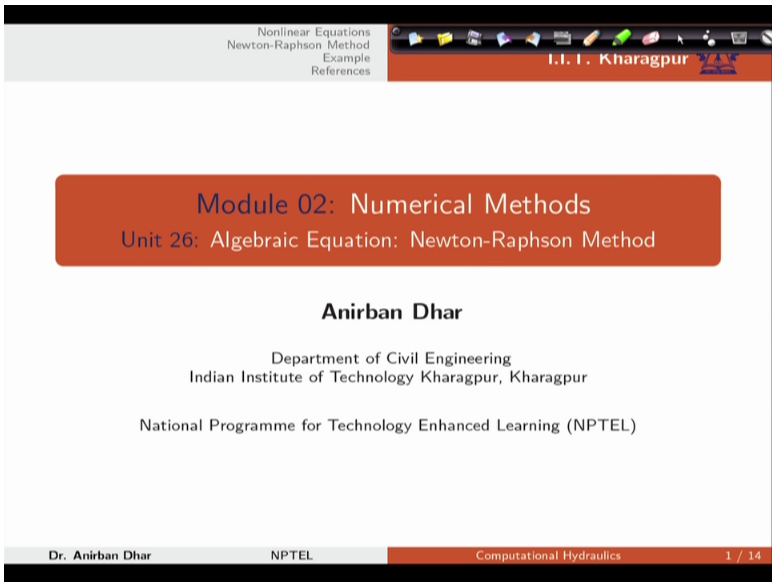
Computational Hydraulics
Professor Anirban Dhar
Department of Civil Engineering
Indian Institute of Technology Kharagpur
Lecture 30
Algebraic Equation: Newton - Raphson Method

Welcome to this lecture number 30 of course computational hydraulics. We are in module 2, numerical methods. And this is unit number 26, algebraic equation, Newton Raphson method.

(Refer Slide Time: 00:35)



Nonlinear Equations
Newton-Raphson Method
Example
References

I. I. T. Kharagpur

Module 02: Numerical Methods
Unit 26: Algebraic Equation: Newton-Raphson Method

Anirban Dhar

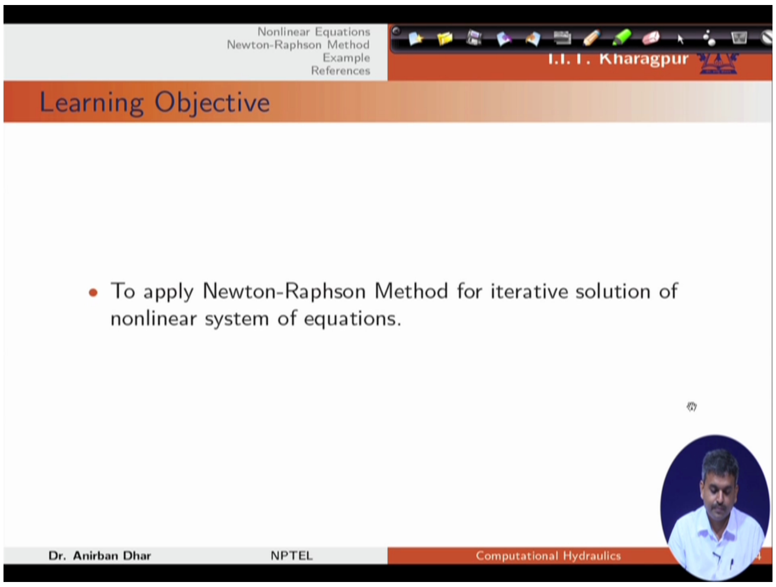
Department of Civil Engineering
Indian Institute of Technology Kharagpur, Kharagpur

National Programme for Technology Enhanced Learning (NPTEL)

Dr. Anirban Dhar NPTEL Computational Hydraulics 1 / 14

Learning objective. At the end this particular unit students will be able to apply Newton Raphson method for iterative solution of nonlinear system of equations.

(Refer Slide Time: 00:53)


Nonlinear Equations
Newton-Raphson Method
Example
References

I. I. T. Kharagpur

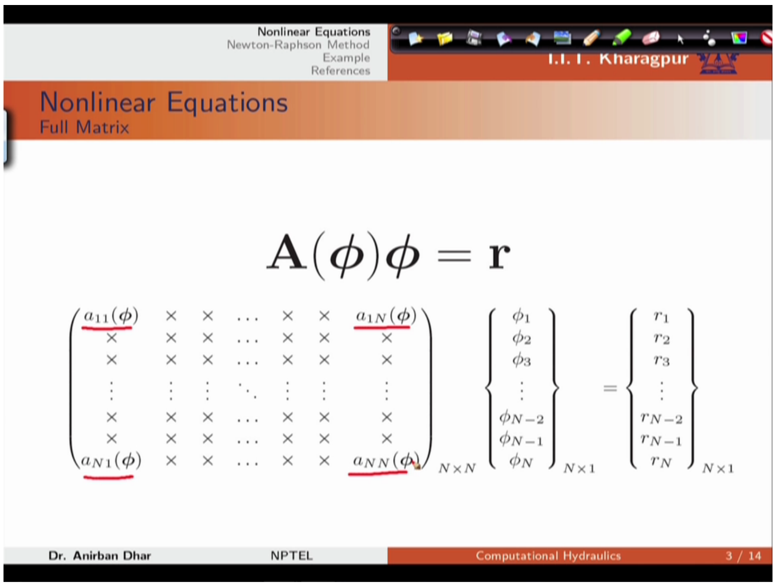
Learning Objective

- To apply Newton-Raphson Method for iterative solution of nonlinear system of equations.

Dr. Anirban Dhar NPTEL Computational Hydraulics

Nonlinear equations. We have seen that if we have A phi equals to r kind of structure then we can solve it using linear direct or iterative approach. But in this case the requirement is that A should be with constant coefficients. But if this matrix A or its coefficients functions phi then the situation will be different. So in this case we can write the final form like this where we have a11 is function of phi, a1N, aN1, aNN is function of phi.

(Refer Slide Time: 02:05)


Nonlinear Equations
Newton-Raphson Method
Example
References

I. I. T. Kharagpur

Nonlinear Equations
Full Matrix

$$\mathbf{A}(\phi)\phi = \mathbf{r}$$

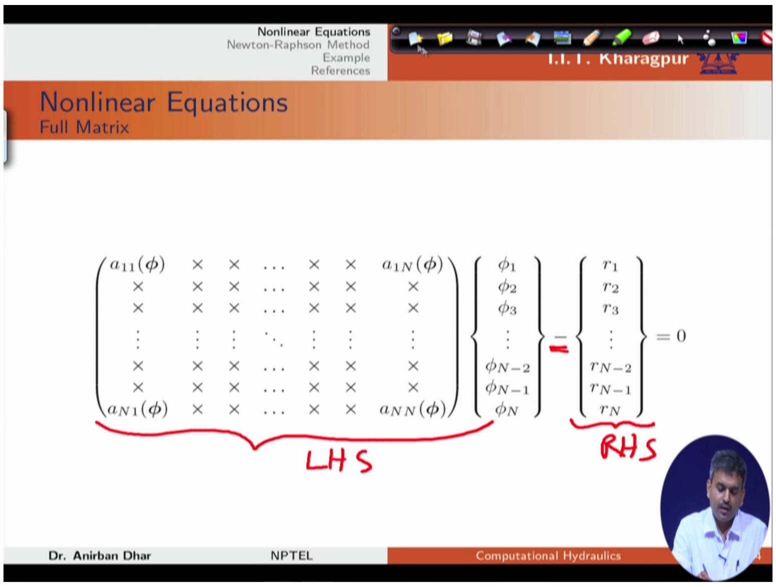
$$\begin{pmatrix} \underline{a_{11}(\phi)} & \times & \times & \cdots & \times & \times & \underline{a_{1N}(\phi)} \\ \times & \times & \times & \cdots & \times & \times & \times \\ \times & \times & \times & \cdots & \times & \times & \times \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \times & \times & \times & \cdots & \times & \times & \times \\ \times & \times & \times & \cdots & \times & \times & \times \\ \underline{a_{N1}(\phi)} & \times & \times & \cdots & \times & \times & \underline{a_{NN}(\phi)} \end{pmatrix}_{N\times N} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_{N-2} \\ \phi_{N-1} \\ \phi_N \end{Bmatrix}_{N\times 1} = \begin{Bmatrix} r_1 \\ r_2 \\ r_3 \\ \vdots \\ r_{N-2} \\ r_{N-1} \\ r_N \end{Bmatrix}_{N\times 1}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics 3 / 14

Now you may get this kind of structure during your discretization process. Now how to solve this kind of problem? In lecture number 24 I have talked about two approaches one is based on Taylor series another one is based on optimisation. So Newton Raphson approach is necessarily based on Taylor series expansion approach.

Now in this case we can write again the full matrix form and we can transfer this right hand side vector on the left hand side. This is left hand side portion and this is right hand side portion. I can just transfer it with the negative sign.

(Refer Slide Time: 03:16)



Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

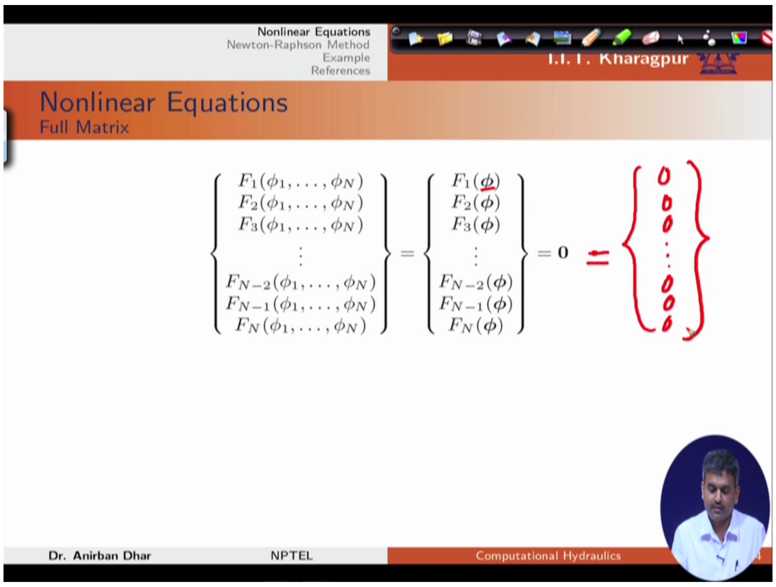
Nonlinear Equations
Full Matrix

$$\underbrace{\begin{pmatrix} a_{11}(\phi) & \times & \times & \dots & \times & \times & a_{1N}(\phi) \\ \times & \times & \times & \dots & \times & \times & \times \\ \times & \times & \times & \dots & \times & \times & \times \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \times & \times & \times & \dots & \times & \times & \times \\ \times & \times & \times & \dots & \times & \times & \times \\ a_{N1}(\phi) & \times & \times & \dots & \times & \times & a_{NN}(\phi) \end{pmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_{N-2} \\ \phi_{N-1} \\ \phi_N \end{Bmatrix}}_{LHS} - \underbrace{\begin{Bmatrix} r_1 \\ r_2 \\ r_3 \\ \vdots \\ r_{N-2} \\ r_{N-1} \\ r_N \end{Bmatrix}}_{RHS} = 0$$

Dr. Anirban Dhar | NPTEL | Computational Hydraulics

Now if I write that form with this F1, F2, F3, up to FN. Remember that in this case we are representing each row with different nonlinear functions. F1 means this represents the first row, F2 it is a second row, F3 third row like that, FN is nth row. In compact form or with phi vector I can write this as F phi. This is equal to zero. This is again vector. Obviously individual terms will be zero in this case.

