Computer Aided Power System Analysis
Prof. Biswarup Das
Department of Electrical Engineering
Indian Institute of Technology-Roorkee

Lecture - 07
Basic Gauss-Seidel Numeric Method

Welcome to this lecture of the course computer aided power system analysis. In the last lecture we have looked into the power flow equations and we also we have also looked into the classification of buses and also we have said that depending upon the type of the buses there are two quantities which are being specified at each and every buses and the remaining 2 quantities are to be calculated.

So then therefore altogether for N bus system although we do have for N quantities to be known but because we have got only 2N number of equations so then therefore we need to actually pre-specify 2N number of quantities so that we can calculate the remaining 2N number of quantities by using this 2N number of equations. So now from this lecture onwards we would be looking into the different methods of solving this 2N powerful equations to calculate this 2N quantities.

As we have already mentioned at the end of the last lectures, essentially we are interested to calculate the voltage magnitudes and the angles at each and every bus because if we know the voltage magnitude and the angle at each and every bus we are able to calculate the value of injected real and reactive power at each and every bus. So then therefore we really need not calculate the injected value of P and Q i at the other buses explicitly.

We actually should only concentrate on calculating the voltage magnitude on angle and after that having the knowledge of voltage magnitude and angle with us we would be able to calculate the values of P and Q i and also the other quantities in the system. For example the line current magnitudes, line current angles, power flow at each and every lines everything. Everything would be known once we know the voltage magnitudes and angles at each and every bus.

(Refer Slide Time: 02:50)

Power-flow equations

$$P_i = \sum_{k=1}^{N} V_i V_k Y_{ik} \cos(\theta_i - \theta_k - \alpha_{ik})$$

$$Q_i = \sum_{k=1}^{N} V_i V_k Y_{ik} \sin(\theta_i - \theta_k - \alpha_{ik})$$

Non-linear equations

Methods

i) Gauss-Seidel

ii) Newton-Raphson — Polar / Rectangular

iii) Fast Decoupled

So now let us again look at this power flow equation just to recollect, just recollect this power flow equations. So power flow equations are P i V i, V K. Now because of the presence of cosine and sine terms, so these equations are nonlinear equations. So then therefore our power system is actually represented by a set of 2N nonlinear simultaneous algebraic equations which we need to solve for the voltage magnitudes and the angles at each and every bus.

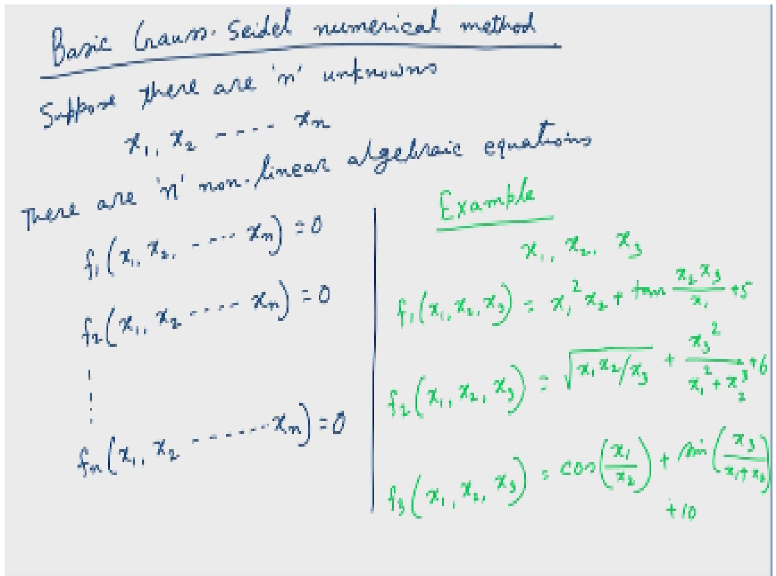
Now because this equations are nonlinear equation so then therefore there is no a close form solution of this equation. So we have to only solve them by using some appropriate numerical method. Now for solving this power flow equations usually we do use 4 different numerical methods. Actually this is 3. So these methods are, so numerical methods are one is Gauss-Seidel. Then Newton-Raphson. Now Newton-Raphson has 2 versions. One is polar, one is rectangular.

