Computational Techniques

Prof. Dr. Niket Kaisare

Department of Chemical Engineering

Indian Institute of Technology, Madras

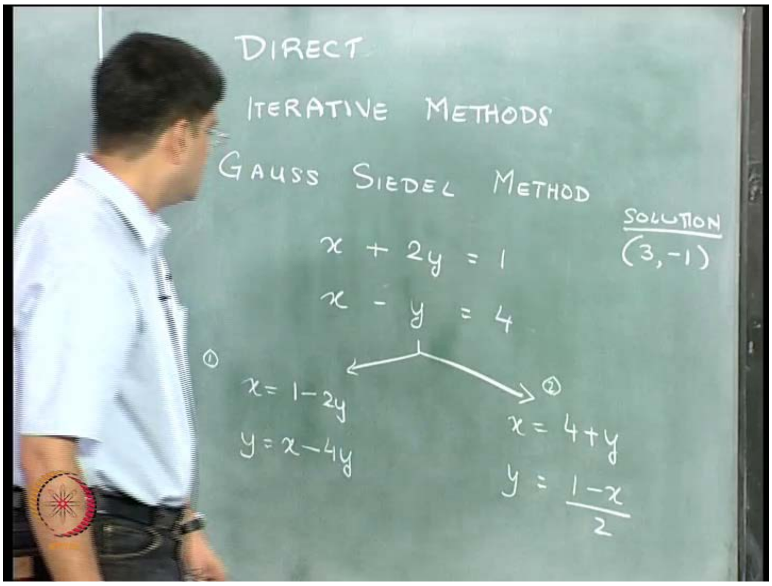
Module No. # 03

Lecture No. # 06

Linear Equations

Hello, so welcome again to module 3, where we were discussing different numerical ways of solving linear equations. In the previous lectures of the modules, we covered what is known as, the direct method for solving the linear equations.

(Refer Slide Time: 00:28)



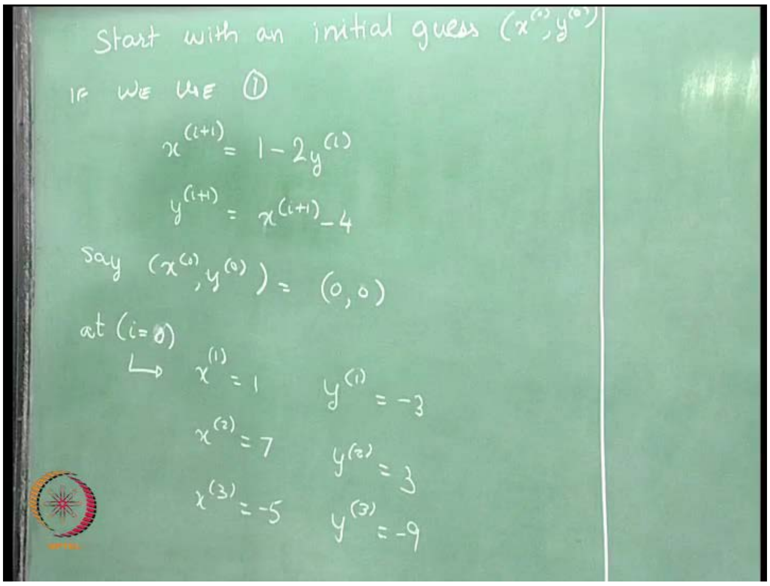
The direct methods that we considered included Gauss Elimination method, Gauss Jordan method, the LU decomposition method. And we also covered basically a tridiagonal matrix algorithm, which is specific to a special type of tridiagonal matrices. As opposed to this direct methods, the quality of the direct methods that we looked at essentially was that, when you get set of equations, and we solve these set of equations, provided we do not have any round-off errors, the results that we are going to get or going to be exactly accurate. So, that is… And the reason why they are called direct methods is because once you get the solution, you have the solution, we do not iterate over the solution, multiple number of times.

As oppose to the direct method, there is another class of methods known as iterative methods for solving linear equations; and we will basically be covering the iterative methods in this part of the lectures. So, the examples of iterative method, the first example that we will start off with is known as the Gauss Siedel method. And I will take relatively simple example of the two equations and two variables; and actually, I will show you how the Gauss Siedel method works; under certain situations, how the Gauss Siedel method does not work; what what we need to ensure in order to say that the Gauss Siedel method, we can get it to work; and we will look at essentially, how how to analyze the results that we get from the Gauss Siedel method.