(Refer Slide Time: 04:16)



Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

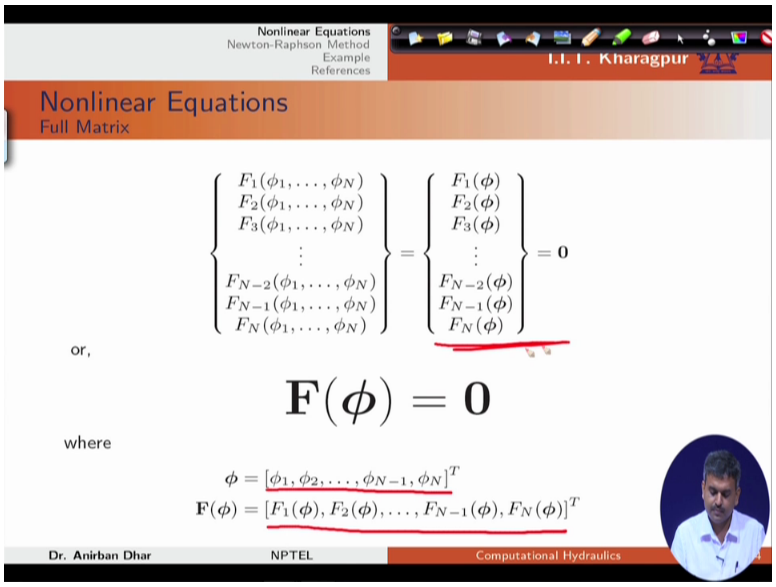
Nonlinear Equations
Full Matrix

$$\begin{Bmatrix} F_1(\phi_1,\dots,\phi_N) \\ F_2(\phi_1,\dots,\phi_N) \\ F_3(\phi_1,\dots,\phi_N) \\ \vdots \\ F_{N-2}(\phi_1,\dots,\phi_N) \\ F_{N-1}(\phi_1,\dots,\phi_N) \\ F_N(\phi_1,\dots,\phi_N) \end{Bmatrix} = \begin{Bmatrix} F_1(\underline{\phi}) \\ F_2(\phi) \\ F_3(\phi) \\ \vdots \\ F_{N-2}(\phi) \\ F_{N-1}(\phi) \\ F_N(\phi) \end{Bmatrix} = 0 = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{Bmatrix}$$

Dr. Anirban Dhar | NPTEL | Computational Hydraulics

So how to solve this particular problem? In final form we can write this as F is again vector. Phi is vector. Zero, this is again in vector form. This will be column vector or zero values. So we have to solve this nonlinear system of equations. And phi, this vector is represented in terms of column vector. I have used transpose. Similarly F is also column vector like this one.

(Refer Slide Time: 05:08)


Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Nonlinear Equations
Full Matrix

$$\begin{Bmatrix} F_1(\phi_1,\dots,\phi_N) \\ F_2(\phi_1,\dots,\phi_N) \\ F_3(\phi_1,\dots,\phi_N) \\ \vdots \\ F_{N-2}(\phi_1,\dots,\phi_N) \\ F_{N-1}(\phi_1,\dots,\phi_N) \\ F_N(\phi_1,\dots,\phi_N) \end{Bmatrix} = \begin{Bmatrix} F_1(\phi) \\ F_2(\phi) \\ F_3(\phi) \\ \vdots \\ F_{N-2}(\phi) \\ F_{N-1}(\phi) \\ F_N(\phi) \end{Bmatrix} = \mathbf{0}$$

or,

$$\mathbf{F}(\phi) = \mathbf{0}$$

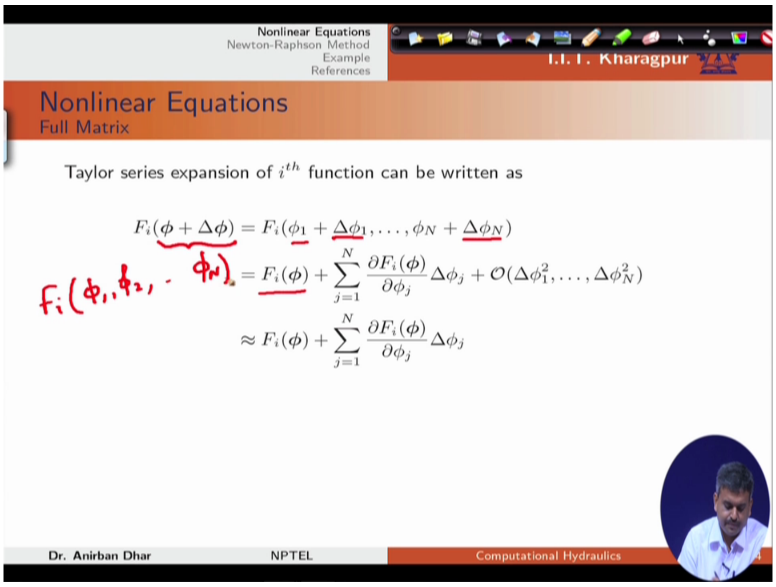
where

$$\phi = \underline{[\phi_1, \phi_2, \dots, \phi_{N-1}, \phi_N]^T}$$
$$\mathbf{F}(\phi) = \underline{[F_1(\phi), F_2(\phi), \dots, F_{N-1}(\phi), F_N(\phi)]^T}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

Now let us see how to use our Taylor series expansion of ith function? Let us say Fi is any arbitrary function. So to represent that if we have F phi plus del phi, phi del phi, both are vectors. Now in this case if we give increment del phi 1 to phi 1 and del phi N to phi N we can write the equation or this expression in terms of Fi phi. Fi phi is necessarily phi 1, phi2 up to phi N.

(Refer Slide Time: 06:07)


Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Nonlinear Equations
Full Matrix

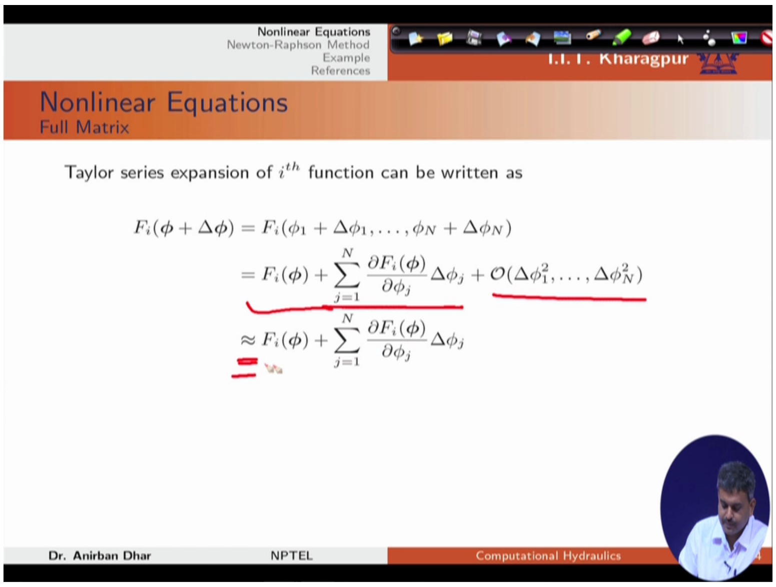
Taylor series expansion of i^{th} function can be written as

$$F_i(\underline{\phi + \Delta\phi}) = F_i(\underline{\phi_1} + \underline{\Delta\phi_1}, \dots, \phi_N + \underline{\Delta\phi_N})$$
$$F_i(\phi_1, \phi_2, \cdots \phi_N) = \underline{F_i(\phi)} + \sum_{j=1}^{N} \frac{\partial F_i(\phi)}{\partial \phi_j}\Delta\phi_j + \mathcal{O}(\Delta\phi_1^2, \dots, \Delta\phi_N^2)$$
$$\approx F_i(\phi) + \sum_{j=1}^{N} \frac{\partial F_i(\phi)}{\partial \phi_j}\Delta\phi_j$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

Now in this case if we expand it up to first order, we will get first order derivative and higher order although terms will be there. So in this case we are taking up to first order starting from function value. And we are neglecting that is why we are using this approximate sign here.

(Refer Slide Time: 06:44)



Nonlinear Equations
Full Matrix

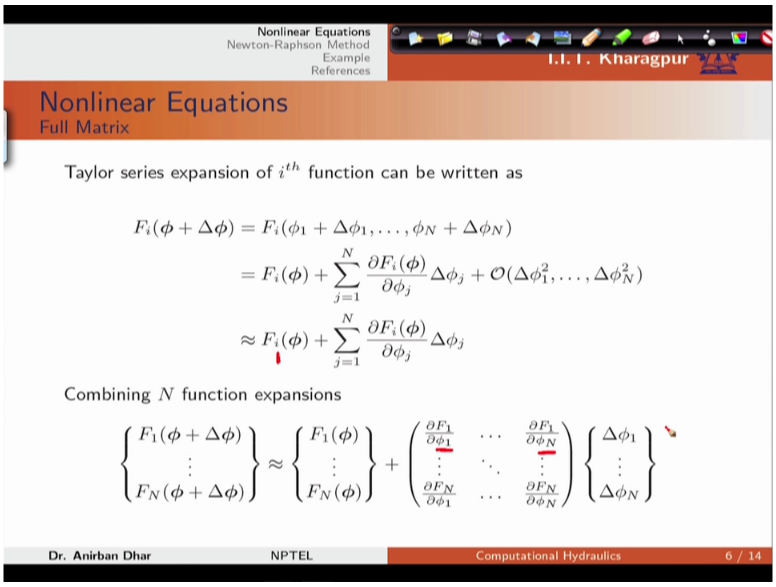
Taylor series expansion of i^{th} function can be written as

$$F_i(\phi + \Delta\phi) = F_i(\phi_1 + \Delta\phi_1, \dots, \phi_N + \Delta\phi_N)$$
$$= \underbrace{F_i(\phi) + \sum_{j=1}^{N} \frac{\partial F_i(\phi)}{\partial \phi_j}\Delta\phi_j}_{} + \underbrace{\mathcal{O}(\Delta\phi_1^2, \dots, \Delta\phi_N^2)}_{}$$
$$\approx F_i(\phi) + \sum_{j=1}^{N} \frac{\partial F_i(\phi)}{\partial \phi_j}\Delta\phi_j$$

So J starting from 1 to N, to consider all the variables. Now in this case combining N function expansions. Let us say that we have used it for ith function. Let us write it for first one, nth one. And in between for all the functions if we write, we can approximate it with F1. This is corresponding to this vector or function vector at phi plus this F1 is represented or this is differentiated with respect to all the variables starting from phi 1 to phi N.

And we can multiply this column vector del phi 1 to the del phi N with this one. So this is a matrix representation of our Taylor series.

(Refer Slide Time: 07:57)



Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Nonlinear Equations
Full Matrix

Taylor series expansion of i^{th} function can be written as

$$\begin{aligned} F_i(\phi+\Delta\phi) &= F_i(\phi_1+\Delta\phi_1,\ldots,\phi_N+\Delta\phi_N) \\ &= F_i(\phi)+\sum_{j=1}^{N}\frac{\partial F_i(\phi)}{\partial\phi_j}\Delta\phi_j+\mathcal{O}(\Delta\phi_1^2,\ldots,\Delta\phi_N^2) \\ &\approx F_i(\phi)+\sum_{j=1}^{N}\frac{\partial F_i(\phi)}{\partial\phi_j}\Delta\phi_j \end{aligned}$$

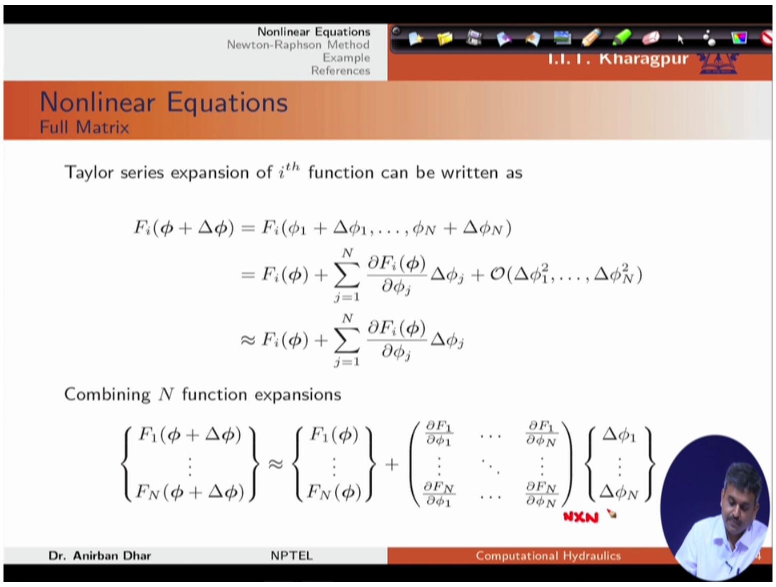
Combining N function expansions

$$\begin{Bmatrix} F_1(\phi+\Delta\phi) \\ \vdots \\ F_N(\phi+\Delta\phi) \end{Bmatrix} \approx \begin{Bmatrix} F_1(\phi) \\ \vdots \\ F_N(\phi) \end{Bmatrix} + \begin{pmatrix} \frac{\partial F_1}{\partial\phi_1} & \cdots & \frac{\partial F_1}{\partial\phi_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_N}{\partial\phi_1} & \cdots & \frac{\partial F_N}{\partial\phi_N} \end{pmatrix} \begin{Bmatrix} \Delta\phi_1 \\ \vdots \\ \Delta\phi_N \end{Bmatrix}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics 6 / 14

Now from this one we can get some interesting results. Obviously in this case we have 1 to N. So N cross N is the size of this Jacobian matrix. This is called as Jacobian matrix containing the first order derivative or the functions with respect to the variables.