We will discuss what is meant by this polar and as well as rectangular version and third is Fast Decoupled. So then therefore there are although there are 3 major types, there are actually 4 types. Gauss-Seidel, Newton-Raphson polar, Newton-Raphson rectangular, and Fast Decoupled method for solving this power flow equations to calculate the voltage magnitudes and angles at each and every bus.

Now today in this lecture, we would be talking about the basic Gauss-Seidel method, to introduce with the basic Gauss-Seidel method such that I mean once we know this very basic

Gauss-Seidel method that how does it proceed so then we would be able to apply it for solving this equations. So let us today our task is to discuss about the basic Gauss-Seidel numerical method.

(Refer Slide Time: 06:13)



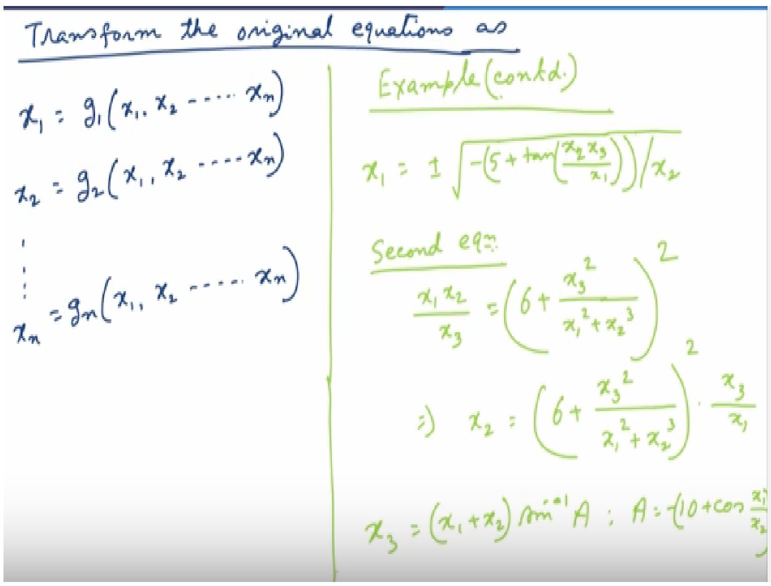
So we are discussing today basic Gauss-Seidel numerical method. Say suppose there are n unknowns. There are unknowns. There are n unknowns x 1, x 2 up to x n. And also there are n equations, n nonlinear simultaneous, nonlinear algebraic equations say f 1(x 1, x 2 up to x n) = 0. Remember f 1 is a function of x 1, x 2 and x n. So this is basically any expression. Now here we are not assuming any particular form of f 1. I mean this can be anything.

F 2 (x 1, x 2, x n) = 0 … f n (x 1, x 2…x n) = 0. So here f 1, f 2, f n are essentially nothing but the functions of x 1, x 2, x n. So then therefore these functions are basically representing some equations involving x 1, x 2, x n and here in this case we are not assuming any particular form of this function. They can be any function having all kind of nonlinearities. Just as an example, suppose my unknowns are x 1, x 2, x 3.

Suppose there are 3 unknowns and one equation is let us say f 1 is let us say I am just drawing it arbitrarily here x 1 square x 2 + tan x 2 x 3 x 1 + 5. Let us say this is f 1. F 2 (x1, x 2, x 3 is let us say root over x 1 x 2/x 3 + x 3 whole square/x 1 square + x 2 cube + 6. And f 3 is x 1, x 2, x 3 is let us say cosine (x 1/x 2) + sin (x 3/x 1 + x 2) + 10. Remember here simply we have written arbitrarily these functions. So here you know all kind of nonlinearity is there.