So, the simple example that we are going to talk about again is basically going to be x plus 2y, let us say equal to 1, and x minus y equal to 4. So, the solution for these two equations, we know this (3,minus 1) that is the solution. And we can solve it fairly, in a fairly straightforward manner using Gauss elimination method, what we will see now is how to use the Gauss Siedel method. What we do in Gauss Siedel method is, rearrange this equation, so that we write it in terms of x equal to something to the right hand side; we rearrange this equation, so that it becomes y equal to something to the right hand side.

And two possible ways of rearranging this equation are fairly clear. So, you can take 2 y on to the other side, and you can write x equal to 1 minus 2y; and the second equation you can write this as y equal to x minus 4 y. So, what we have done is, we have used the first equation to express the overall result in terms of x, we have used the second equation in order to express the overall result in terms of y. So, that is one way of rearranging these equations. The second way of rearranging this equations is to use this equation to obtain x. So, we will write the second equation as x equal to 4 plus y, and the first equation, we will write that as y equal to 1 minus x divided by 2. So, the first step as we have said, we have done nothing but just to rearrange these equations, such that we have the unknown variables to the left hand side, and we have what we will for the time being considered as known variables to the right hand side.

(Refer Slide Time: 04:40)



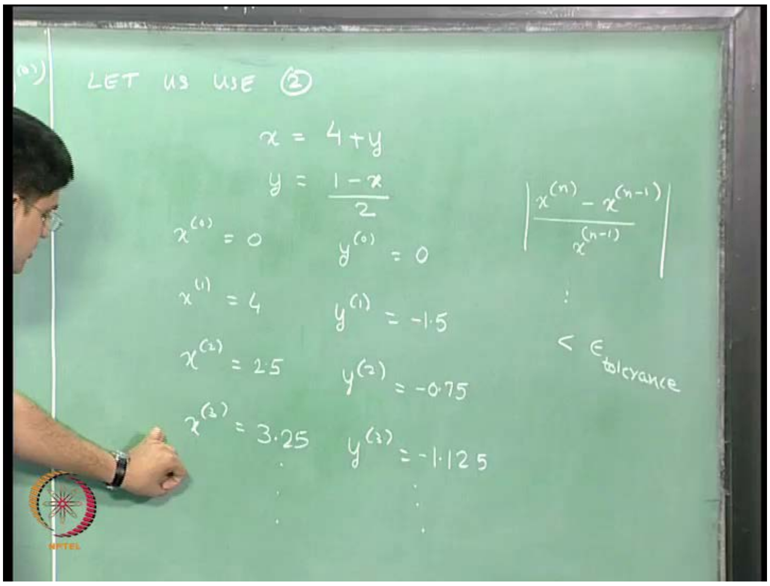
Now, it is an iterative procedure, so what we will do is, we will start with an initial guess; x superscript 0, y superscript 0 I represents superscript in the bracket, in order to say, this is the initial value, the 0 represents this is the initial value, and as we iterate multiple number of times, this particular coefficient will become x 1, y 1, x 2, y 2, x 3, y 3, so on and so forth. So, we use these rearranged equations as an update equation, in order to improve the value of x and y. So, let us look at if we use the first equation, in that case, we will write x i plus 1 equal to 1 minus 2y, the value at the i eth i eth equation. And then the second equation, we will write this as y i plus 1 equal to x i plus 1 minus 4, so these are the two equations, now we are we are we are going to use. Notice what we have done is in the equation for x, we have used the value, the previous value of y, y i in the equation of y we have use the newest value of x.

So, what we do essentially in Gauss Siedel iteration is used the latest values of x and y that we have computed through the iterative process. And we will start with some value, and let us say, we will just for convenience sake, we will start with origin. So, we start with origin, and use these equations in order to find out the new solutions of x. So, let us say, now what happens it at iteration number 1; or rather I should say at i equal to 0, what we get is x 1 equal to 1 minus 2 times y 0, which means basically our x 1 is going to be equal to 1, and we substitute that value 1 in this particular equation, and y 1 we are going to get is 1 minus 4 that is equal to minus 3.

And at this point of time, what what you can do is, just try to basically solve these equations in order to get x 2, x 3, x 4, x 5 and so on, and we will keep doing that. So, we will substitute the value y equal to minus 3 in this equation, so this becomes 1 plus 6, so x 2 become 7; y 2, when we substitute that value in this particular equation y 2, will get the value of 3. Now what we do is, we iterate on this again, x 3 is essentially going to be 1 minus 6 that will end up being minus 5, and essentially, y 3 will end up being minus 5 minus 4 that is going to be equal to minus 9.