(Refer Slide Time: 08:30)



Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Nonlinear Equations
Full Matrix

Taylor series expansion of i^{th} function can be written as

$$\begin{aligned} F_i(\phi+\Delta\phi) &= F_i(\phi_1+\Delta\phi_1,\ldots,\phi_N+\Delta\phi_N) \\ &= F_i(\phi)+\sum_{j=1}^{N}\frac{\partial F_i(\phi)}{\partial\phi_j}\Delta\phi_j+\mathcal{O}(\Delta\phi_1^2,\ldots,\Delta\phi_N^2) \\ &\approx F_i(\phi)+\sum_{j=1}^{N}\frac{\partial F_i(\phi)}{\partial\phi_j}\Delta\phi_j \end{aligned}$$

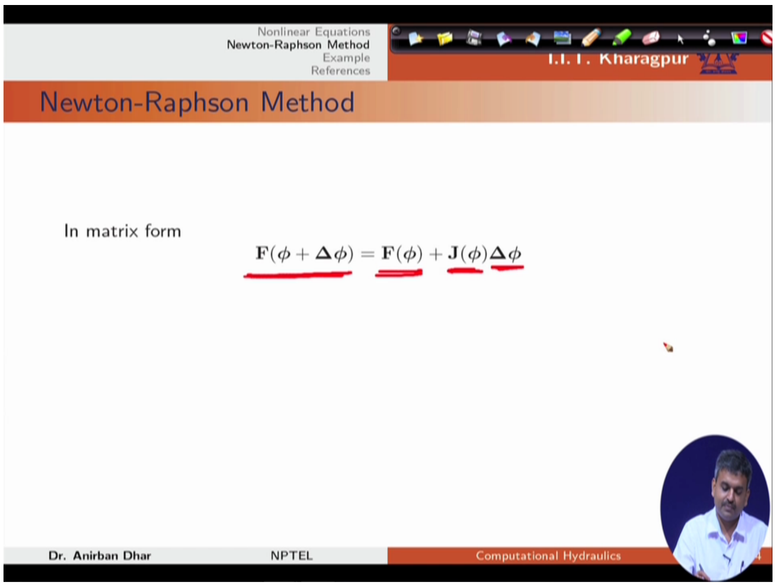
Combining N function expansions

$$\begin{Bmatrix} F_1(\phi+\Delta\phi) \\ \vdots \\ F_N(\phi+\Delta\phi) \end{Bmatrix} \approx \begin{Bmatrix} F_1(\phi) \\ \vdots \\ F_N(\phi) \end{Bmatrix} + \underbrace{\begin{pmatrix} \frac{\partial F_1}{\partial\phi_1} & \cdots & \frac{\partial F_1}{\partial\phi_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_N}{\partial\phi_1} & \cdots & \frac{\partial F_N}{\partial\phi_N} \end{pmatrix}}_{N\times N} \begin{Bmatrix} \Delta\phi_1 \\ \vdots \\ \Delta\phi_N \end{Bmatrix}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

Now if we utilise this for our calculation, in the next step we can compactly write it in vector form. So left hand side we have represented with vector with incremental variable value and vector with phi calculated at that point. This is the Jacobian matrix. And again this is the column vector for increment, whatever we have considered in our previous lecture.

(Refer Slide Time: 09:11)



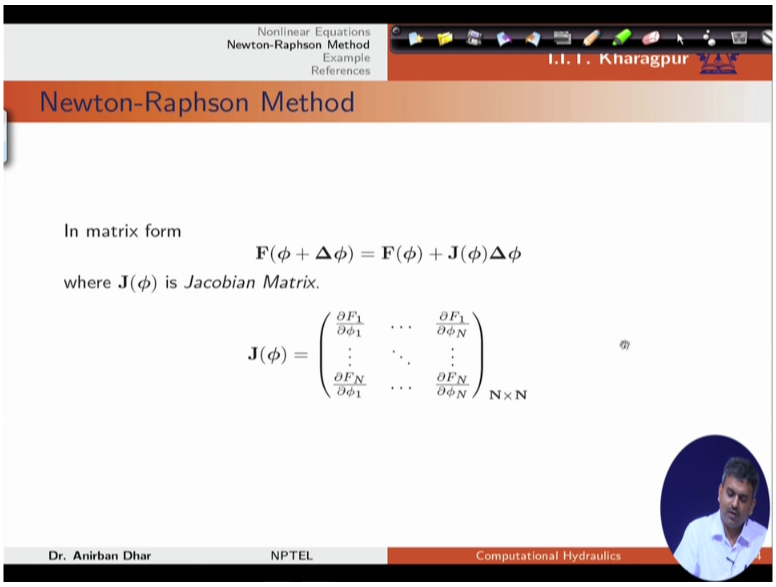
Newton-Raphson Method

In matrix form

$$\underline{\mathbf{F}(\phi + \Delta\phi)} = \underline{\mathbf{F}(\phi)} + \underline{\mathbf{J}(\phi)}\underline{\Delta\phi}$$

Now at this point we can say that this particular J is the Jacobian matrix which is of the size N cross N. So in this case our assumption is that we have N number of equations and we have N variables available. For N variables we have N number of equations available for us.

(Refer Slide Time: 09:48)



Newton-Raphson Method

In matrix form

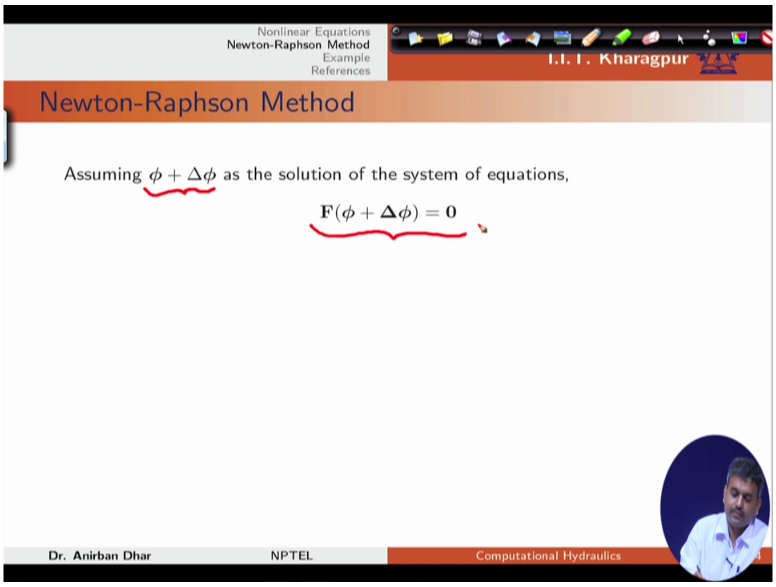
$$\mathbf{F}(\phi + \Delta\phi) = \mathbf{F}(\phi) + \mathbf{J}(\phi)\Delta\phi$$

where $\mathbf{J}(\phi)$ is *Jacobian Matrix*.

$$\mathbf{J}(\phi) = \begin{pmatrix} \frac{\partial F_1}{\partial \phi_1} & \cdots & \frac{\partial F_1}{\partial \phi_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_N}{\partial \phi_1} & \cdots & \frac{\partial F_N}{\partial \phi_N} \end{pmatrix}_{\mathbf{N}\times\mathbf{N}}$$

Now assuming that, when we are giving increment to this phi, this will lead us to the solution of the system. So we can assume that F phi plus del phi, this is equals to zero.

(Refer Slide Time: 10:16)


Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Newton-Raphson Method

Assuming $\underbrace{\phi + \Delta\phi}$ as the solution of the system of equations,

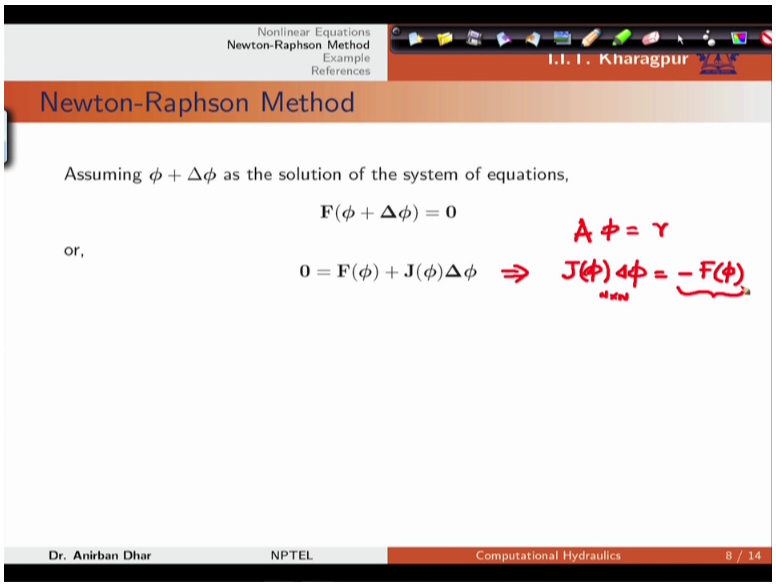
$$\underbrace{\mathbf{F}(\phi + \Delta\phi) = \mathbf{0}}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

Now this is nothing but our previous equation. We have used zero in place of F phi plus del phi. Now in this case we can write it like this. So this is J phi del phi equals to minus F phi. Interestingly this is linear in nature. Why? Because Jacobian matrix that we are calculating, that is evaluated at the point phi. So J is evaluated at point phi and its size is N cross N.

So we can say that it is equivalent to our original system which is A or A phi equals to r where J is the coefficient matrix and in place of phi we are writing this del phi. So, on the right hand side in place of r we are writing this minus F. That means function values evaluated at this phi point.

(Refer Slide Time: 11:55)


Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Newton-Raphson Method

Assuming $\phi + \Delta\phi$ as the solution of the system of equations,

$$\mathbf{F}(\phi + \Delta\phi) = \mathbf{0}$$

$$A\phi = r$$

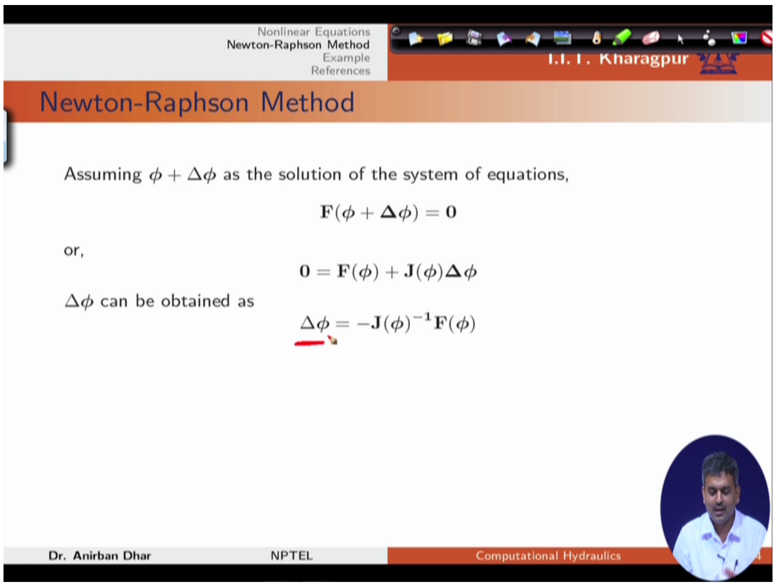
or,

$$\mathbf{0} = \mathbf{F}(\phi) + \mathbf{J}(\phi)\Delta\phi \Rightarrow \underset{N\times N}{J(\phi)}\Delta\phi = \underbrace{-F(\phi)}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics 8 / 14

Now if we write the next step, this is nothing but we can invite this J and we can get the increment value for a particular point to get the ultimate solution. Now for a particular point phi, if we apply this del phi increment, we should get the solution. So we can apply this increment.