There is a square, there is a tangent function, there is a cosine function, there is a sine function, there is a root over, there is square, everything. There is a cube everything, everything. So then here what we have? So essentially what we want to stress that there is absolutely no limitation of this form which this functions f 1, f 2, f n can take. I mean these can be any functions. So now having this original equations then what we do is we transform this original equations.

(Refer Slide Time: 10:28)



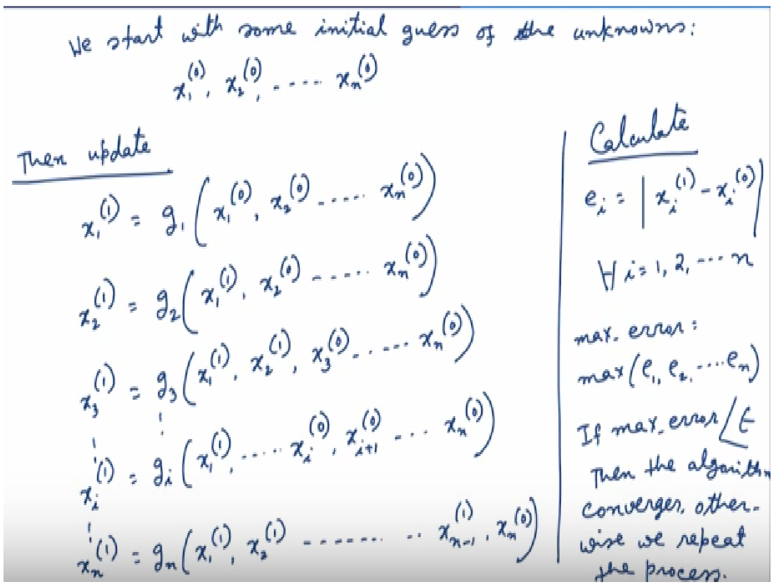
Transform the original equations as we say we actually express x 1 as some other function of g 1 x n. x 2 as some other function of g 2 (x1, x 2, x n) and x n as some other function of x 1, x 2, x n. for example if we take this, if we take these examples for example from the first equation, we can say that x 1 is equal to again for example if we take this first equation example continued. From the first equation we can write down that x 1 = plus minus root over minus of by x 2.

X 2 is from the second equation it would be from x 2 is for example from second equation x 1, x 2, x 3, x 1 x 2/ x3 would be equal to 6 + x 3 square by. So x 2 = 6 + x 3 square + x 1 square + x 2 cube whole square into x 3 by x 1. Similarly, from the third equation, x 3 can be represented. For example from the third equation x 3 would be (x 1 + x 2) sin inverse A where A is -10 + cosine.

So you see here so from the original equations by using some simple algebra we can simply represent x 1, x 2 and x3 as some other function of x 1, x 2, x 3, some function. So we need not really worry. Now here you can see from these 3 expression it is just not possible to calculate the or rather to find out the expression of x 1, x 2 and x 3 by any closed form solution. So we have to use some kind of numerical techniques. So now how do you proceed?

So because it is not possible to find out any close form solution so we have to apply some numerical technique.

(Refer Slide Time: 14:38)



So to solve them what we do is we simply take, start with some initial guess of the unknowns. That is we assume some values. We take some educated guess depending upon the problem we take some educated guess. Fortunately in our case for the power system analysis it is very easy to take this educated guess. So we really did not worry about our case that from where we would be getting this educated guess. For our case would be always we will be able to get this guess.

So what we do is that we will first take some educated guess x 1 nought, x 2 nought, x n nought. Then what we do? Then calculate then update. Now what we update? We update x 1 at the first iteration as function g 1 and we replace x 1 0, x 2 0, x n 0. When we try to update x 2 1 then we use this function g 2. Now in the first equation we have used x 1 0 because when we are trying to update x 1 we did not have any other value x 1.

But when we are trying to update x 2 we have already calculated x 1. So then therefore for g 2 instead of using x 1 0 we use x 1 1, the latest value available. But for x 2 we did not have as yet. For x 3 g 3. For x 1 and x 2 we have already got. But for x 3 we did not get as yet at this right hand side, we are trying to calculate x n 0. And … so then x i 1 would be g i x 1 1 … x i 1 then x i + 1 0. Please not that sorry x i 0 sorry x i 0. X 1 + 0 and x n 0.

