems of divergence. What does it mean? It means that I may not converge. Mathematically lot of issues, but let me say that the time step is important for or load step is important for, both of them are synonymous; load step or time step both of them are synonymous, is important for convergence.

Now, that does not mean I can keep on applying naught naught naught 1 all the time. Because, I will take then 1 by naught naught naught 1 to reach 1; that is 10,000 times I will have to apply the load. What do I do or what does all practical softwares do? What they do is, they start with say naught naught naught 1 initially, go and converge, look at how it is converged; converge nicely, within 3 iteration you have converged, then it will double it or in other words most softwares use variable steps. It will start with naught naught naught 1, it will keep on varying the time step. It will go to say point naught naught 5, naught naught 5 convergence is very good, then it will go to naught 1, naught 1 convergence is very good, it will go to 0.1 and so on.

In other words, this helps in optimizing. This helps in optimizing the time required to solve the problem. Is it clear? It helps in optimizing the time required for solving the problem and hence we have what is called as a time step. Is it clear? We have what is called as a time step here. If it is load, it is difficult to handle; for a software to handle. Best thing is to say that, next step I am going to apply naught naught 1, next step I will apply naught naught 2 and so on; it is easy to say that. So, load step and time step are synonymous. Please note that. So, first step I start with a particular time step. So, I say that for that corresponding time step, I have say delta f_1; I apply the load delta f_1 applied.

(Refer Slide Time: 40:54)



① Δf_1 applied.

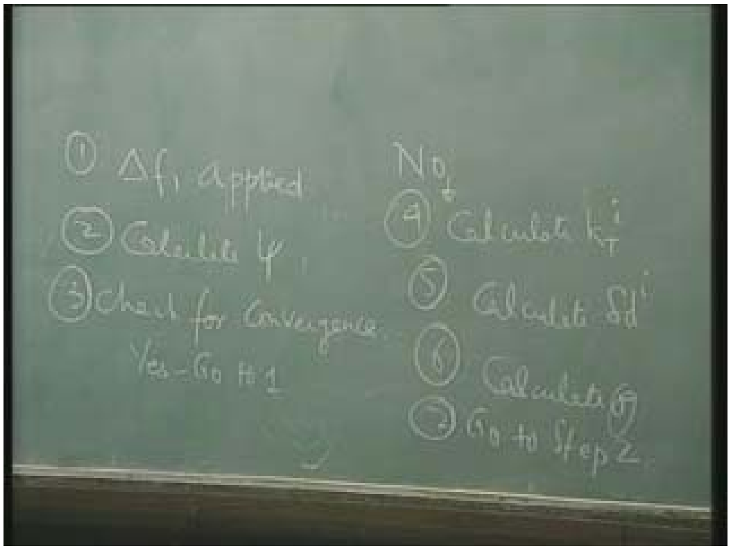
② Calculate ψ

③ Check for Convergence.

Yes – Go to 1

That is what I do. What is my next step? Delta f_1 for a multi axial situation may be a vector, because in a problem you can have loads at different places. For uni axial problem, it can be one particular load. Is it clear? So, what is my next step? Calculating, yeah, but you are jumping. So next step, calculate the error. Since we have started, now the error will be high. So, calculate psi. That is my next step, very good. So, what is my third step? Check for convergence. That is what you said. I will change it later a bit, but nevertheless your steps are right; check for the convergence. Check for convergence that is the third step. That is the third step, check for convergence. If convergence has been achieved, yes go to one; go to one. Is it clear? If no, we will put the correct one later in the next class, but let us, because your understanding will go up if you start thinking.

(Refer Slide Time: 43:19)



① Δf_i applied

② Calculate ψ,

③ Check for Convergence.

Yes – Go to 1

No

④ Calculate K_T^i

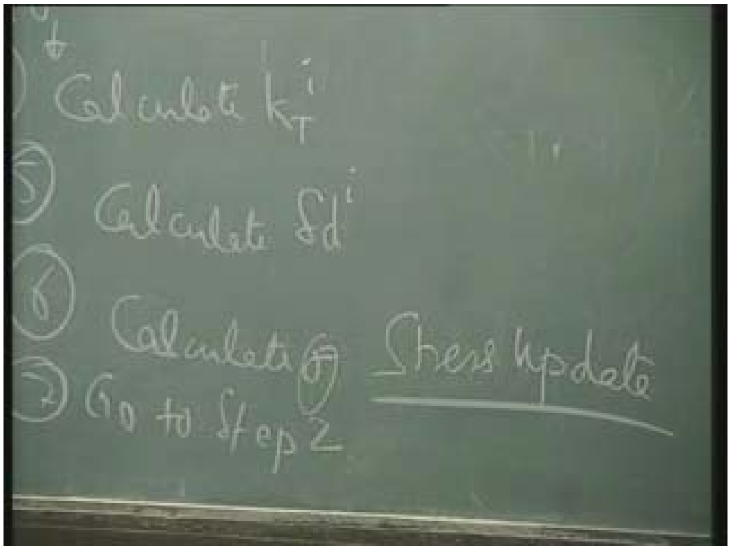
⑤ Calculate δd^i

⑥ Calculate σ

⑦ Go to Step 2

If no, if no then, you come to the step into the iteration load, step four. No means you go into the step four. What is it that you do here? Iteration; fine. What is that you do here as a first step? Calculate K_T; very good. So, calculate K_T. What is it that you do afterwards? Calculate delta d, delta d is also a vector. Now, I have to put, I have to have a count. So, I put an iterative count. Then what do I do? What is the next step? Six; no, I will calculate strain and stress and so on, because B transpose sigma is there, that is p. p minus f B transpose sigma dV minus f. So, calculate sigma, then go to step 2. That is why I said you can reorganize it in a better fashion. Go to step 2, go to step 2; calculate error, check for convergence. If yes go back to 1, if no come back again, calculate this convergence, no convergence get to do all these.

(Refer Slide Time: 45:43)



Calculate k_T^i

(5) Calculate δd^i

(6) Calculate σ Stress Update

(7) Go to Step 2

In fact, sigma calculation is a very, very important step and is called as stress update step. There has been lot of research on stress update algorithms. So, that is the next step that we have to concentrate on. Is it clear? Please put this in a better fashion, in the sense that it is a much more appropriate fashion here. These are the steps, no doubt about it. So that we will discuss, the next step of stress update in the next class. But, there are certain issues here which I want you to keep in mind and think about it, so that we will not loose track of what we are doing. For example one of the things is that the stress should be properly updated, so the constitutive equation is correctly maintained and so on. We will see about the stress update algorithm in the next class.