(Refer Slide Time: 12:32)



Nonlinear Equations
Newton-Raphson Method
Example
References

I. I. T. Kharagpur

Newton-Raphson Method

Assuming $\phi + \Delta\phi$ as the solution of the system of equations,

$$\mathbf{F}(\phi + \boldsymbol{\Delta}\phi) = \mathbf{0}$$

or,

$$\mathbf{0} = \mathbf{F}(\phi) + \mathbf{J}(\phi)\boldsymbol{\Delta}\phi$$

$\Delta\phi$ can be obtained as

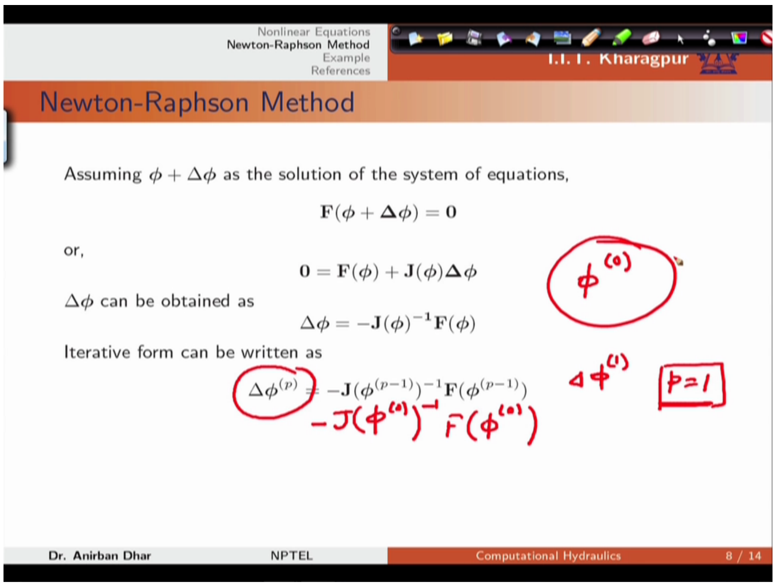
$$\Delta\phi = -\mathbf{J}(\phi)^{-1}\mathbf{F}(\phi)$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

So if we apply this increment and we can write it in iterative form. Let us say that we are starting from phi 0 point. And for pth or first one, that means at first iteration we need to calculate this del phi 1. So if p equals to 1, so obviously this J matrix should be evaluated at phi 0. At first iteration we should take inverse this F phi 0.

That means we can start from our guess value like the iterative technique that we have utilised for constant coefficient case. Now we can get information about this del phi vector at pth iteration where p equals to 1. We are starting from this guess value which is phi 0.

(Refer Slide Time: 13:54)



Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Newton-Raphson Method

Assuming $\phi + \Delta\phi$ as the solution of the system of equations,

$$\mathbf{F}(\phi + \boldsymbol{\Delta}\phi) = \mathbf{0}$$

or,

$$\mathbf{0} = \mathbf{F}(\phi) + \mathbf{J}(\phi)\boldsymbol{\Delta}\phi$$

$$\boxed{\phi^{(0)}}$$

$\Delta\phi$ can be obtained as

$$\Delta\phi = -\mathbf{J}(\phi)^{-1}\mathbf{F}(\phi)$$

Iterative form can be written as

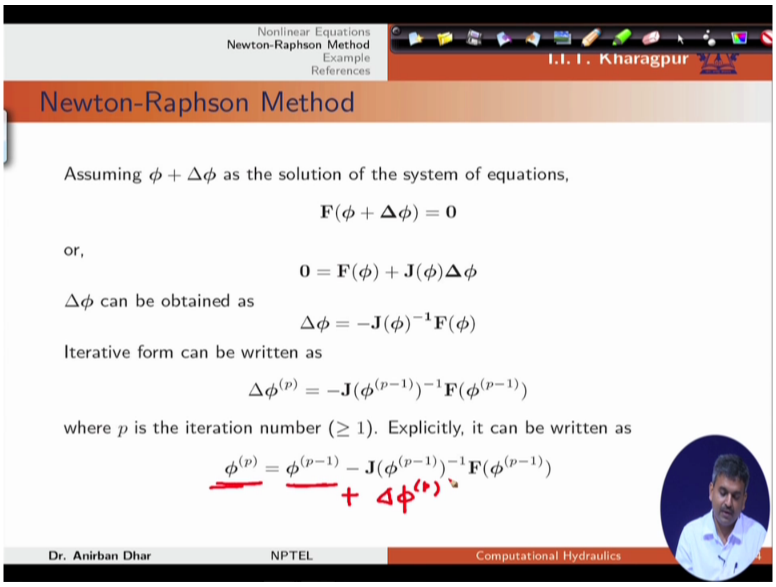
$$\boxed{\Delta\phi^{(p)}} = -\mathbf{J}(\phi^{(p-1)})^{-1}\mathbf{F}(\phi^{(p-1)}) \qquad \Delta\phi^{(1)} \qquad \boxed{p=1}$$

$$-J(\phi^{(0)})^{-1} F(\phi^{(0)})$$

Dr. Anirban Dhar NPTEL Computational Hydraulics 8 / 14

Now this particular step is valid for p greater than equals to 1. That means starting from 1, we can start the iteration and we can add the increment with our original or starting point. So explicitly we can write that, this phi p or updated value equals to my old value. And plus del phi p. Now del phi p is again minus J inverse F phi. So we are directly adding this with the old value. You can compare this step with our iterative technique used for linear equations. This is similar. We are adding or updating the value at a particular point.

(Refer Slide Time: 15:12)



Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Newton-Raphson Method

Assuming $\phi + \Delta\phi$ as the solution of the system of equations,

$$\mathbf{F}(\phi + \boldsymbol{\Delta}\phi) = \mathbf{0}$$

or,

$$\mathbf{0} = \mathbf{F}(\phi) + \mathbf{J}(\phi)\boldsymbol{\Delta}\phi$$

$\Delta\phi$ can be obtained as

$$\Delta\phi = -\mathbf{J}(\phi)^{-1}\mathbf{F}(\phi)$$

Iterative form can be written as

$$\Delta\phi^{(p)} = -\mathbf{J}(\phi^{(p-1)})^{-1}\mathbf{F}(\phi^{(p-1)})$$

where p is the iteration number (≥ 1). Explicitly, it can be written as

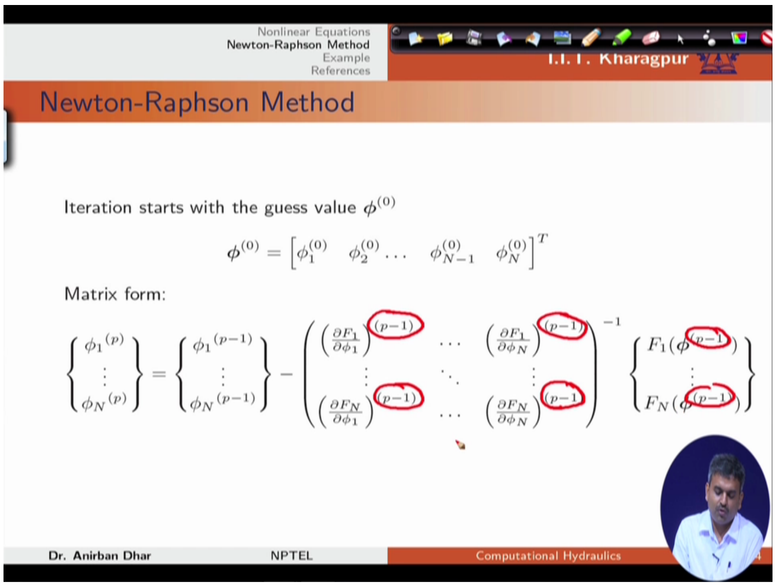
$$\underline{\phi^{(p)}} = \underline{\phi^{(p-1)}} - \mathbf{J}(\phi^{(p-1)})^{-1}\mathbf{F}(\phi^{(p-1)})$$

$$+ \Delta\phi^{(p)}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

So in this case this iteration starts with a guess value which is phi 1 0, phi 2 0, phi N minus 1 0, phi N 0. Now in matrix form essentially we are doing this and this part is calculated at present information level. So present information level means that whatever value is available we are calculating these derivatives, all these functions values with that information.

(Refer Slide Time: 16:04)


Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Newton-Raphson Method

Iteration starts with the guess value $\phi^{(0)}$

$$\phi^{(0)} = \left[\phi_1^{(0)} \quad \phi_2^{(0)} \ldots \quad \phi_{N-1}^{(0)} \quad \phi_N^{(0)}\right]^T$$

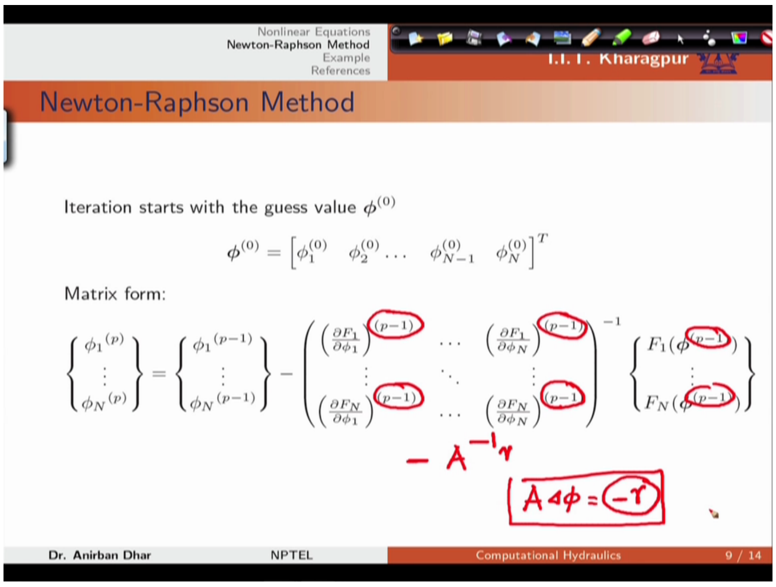
Matrix form:

$$\begin{Bmatrix} \phi_1{}^{(p)} \\ \vdots \\ \phi_N{}^{(p)} \end{Bmatrix} = \begin{Bmatrix} \phi_1{}^{(p-1)} \\ \vdots \\ \phi_N{}^{(p-1)} \end{Bmatrix} - \begin{pmatrix} \left(\frac{\partial F_1}{\partial \phi_1}\right)^{(p-1)} & \cdots & \left(\frac{\partial F_1}{\partial \phi_N}\right)^{(p-1)} \\ \vdots & \ddots & \vdots \\ \left(\frac{\partial F_N}{\partial \phi_1}\right)^{(p-1)} & \cdots & \left(\frac{\partial F_N}{\partial \phi_N}\right)^{(p-1)} \end{pmatrix}^{-1} \begin{Bmatrix} F_1(\phi^{(p-1)}) \\ \vdots \\ F_N(\phi^{(p-1)}) \end{Bmatrix}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

That means your constant coefficient thing, this is necessarily A inverse and right hand side this is r and one negative sign is there. We can utilise this or we can directly solve this A del phi equals to minus r kind of thing. So if we provide this minus r where r vector is your F vector. So we can utilise our linear, either direct or iterative approach to get the solution or this del phi vector. So we can utilise maybe LU decomposition in this case to get the solution.

(Refer Slide Time: 17:02)


Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Newton-Raphson Method

Iteration starts with the guess value $\phi^{(0)}$

$$\phi^{(0)} = \left[\phi_1^{(0)} \quad \phi_2^{(0)} \ldots \quad \phi_{N-1}^{(0)} \quad \phi_N^{(0)}\right]^T$$

Matrix form:

$$\begin{Bmatrix} \phi_1{}^{(p)} \\ \vdots \\ \phi_N{}^{(p)} \end{Bmatrix} = \begin{Bmatrix} \phi_1{}^{(p-1)} \\ \vdots \\ \phi_N{}^{(p-1)} \end{Bmatrix} - \begin{pmatrix} \left(\frac{\partial F_1}{\partial \phi_1}\right)^{(p-1)} & \cdots & \left(\frac{\partial F_1}{\partial \phi_N}\right)^{(p-1)} \\ \vdots & \ddots & \vdots \\ \left(\frac{\partial F_N}{\partial \phi_1}\right)^{(p-1)} & \cdots & \left(\frac{\partial F_N}{\partial \phi_N}\right)^{(p-1)} \end{pmatrix}^{-1} \begin{Bmatrix} F_1(\phi^{(p-1)}) \\ \vdots \\ F_N(\phi^{(p-1)}) \end{Bmatrix}$$

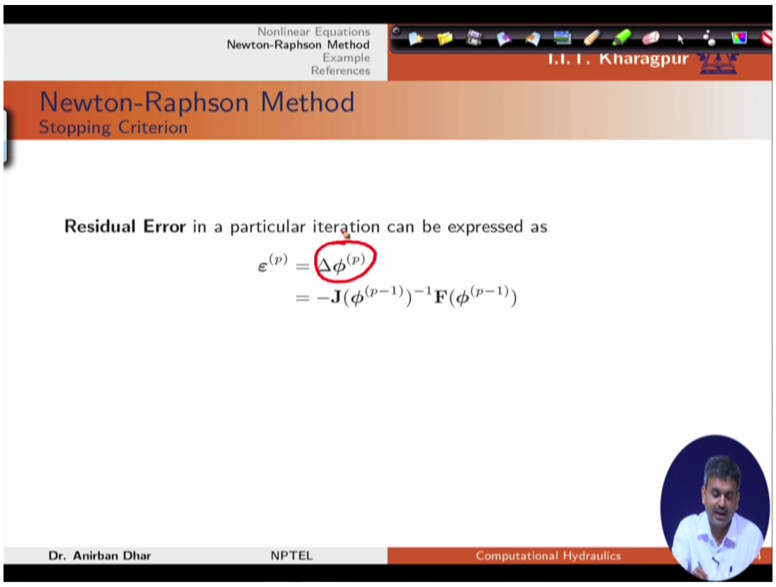
$$- A^{-1} r$$

$$\boxed{A \Delta\phi = \boxed{-r}}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics 9 / 14

Now residual error like our iterative technique we need to see residual error. This epsilon at pth iteration again this is the increment. So obviously this increment should be checked.