And then … last x n 1 would be g n x 1 1, x 2 1…x n -1 1, x n 0. So that means everywhere we are simply taking the most updated value. And x 1 1 I mean this 1 basically stands for the iteration count. So this is the first iteration, first iteration, first iteration, first iteration, first iteration. So what we do? We first take an educated guess x 1 0, x 2 0, x n 0 and then we update and whenever you are updating all this variables we always use the most updated values whenever we are substituting this values.

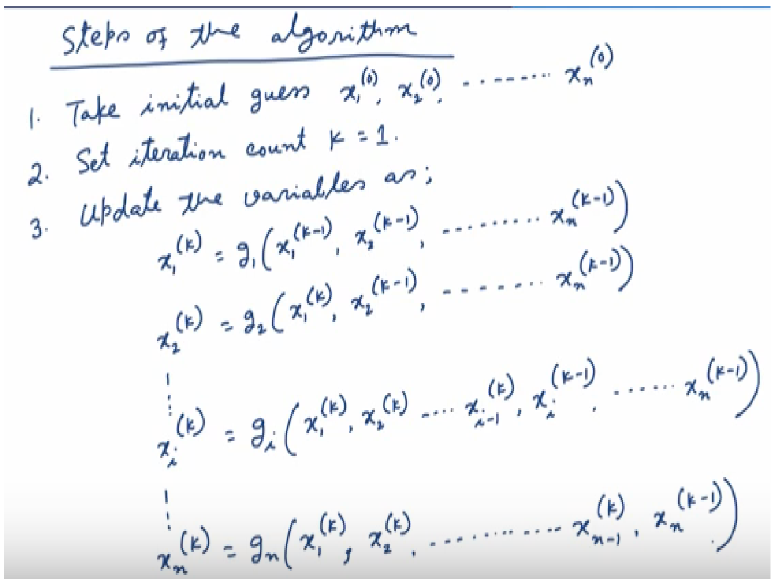
After we do this updation once then what we do we calculate error e i as x i 1 – x i 0. That is the magnitude for all i = 1 to n. So then therefore you would be getting n values of error e i. And because we are taking the magnitude whether this value is more than all this then this value e i would be always a positive quantity. Then we calculate maximum error max error as the maximum of e 1, e 2, e n.

If this max error is max error is less than some epsilon that is some threshold values then the algorithm converges. Otherwise we repeat the process. So now, so then what we do? We first take initial guess. We update, we calculate this error. We calculate the maximum error and if this maximum error is less than some threshold usually this threshold is taking 10 to the power -6 at least if this particular threshold value I mean if this maximum error is actually less than this threshold value we say that my algorithm is converged.

So that means these values are not really changing much. So then therefore they have converged. Otherwise, we simply repeat. So then in the second iteration what we do? We simply use a g 1, x 1 1, x 2 1 up to x n 1. X 2 2 that is in the second iteration that is we are trying to update the value of x 2 corresponding to the second iteration. That we will use g 2 into x 1 2 because here we have already calculated.

Here we have already calculated the updated value of x 1 corresponding to the I mean second iteration so x 1 2. But here you would be using x 2 1 then x n 1 and so on and so forth. So then therefore if we look into the algorithms so what would be the steps of the algorithm?