And we continue doing this over and over again, and we will find that this particular solution does not converge to the desired the to the actual solution x equal to 3, and y equal to minus 1. So, we keep keep on repeating the process, and we will find basically that x is going to go to minus infinity or plus infinity, and y is also going to go quickly diverged to plus or minus infinity. So, that is what if we use, if we naively use the form one, in that is this particular equation is a recast in this form, and this particular equation is recast in this form. I actually made an error over here, so we will have y equal to x minus 4, and not x minus 4y. So, if this particular form is used, what we see is that we are not converging to the desired solution.

(Refer Slide Time: 09:26)



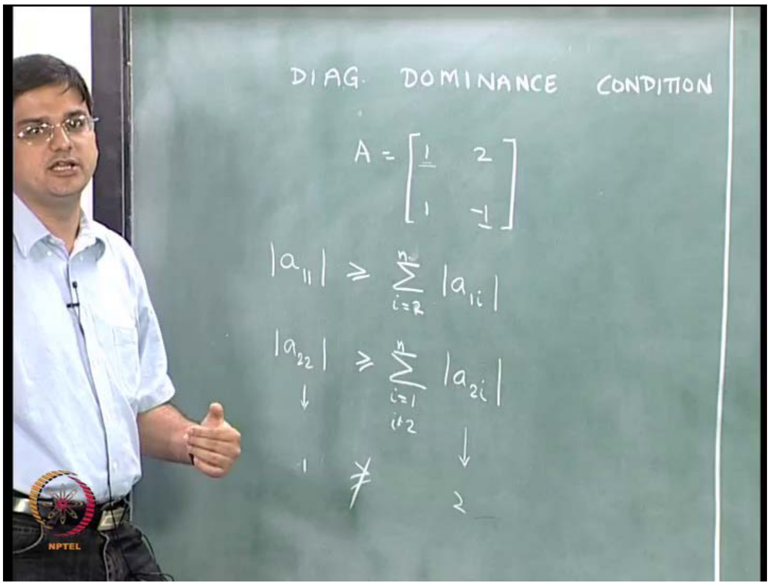
Let us use the second form of this equation; and the second form essentially was x equal to 4 plus y, and y was equal to 1 minus x divided by 2; we started off with x 0 equal to 0, y 0 equal to 0. So, we substitute y 0 in this particular equation, we will get x 1 equal to 4 plus 0, that is 4; y 1 is going to be 1 minus 4, that is negative 3 divided by 2, that is minus 1.5. Then we substitute minus 1.5 in this equation, and x 2, we will get x 2 as nothing but 4 minus 1.5, that is 2.5; y 2 is going to be nothing but 1 minus 2.5, that is minus 1.5 divided by 2, which is minus 0.75; x 3 continue this, we substitute y 2 into this equation, that is 4 minus 0.75, we will end up getting this as 3.25; and y 3 is going to be 1 minus 3.25, that is minus 2.25 divided by 2, that is minus 1.125. And we continue doing this until along enough time, and we will get our solution of x 3 equal to 3, and sorry and x n equal to 3, and y n equal to minus 1, where n we choose as large enough such that the results that we get, we say are converged.

So, what is the stopping criteria that we are going to use? The stopping criteria, you have keep in mind that essentially, we do not know a priori what the solution is; for this particular example, yes we do know that the solution is (3,minus 1), but in general, we do not know, what the actual solution is. So, what we have to do is, we have to compare these solution x n with x n minus 1 divided by x n minus 1, this was our approximation error, and we do that for all of the variables, and if this value turns out to be less than some tolerance value that we specify; at that particular condition, we say that our overall iterations have converged.

For example in this case, x 0 has changed from 0 to 4, that is the pretty large change that that we observe; in this case, it has changed from 4 to 2.5, so the change is actually 1.5 divided by 4. So, that is a pretty large percentage of change; almost 35 percent change, when you go from 4 to 2.5; likewise from 2.5, you are going to a 3.25 that is also a pretty large change; after a certain point we will see that x n minus 1 and x n, so on and so forth. They will start differing from each other with relatively small amounts, and when that difference falls to below epsilon tolerance at that time, we say that the overall Gauss Siedel iterations have converged.

So, the question is, when when do we use in the methodology number 1, and when do we use the methodology number 2, and we will not prove this for now, but we will keep that proof for a later time, we will do that proof in, perhaps in the fourth or third or fourth lecture in module module 4, where when we talk about non-linear systems, but what is required to ensure that these equations converge, the Gauss Siedel methods when we are applied on these equations will converge is what is known as the diagonal dominance condition.