(Refer Slide Time: 17:22)


Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Newton-Raphson Method
Stopping Criterion

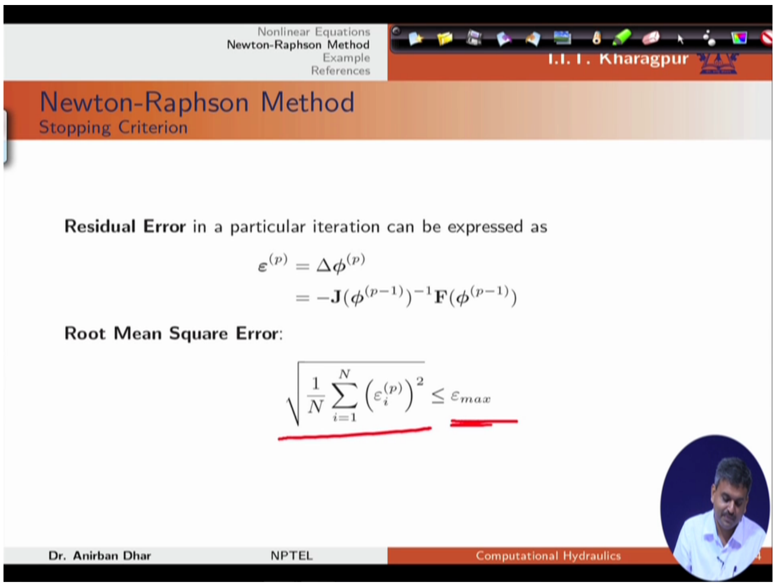
Residual Error in a particular iteration can be expressed as

$$\begin{aligned}\varepsilon^{(p)} &= \boxed{\Delta\phi^{(p)}}\\ &= -\mathbf{J}(\phi^{(p-1)})^{-1}\mathbf{F}(\phi^{(p-1)})\end{aligned}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

Now what we can do, we can calculate root mean square error and this root mean square error should be less than equal to some specified root mean square error level.

(Refer Slide Time: 17:45)


Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Newton-Raphson Method
Stopping Criterion

Residual Error in a particular iteration can be expressed as

$$\begin{aligned}\varepsilon^{(p)} &= \Delta\phi^{(p)}\\ &= -\mathbf{J}(\phi^{(p-1)})^{-1}\mathbf{F}(\phi^{(p-1)})\end{aligned}$$

Root Mean Square Error:

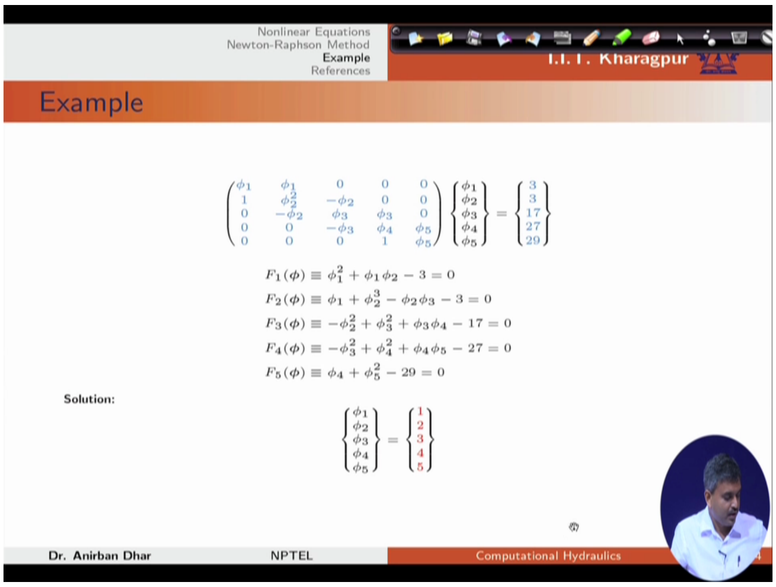
$$\underline{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\varepsilon_i^{(p)}\right)^2}} \le \underline{\varepsilon_{max}}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

So this is a stopping criteria for this Newton Raphson method. Now let us consider one example. Example is that we can multiply this particular matrix which is A matrix. This is with variable coefficients. This is not with constant coefficients because phi 1, phi 2, phi 3, these are included in the coefficient matrix. Now if we simplify this we can write it in terms of F1, F2, F3, F4, F5.

Now the solution, we already know for this system is 1, 2, 3, 4, 5. Now let us see how to solve this problem using Newton Raphson approach with our scilab programming.

(Refer Slide Time: 18:48)


Nonlinear Equations
Newton-Raphson Method
Example
References
I.I.T. Kharagpur

Example

$$\begin{pmatrix} \phi_1 & \phi_1 & 0 & 0 & 0 \\ 1 & \phi_2^2 & -\phi_2 & 0 & 0 \\ 0 & -\phi_2 & \phi_3 & \phi_3 & 0 \\ 0 & 0 & -\phi_3 & \phi_4 & \phi_5 \\ 0 & 0 & 0 & 1 & \phi_5 \end{pmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \end{Bmatrix} = \begin{Bmatrix} 3 \\ 3 \\ 17 \\ 27 \\ 29 \end{Bmatrix}$$

$$F_1(\phi) \equiv \phi_1^2 + \phi_1\phi_2 - 3 = 0$$
$$F_2(\phi) \equiv \phi_1 + \phi_2^3 - \phi_2\phi_3 - 3 = 0$$
$$F_3(\phi) \equiv -\phi_2^2 + \phi_3^2 + \phi_3\phi_4 - 17 = 0$$
$$F_4(\phi) \equiv -\phi_3^2 + \phi_4^2 + \phi_4\phi_5 - 27 = 0$$
$$F_5(\phi) \equiv \phi_4 + \phi_5^2 - 29 = 0$$

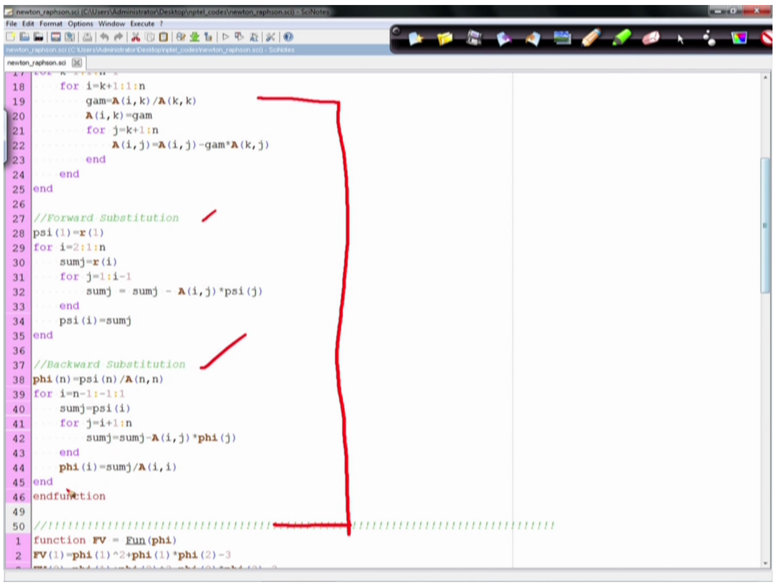
Solution:

$$\begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{Bmatrix}$$

Now this part is clear because initially we start the program with clc that means clear the screen and clear the console. Then clear means we are clearing all the values that are stored in scilab from our previous operation. Now in this case I am utilising LU decomposition. Whatever LU decomposition code we have written using scilab I will be directly utilising for this particular step.

Now starting from this one, we can say that up to this we have LU decomposition. Now this is forward substitution, then backward substitution and this is the end function.

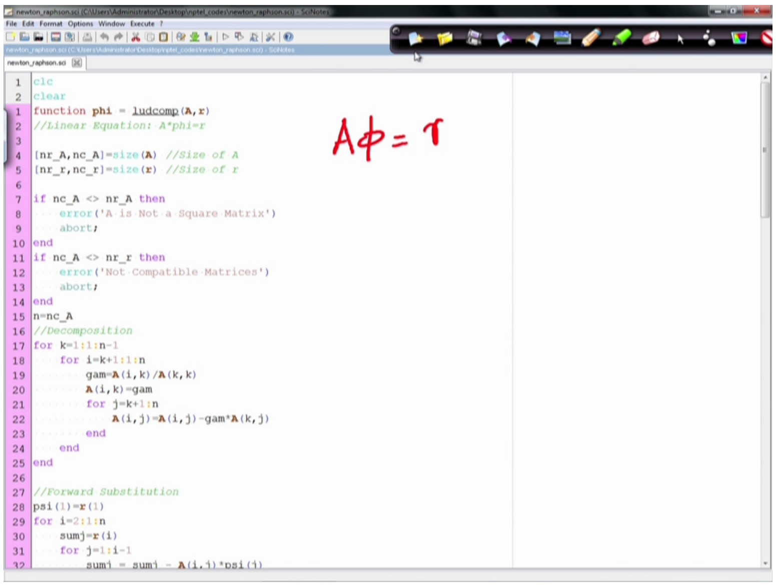
(Refer Slide Time: 20:14)


newton_raphson.sci (C:\Users\Administrator\Desktop\nptel_codes\newton_raphson.sci) - SciNotes
File Edit Format Options Window Execute ?
newton_raphson.sci

```
    for i=k+1:1:n
        gam=A(i,k)/A(k,k)
        A(i,k)=gam
        for j=k+1:n
            A(i,j)=A(i,j)-gam*A(k,j)
        end
    end
end

//Forward Substitution
psi(1)=r(1)
for i=2:1:n
    sumj=r(i)
    for j=1:i-1
        sumj = sumj - A(i,j)*psi(j)
    end
    psi(i)=sumj
end

//Backward Substitution
phi(n)=psi(n)/A(n,n)
for i=n-1:-1:1
    sumj=psi(i)
    for j=i+1:n
        sumj=sumj-A(i,j)*phi(j)
    end
    phi(i)=sumj/A(i,i)
end
endfunction

//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
function FV = Fun(phi)
FV(1)=phi(1)^2+phi(1)*phi(2)-3
```

So final we will be getting the phi value like our original problem which is linear in nature, A phi equals to r.