(Refer Slide Time: 21:13)

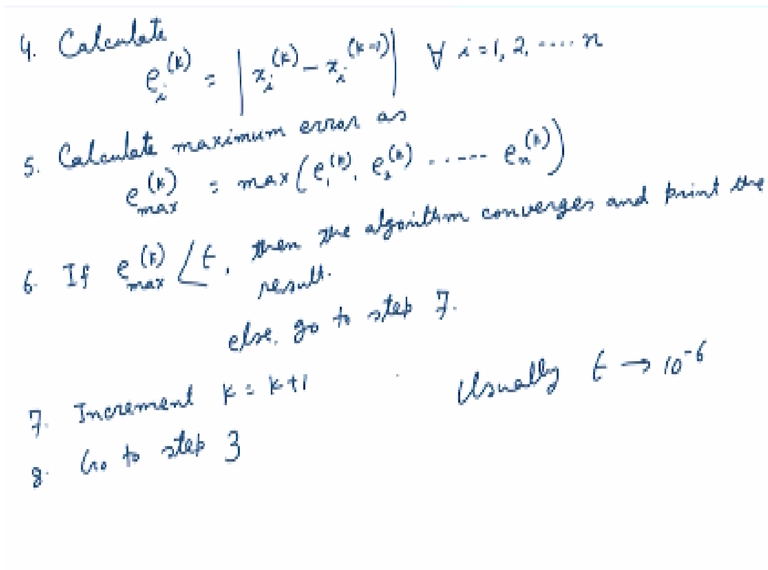


So the steps of the algorithm, steps of the algorithm would be take initial guess x 1 0, x 2 0 up to x n 0. Then we take a set iteration count K = 1. Then we update the variables as x 1 corresponding to kth iteration. Now here when you have started count k = 1. So here at this stage we have k have said been 1 so it is x 1 1. So it would be g 1 x 1. Now when we are calculating, when we are trying to update x 1 at kth iteration we still do not.

So we have only got the last value. So it would be corresponding to k – 1th iteration. Now because here k = 1 so it would be x 1 0. Then x 2 (k -1) … x n (k -1). X 2 corresponding to kth iteration it will be g 2. It would be x 1 k because we have already got this x 1 k here from this equation but for x 2 we have not we are trying to calculate. So we have got only this value right now and then … x n k -1.

X i ith it would be g 1 x 1 k, x 2 k… we would have by now solved or rather updated the up to i – 1th variable corresponding to this kth iteration right? But when we have to use the value of x i in this equation we still do not have anything so it would be x i corresponding to k -1th iteration that is the last iteration and …x n (k-1). And x n (k) g n (x 1 (k), x 2 (k)…x n-1 (k) because we have already updated up to n-1th variable in this iteration. But nth variable we have not yet updated, so we only note this. So we do this updation.

(Refer Slide Time: 25:24)



After we do this updation then we calculate e i (k) that is the kth corresponding to kth iteration as x i (k) – x i (k - 1). That is the absolute value of the difference between the present value of ith variable minus the value of ith variable corresponding to the last iteration for all i = 1 to n. So we do this calculation for all variables, all n variables. Then we calculate maximum error as e max (k) corresponding to the kth equation as max e 1 (k), e 2 (k), e n (k).

If e max (k) is less than some threshold then the algorithm converges and print the result. Else go to step 7. What we do in step 7, step 7 is increment k = k +1. So now we increment the iteration count and 8 go to step 3. So we calculate maximum error. If this maximum error is less than some threshold value our algorithm stops. If it is not less than this we then increment the iteration count and then we can start doing this same updation again and again.

So we do this. So then we simply repeat this calculation again and again such that and after some time if our initial guesses are good enough so then after some time this algorithm will converge, converge in a sense that there will be not much an difference between the calculated values of all the variables between 2 consecutive iterations. And if there is not much of a difference between the calculated values of any variable between 2 consecutive iterations we say that my algorithm has converged.

And we say that this calculation is finished and then we take the final print. Usually, epsilon is usually at least 10 to the power -6. So then therefore because epsilon is 10 to the power -6 so then therefore we are simply assuring here that the change in the variable if at all it occurs it will only occur only after 5 decimal point. So then therefore if any change is occurring only after 5 decimal point for any variable for all practical engineering purpose that value is constant.

So then therefore we can say that the algorithm converges and the final value of all the variables had been obtained. So with this basic knowledge of this basic procedure of the Gauss-Seidel iteration method, in the next class we will apply this method for the solution of our power flow equations. Thank you.