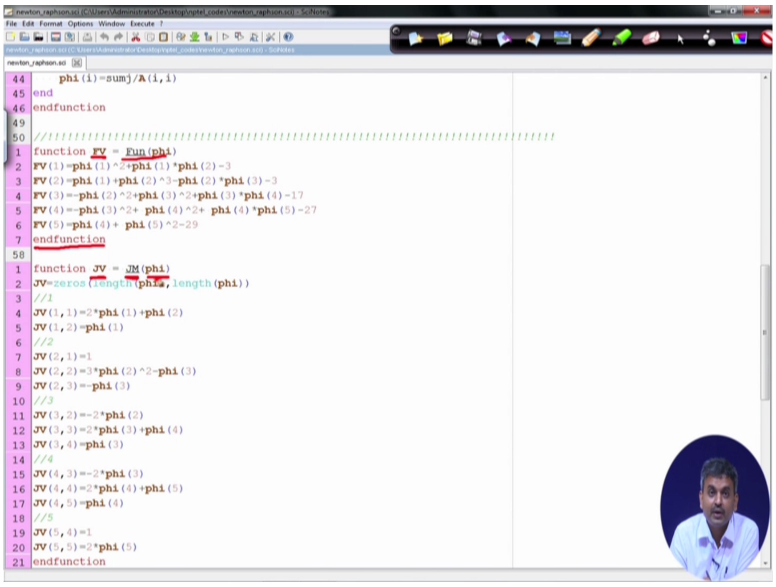
(Refer Slide Time: 20:30)


newton_raphson.sci (C:\Users\Administrator\Desktop\nptel_codes\newton_raphson.sci) - SciNotes
File Edit Format Options Window Execute ?
newton_raphson.sci
$A\phi = r$

```
clc
clear
function phi = ludcomp(A,r)
//Linear Equation: A*phi=r

[nr_A,nc_A]=size(A) //Size of A
[nr_r,nc_r]=size(r) //Size of r

if nc_A <> nr_A then
    error('A is Not a Square Matrix')
    abort;
end
if nc_A <> nr_r then
    error('Not Compatible Matrices')
    abort;
end
n=nc_A
//Decomposition
for k=1:1:n-1
    for i=k+1:1:n
        gam=A(i,k)/A(k,k)
        A(i,k)=gam
        for j=k+1:n
            A(i,j)=A(i,j)-gam*A(k,j)
        end
    end
end

//Forward Substitution
psi(1)=r(1)
for i=2:1:n
    sumj=r(i)
    for j=1:i-1
        sumj = sumj - A(i,j)*psi(j)
```

Now let us write other vectors or matrixes for a Newton Raphson method. So for Newton Raphson method we need function values. So for function values I have included one function which is FV. FV is the function. This will give me function value. So Fun phi. So if you (in) give input phi, so this will give me the function value and this is end function. And another thing is required that is Jacobian value or JV. This is Jacobian matrix. We are giving input phi. So we should get Jacobian matrix for this one.

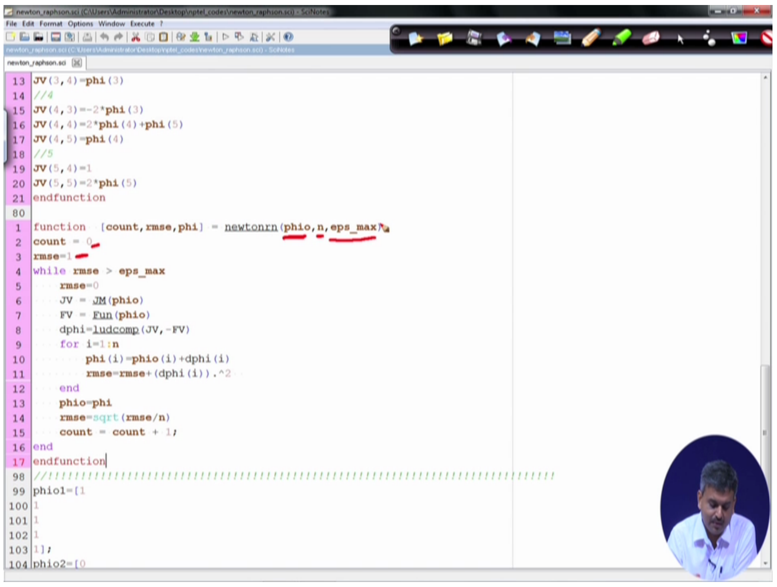
(Refer Slide Time: 21:21)


newton_raphson.sci (C:\Users\Administrator\Desktop\nptel_codes\newton_raphson.sci) - SciNotes
File Edit Format Options Window Execute ?
newton_raphson.sci

```
    phi(i)=sumj/A(i,i)
end
endfunction

//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
function FV = Fun(phi)
FV(1)=phi(1)^2+phi(1)*phi(2)-3
FV(2)=phi(1)+phi(2)^3-phi(2)*phi(3)-3
FV(3)=-phi(2)^2+phi(3)^2+phi(3)*phi(4)-17
FV(4)=-phi(3)^2+ phi(4)^2+ phi(4)*phi(5)-27
FV(5)=phi(4)+ phi(5)^2-29
endfunction

function JV = JM(phi)
JV=zeros(length(phi),length(phi))
//1
JV(1,1)=2*phi(1)+phi(2)
JV(1,2)=phi(1)
//2
JV(2,1)=1
JV(2,2)=3*phi(2)^2-phi(3)
JV(2,3)=-phi(3)
//3
JV(3,2)=-2*phi(2)
JV(3,3)=2*phi(3)+phi(4)
JV(3,4)=phi(3)
//4
JV(4,3)=-2*phi(3)
JV(4,4)=2*phi(4)+phi(5)
JV(4,5)=phi(4)
//5
JV(5,4)=1
JV(5,5)=2*phi(5)
endfunction
```

Now if we complete this function value and Jacobian matrix then we can start our original algorithm for Newton Raphson. Now how to write that algorithm? So we can start from simple count equals to zero. Count means number of iterations counter in this case. And rmse is equals to 1 to enter in this while loop. So phi o, this is Newton rn, Newton Raphson method. So Newton rn phi o is the guess value. N, n is the number of variables. Eps max, this is the limit for rmse.

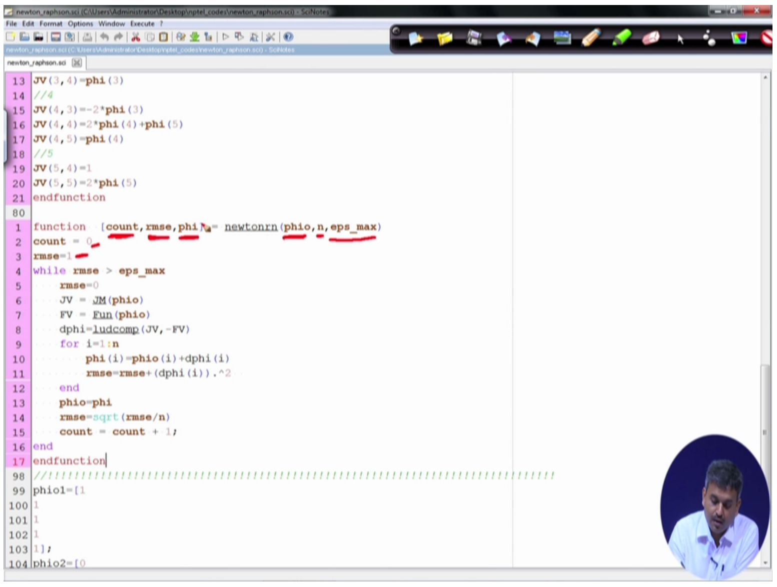
(Refer Slide Time: 22:39)



```
JV(3,4)=phi(3)
//4
JV(4,3)=-2*phi(3)
JV(4,4)=2*phi(4)+phi(5)
JV(4,5)=phi(4)
//5
JV(5,4)=1
JV(5,5)=2*phi(5)
endfunction

function  [count,rmse,phi] = newtonrn(phio,n,eps_max)
count = 0
rmse=1
while rmse > eps_max
    rmse=0
    JV = JM(phio)
    FV = Fun(phio)
    dphi=ludcomp(JV,-FV)
    for i=1:n
        phi(i)=phio(i)+dphi(i)
        rmse=rmse+(dphi(i)).^2
    end
    phio=phi
    rmse=sqrt(rmse/n)
    count = count + 1;
end
endfunction
//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
phio1=[1
1
1
1
1];
phio2=[0
```

Now what is the output? Output is count that means number of iterations required. Rmse, root mean square error and phi that means after iteration whatever value we are getting as solution.

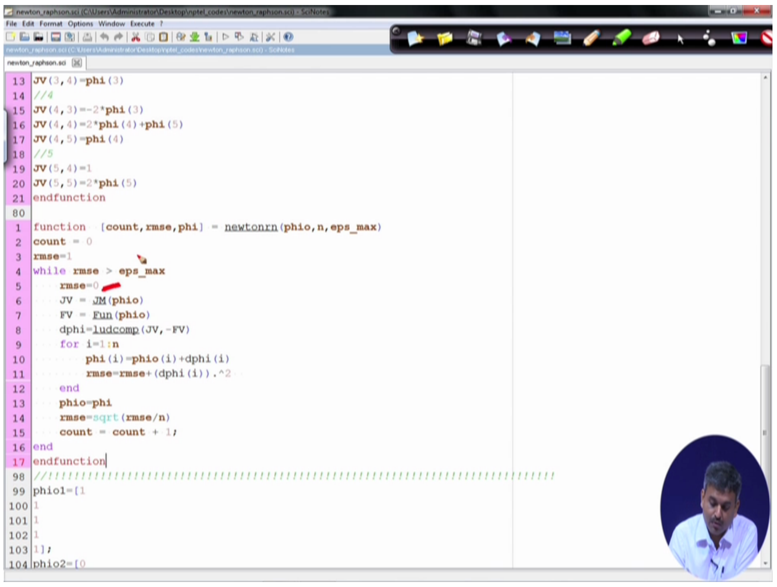
(Refer Slide Time: 22:59)



```
JV(3,4)=phi(3)
//4
JV(4,3)=-2*phi(3)
JV(4,4)=2*phi(4)+phi(5)
JV(4,5)=phi(4)
//5
JV(5,4)=1
JV(5,5)=2*phi(5)
endfunction

function  [count,rmse,phi] = newtonrn(phio,n,eps_max)
count = 0
rmse=1
while rmse > eps_max
    rmse=0
    JV = JM(phio)
    FV = Fun(phio)
    dphi=ludcomp(JV,-FV)
    for i=1:n
        phi(i)=phio(i)+dphi(i)
        rmse=rmse+(dphi(i)).^2
    end
    phio=phi
    rmse=sqrt(rmse/n)
    count = count + 1;
end
endfunction
//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
phio1=[1
1
1
1
1];
phio2=[0
```

Now in this case let us start the while loop. Rmse greater than eps max. Obviously if we write rmse equals to 1 that is greater than eps max. And after entering into the loop and writing this rmse equals to zero because I need to calculate this rmse within this calculation or while loop.

(Refer Slide Time: 23:24)

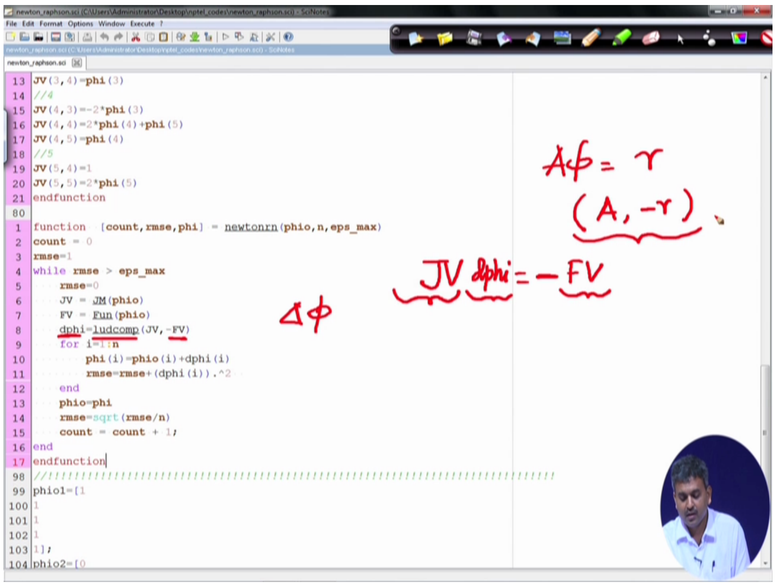


```
JV(3,4)=phi(3)
//4
JV(4,3)=-2*phi(3)
JV(4,4)=2*phi(4)+phi(5)
JV(4,5)=phi(4)
//5
JV(5,4)=1
JV(5,5)=2*phi(5)
endfunction

function  [count,rmse,phi] = newtonrn(phio,n,eps_max)
count = 0
rmse=1
while rmse > eps_max
    rmse=0
    JV = JM(phio)
    FV = Fun(phio)
    dphi=ludcomp(JV,-FV)
    for i=1:n
        phi(i)=phio(i)+dphi(i)
        rmse=rmse+(dphi(i)).^2
    end
    phio=phi
    rmse=sqrt(rmse/n)
    count = count + 1;
end
endfunction
//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
phio1=[1
1
1
1
1];
phio2=[0
```

Now we can first calculate this our Jacobian matrix and function value. So if we write this, so d phi. What is this d phi? D phi is nothing but d phi in this case. So first we have calculated Jacobian matrix, then we have calculated function value or FV. JV, FV and this is our d phi. Now this is matrix, this is column vector, this is column vector. Essentially JV is with constant coefficient for a particular value of phi and FV is again constant value and del phi or d phi is your variable vector.

So in this case if we utilise LU decomposition with a negative sign to minus r because this is required. We need to place minus sign before r to get the solution of del phi. Because this is the general structure. General structure was A phi equals to r. But if you want to use LU decomposition to solve this one that is JV into del phi minus FV, we can transfer this information. That A and r matrix, we can directly transfer to LU decomposition code to get the solution.

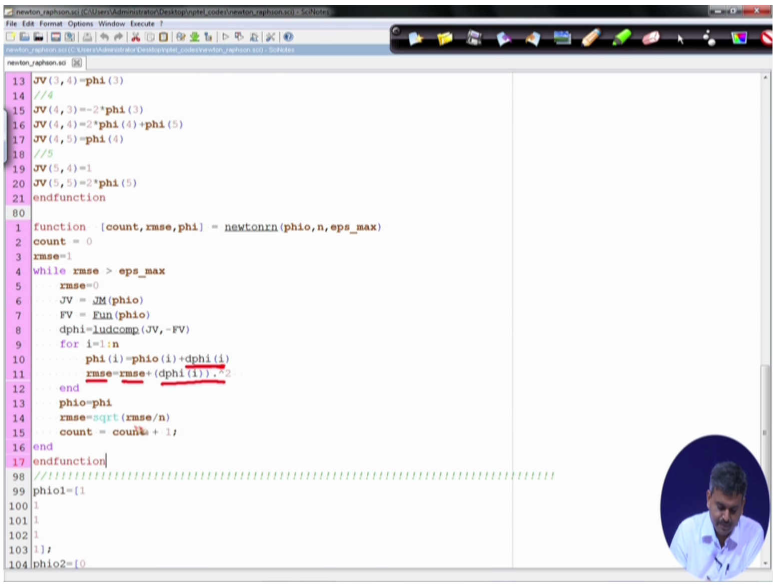
(Refer Slide Time: 25:30)



```
13 JV(3,4)=phi(3)
14 //4
15 JV(4,3)=-2*phi(3)
16 JV(4,4)=2*phi(4)+phi(5)
17 JV(4,5)=phi(4)
18 //5
19 JV(5,4)=1
20 JV(5,5)=2*phi(5)
21 endfunction
80
1  function  [count,rmse,phi] = newtonrn(phio,n,eps_max)
2  count = 0
3  rmse=1
4  while rmse > eps_max
5      rmse=0
6      JV = JM(phio)
7      FV = Fun(phio)
8      dphi=ludcomp(JV,-FV)
9      for i=1:n
10         phi(i)=phio(i)+dphi(i)
11         rmse=rmse+(dphi(i)).^2
12     end
13     phio=phi
14     rmse=sqrt(rmse/n)
15     count = count + 1;
16 end
17 endfunction
98 //!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
99 phio1=[1
100 1
101 1
102 1
103 1];
104 phio2=[0
```

So with this we can add del phi i. That means ith update. And similarly we can calculate rmse or by adding the del phi square values.

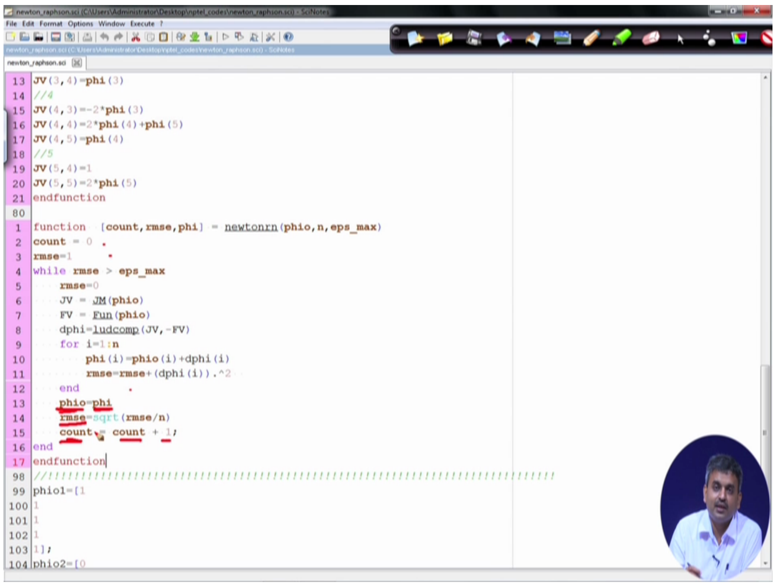
(Refer Slide Time: 25:54)



```
13 JV(3,4)=phi(3)
14 //4
15 JV(4,3)=-2*phi(3)
16 JV(4,4)=2*phi(4)+phi(5)
17 JV(4,5)=phi(4)
18 //5
19 JV(5,4)=1
20 JV(5,5)=2*phi(5)
21 endfunction
80
1  function  [count,rmse,phi] = newtonrn(phio,n,eps_max)
2  count = 0
3  rmse=1
4  while rmse > eps_max
5      rmse=0
6      JV = JM(phio)
7      FV = Fun(phio)
8      dphi=ludcomp(JV,-FV)
9      for i=1:n
10         phi(i)=phio(i)+dphi(i)
11         rmse=rmse+(dphi(i)).^2
12     end
13     phio=phi
14     rmse=sqrt(rmse/n)
15     count = count + 1;
16 end
17 endfunction
98 //!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
99 phio1=[1
100 1
101 1
102 1
103 1];
104 phio2=[0
```

And after this update we can store in place of old value, the new value. And we can again start the iteration thing. At the same time we are also calculating rmse. Rmse equals to square root. Rmse divided by N because we have added all the values, divided by N. N is the number of variable. And count should be count equals to count plus 1. So this is you can see that 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. So 15 lines are required to write this Newton Raphson algorithm.

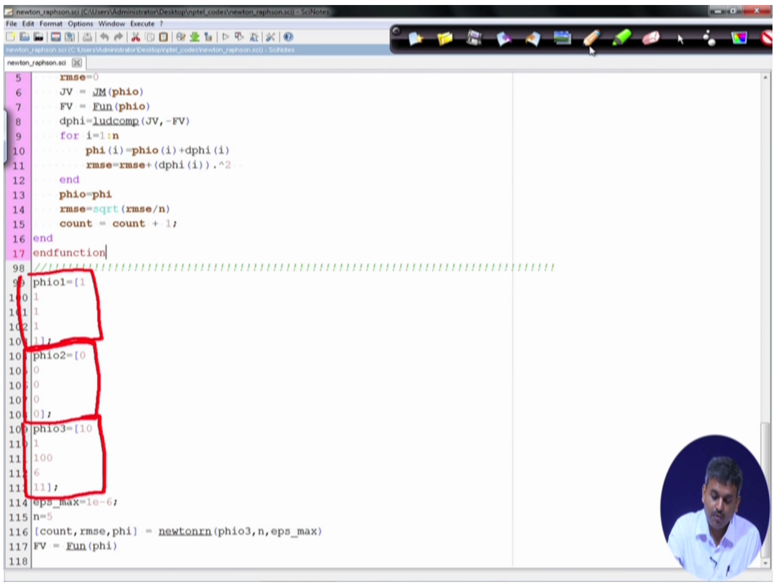
(Refer Slide Time: 26:39)



```
JV(3,4)=phi(3)
//4
JV(4,3)=-2*phi(3)
JV(4,4)=2*phi(4)+phi(5)
JV(4,5)=phi(4)
//5
JV(5,4)=1
JV(5,5)=2*phi(5)
endfunction

function  [count,rmse,phi] = newtonrn(phio,n,eps_max)
count = 0
rmse=1
while rmse > eps_max
    rmse=0
    JV = JM(phio)
    FV = Fun(phio)
    dphi=ludcomp(JV,-FV)
    for i=1:n
        phi(i)=phio(i)+dphi(i)
        rmse=rmse+(dphi(i)).^2
    end
    phio=phi
    rmse=sqrt(rmse/n)
    count = count + 1;
end
endfunction
//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
phio1=[1
1
1
1
1];
phio2=[0
```

Now we can solve it easily using Newton Raphson. Another important thing is that initial guess. For linear case it is much easier to get the solution with our arbitrary values. But it is difficult for nonlinear case. So let us examine that. We have two situations. Let us say phi o 1 where all values we are starting from 1. Phi o 2 where we are starting all values from zero. Phi o 3, this is arbitrary. Arbitrary values, one value is 10 and other one is 1, 100, 6, 11, anything.

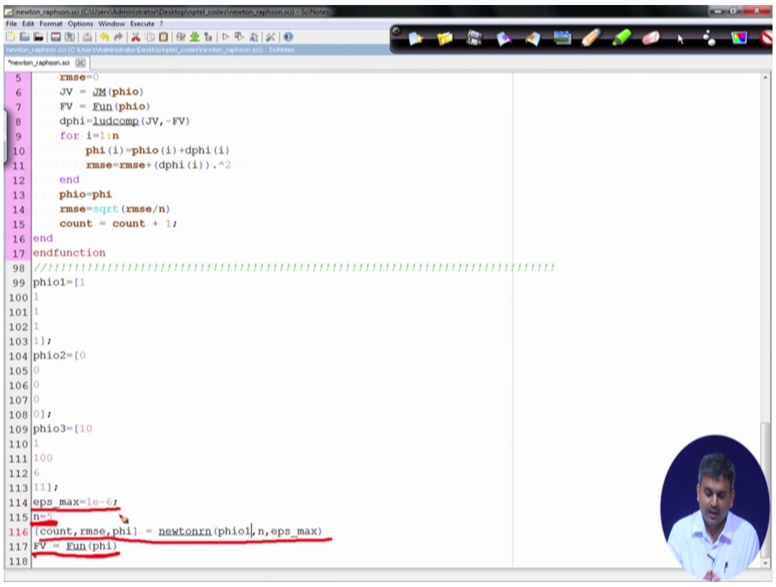
(Refer Slide Time: 27:42)



```
    rmse=0
    JV = JM(phio)
    FV = Fun(phio)
    dphi=ludcomp(JV,-FV)
    for i=1:n
        phi(i)=phio(i)+dphi(i)
        rmse=rmse+(dphi(i)).^2
    end
    phio=phi
    rmse=sqrt(rmse/n)
    count = count + 1;
end
endfunction
//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
phio1=[1
1
1
1
1];
phio2=[0
0
0
0
0];
phio3=[10
1
100
6
11];
eps_max=1e-6;
n=5
[count,rmse,phi] = newtonrn(phio3,n,eps_max)
FV = Fun(phi)
```

So with these three values we will try to examine what is the solution. In this case we have utilised eps max, this 1 into 10 to the power minus 6. And n equals to 5 because we have 5 numbers of variables. And so we can call this Newton Raphson method with values. Let us start with phi o 1. That means first guess which is 1, 1, 1, 1. That means all values are 1. And finally we are calculating this FV or Fun. That means function value. That means if we calculate (fa) function value at solution, it should be close to zero.

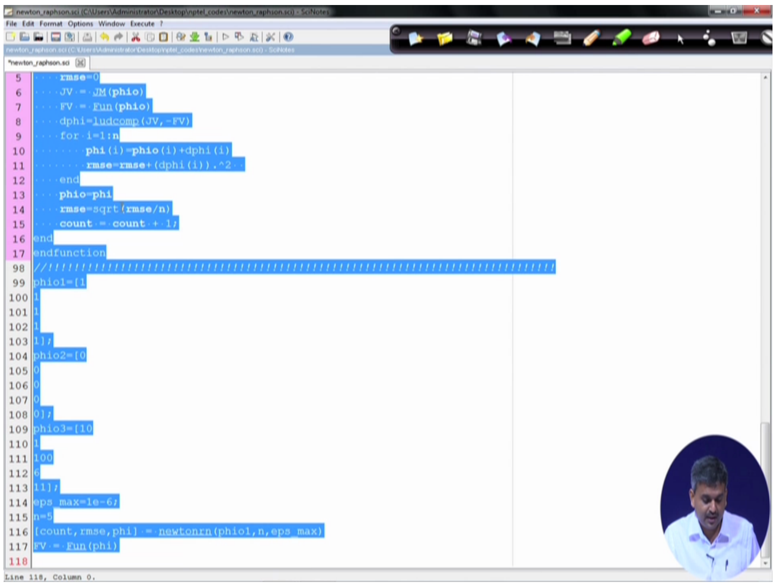
(Refer Slide Time: 28:40)


newton_raphson.sci (C:\Users\Administrator\Desktop\nptel_codes\newton_raphson.sci) - SciNotes
File Edit Format Options Window Execute ?
*newton_raphson.sci

```
    rmse=0
    JV = JM(phio)
    FV = Fun(phio)
    dphi=ludcomp(JV,-FV)
    for i=1:n
        phi(i)=phio(i)+dphi(i)
        rmse=rmse+(dphi(i)).^2
    end
    phio=phi
    rmse=sqrt(rmse/n)
    count = count + 1;
end
endfunction
//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
phio1=[1
1
1
1
1];
phio2=[0
0
0
0
0];
phio3=[10
1
100
6
11];
eps_max=1e-6;
n=5
[count,rmse,phi] = newtonrn(phio1,n,eps_max)
FV = Fun(phi)
```

So let us try this. So we can select this and we can run this.

(Refer Slide Time: 28:55)

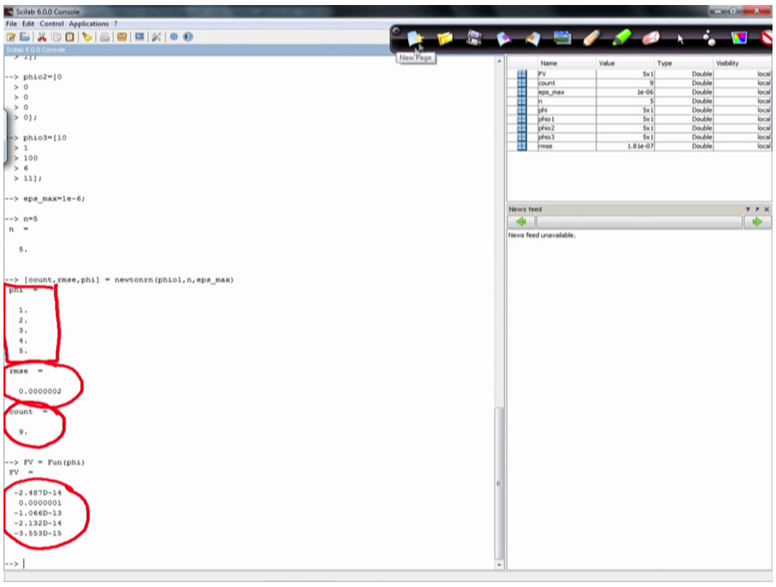

newton_raphson.sci (C:\Users\Administrator\Desktop\nptel_codes\newton_raphson.sci) - SciNotes
File Edit Format Options Window Execute ?
*newton_raphson.sci

```
    rmse=0
    JV = JM(phio)
    FV = Fun(phio)
    dphi=ludcomp(JV,-FV)
    for i=1:n
        phi(i)=phio(i)+dphi(i)
        rmse=rmse+(dphi(i)).^2
    end
    phio=phi
    rmse=sqrt(rmse/n)
    count = count + 1;
end
endfunction
//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
phio1=[1
1
1
1
1];
phio2=[0
0
0
0
0];
phio3=[10
1
100
6
11];
eps_max=1e-6;
n=5
[count,rmse,phi] = newtonrn(phio1,n,eps_max)
FV = Fun(phi)
```

Line 118, Column 0.

If we run this process then easily we can see that we are exactly getting our solution phi equals to 1, 2, 3, 4, 5. That is our solution. And rmse we are getting this point 1, 2, 3, 4, 5, 6, 6 not 2 and with counter 9. That means within 9 iterations we are getting this solution. And finally whatever function value is there that is minus 10 to the power minus 4, minus 13, minus 14, minus 15 and this is very near to zero. So we can say that our solution that is most correct in this case.

(Refer Slide Time: 29:51)



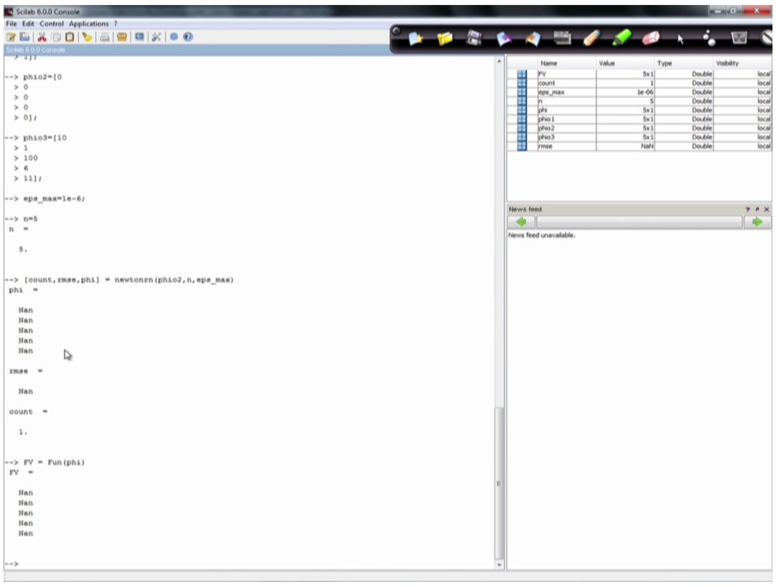
```
--> phio2=[0
  > 0
  > 0
  > 0
  > 0];

--> phio3=[10
  > 1
  > 100
  > 6
  > 11];

--> eps_max=1e-6;

--> n=5
 n  =

   5.


--> [count,rmse,phi] = newtonrn(phio1,n,eps_max)
 phi  =

   1.
   2.
   3.
   4.
   5.

 rmse  =

   0.0000002

 count  =

   9.


--> FV = Fun(phi)
 FV  =

  -2.487D-14
   0.0000001
  -1.066D-13
  -2.132D-14
  -3.553D-15


--> 
```

Name	Value	Type	Visibility
FV	5x1	Double	local
count	9	Double	local
eps_max	1e-06	Double	local
n	5	Double	local
phi	5x1	Double	local
phio1	5x1	Double	local
phio2	5x1	Double	local
phio3	5x1	Double	local
rmse	1.81e-07	Double	local

News feed unavailable.

But let us see what is there if we change the initial guess? Let us say that all initial guess values are zero. Or all variable values are zero or phi o 2. So in this case we evaluate it we are getting Nan. Why we are getting Nan? Because we are unable to calculate the Jacobian matrix properly. All entries in Jacobian matrix is zero. So in this case it is difficult to get the solution or it is impossible to get the solution with this guess.

(Refer Slide Time: 30:30)



```
--> phio2=[0
  > 0
  > 0
  > 0
  > 0];

--> phio3=[10
  > 1
  > 100
  > 6
  > 11];

--> eps_max=1e-6;

--> n=5
 n  =

   5.


--> [count,rmse,phi] = newtonrn(phio2,n,eps_max)
 phi  =

   Nan
   Nan
   Nan
   Nan
   Nan

 rmse  =

   Nan

 count  =

   1.


--> FV = Fun(phi)
 FV  =

   Nan
   Nan
   Nan
   Nan
   Nan


--> 
```

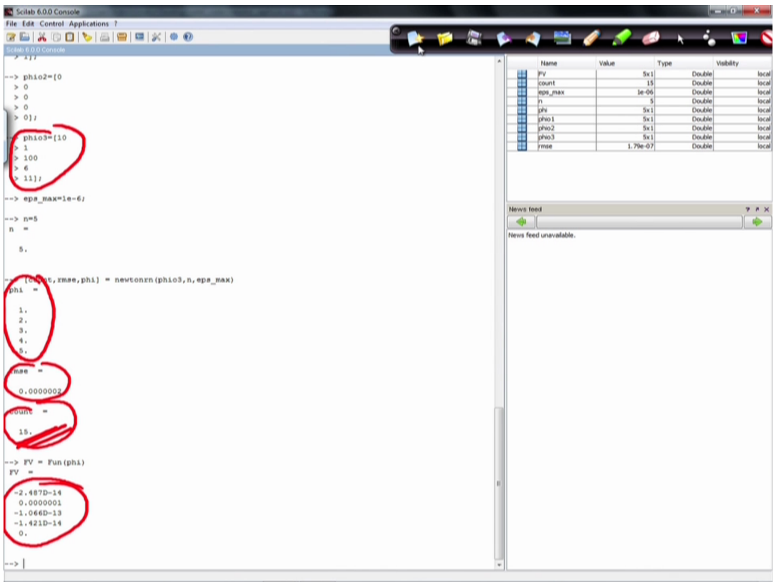
Name	Value	Type	Visibility
FV	5x1	Double	local
count	1	Double	local
eps_max	1e-06	Double	local
n	5	Double	local
phi	5x1	Double	local
phio1	5x1	Double	local
phio2	5x1	Double	local
phio3	5x1	Double	local
rmse	NaN	Double	local

News feed unavailable.

Now if we use arbitrary value for this phi o 3, then again we are getting the solution 1, 2, 3, 4, 5 and rmse is also less. Count, this is important because in previous case when we started the solution with 1, 1, 1 value. That means for first variable we have provided exact value and for others there is less deviation in the variable values. So in this particular case we are using this phi o 3. That means 10, 1, 100, 6, 11. These arbitrary values are there.

Obviously you need more iteration to get convergence. So in this case with 15 iterations we are getting the solution. Again our function values are very close to zero. I think this is clear that depending on the initial or guess value our solution thing will change.

(Refer Slide Time: 32:17)



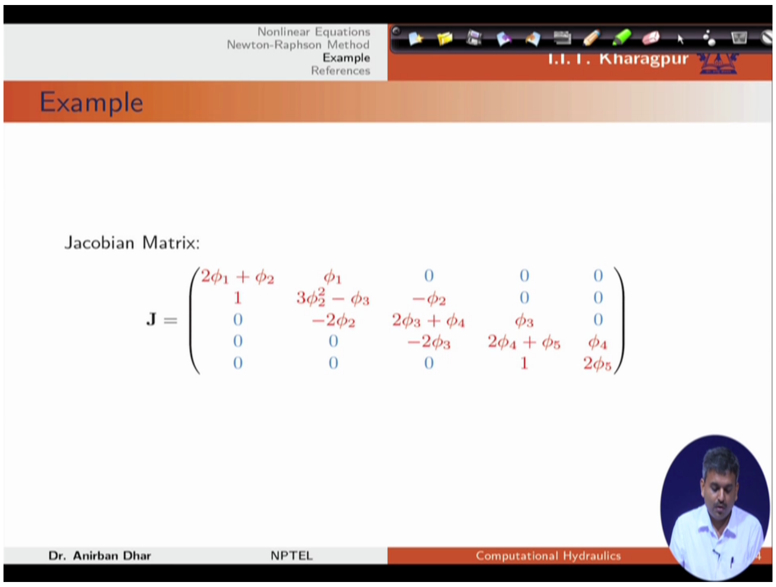
```
--> phio2=[0
  > 0
  > 0
  > 0
  > 0];

--> phio3=[10
  > 1
  > 100
  > 6
  > 11];

--> eps_max=1e-6;

--> n=5
 n  =

   5.

--> [count,rmse,phi] = newtonrn(phio3,n,eps_max)
 phi  =

   1.
   2.
   3.
   4.
   5.
 rmse  =

   0.0000002
 count  =

   15.

--> FV = Fun(phi)
 FV  =

  -2.487D-14
   0.0000001
  -1.066D-13
  -1.421D-14
   0.

--> 
```

Name	Value	Type	Visibility
FV	5x1	Double	local
count	15	Double	local
eps_max	1e-06	Double	local
n	5	Double	local
phi	5x1	Double	local
phio1	5x1	Double	local
phio2	5x1	Double	local
phio3	5x1	Double	local
rmse	1.79e-07	Double	local

News feed unavailable.

So let us examine it from our Jacobian matrix. If we see the Jacobian matrix, the Jacobian matrix almost all diagonal terms that will be zero if we provide this zero value as initial value for the solution process.

(Refer Slide Time: 32:50)


Nonlinear Equations
Newton-Raphson Method
Example
References

I.I.T. Kharagpur

Example

Jacobian Matrix:

$$\mathbf{J} = \begin{pmatrix} 2\phi_1+\phi_2 & \phi_1 & 0 & 0 & 0 \\ 1 & 3\phi_2^2-\phi_3 & -\phi_2 & 0 & 0 \\ 0 & -2\phi_2 & 2\phi_3+\phi_4 & \phi_3 & 0 \\ 0 & 0 & -2\phi_3 & 2\phi_4+\phi_5 & \phi_4 \\ 0 & 0 & 0 & 1 & 2\phi_5 \end{pmatrix}$$

Dr. Anirban Dhar NPTEL Computational Hydraulics

So with this we are finishing this numerical method, this module and from next module onwards we will be discussing the specific discretization for our computational hydraulics problem and we will try to solve those problems using this scilab codes. Thank you.