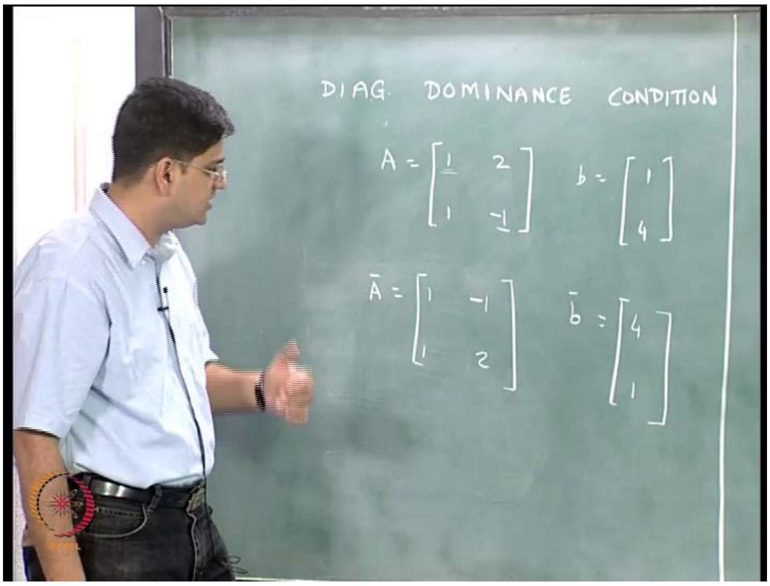
(Refer Slide Time: 13:41)



What that means, with diagonal dominance condition. So, if you look at this particular equation, and we put it in the form a x equal to b; our matrix A we can write this as 1 2 1 minus 1; of these again, these elements are the diagonal elements. Now, if the absolute value of the diagonal elements is greater than the sum of the absolute values of all the elements in that same column; in that particular case, we we say that the diagonal dominance condition is met. So, the diagonal dominance condition is going to be the absolute value of a 11 should be greater than or equal to sorry summation from 2 to n a 1i; in this particular case, the absolute value of 1 is indeed greater than or equal to the sum of the absolute value of the non diagonal elements. So, for this particular condition, this diagonal element is dominant, when we are looking at in the first column.

Also a 22 has to be greater than or equal to summation i equal to 1 to n i not equal to 2 a 2i; in this case, the absolute value of a 22 is 1, and the absolute values of… So, the now the absolute values in that particular column, if we if we actually look at this particular value is actually going to be equal to 2, and this condition is not met. So, what we get with the matrix A is that diagonal dominance condition is not met in in the case where the diagonal dominance condition is not met, we cannot guarantee the stability of the Gauss Siedel iteration. So, as we here increase the number of iterations, the Gauss Siedel iteration may not converge to the true value.

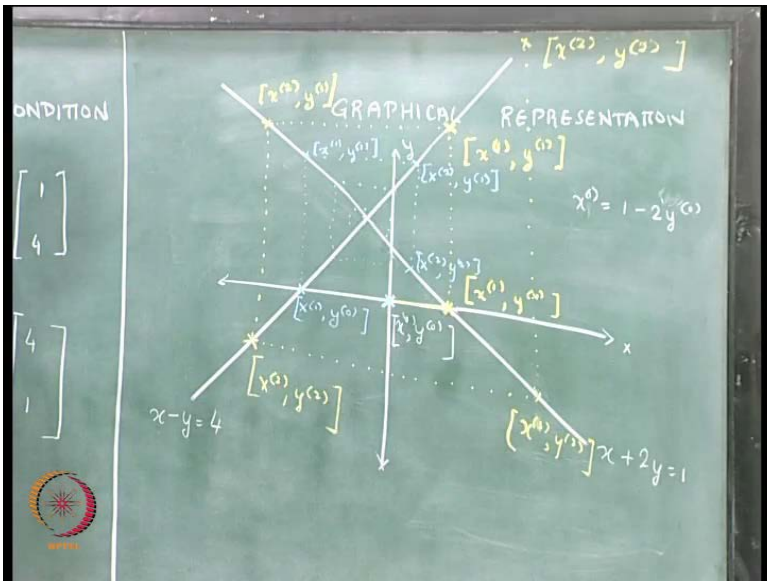
(Refer Slide Time: 16:20)



However, if the diagonal dominance condition is met, that means, instead of writing the equation in this form, we have swapped around this equation, and we have written this equations in the form A bar is 1 minus 1, that is this row, and the second row is 1 2, and of course, we have to change b bar also original b was 1 4. Now, that we have swapped these two equations the b bar is going to be 4 1. The solution of A bar x equal to b bar is the same as the solution of A x equal to A x equal to b, I will write down the b over here as well.

Now, in this condition, in this particular matrix form, what what we actually get is indeed this is the largest value in this particular diagonal, and this is all here in this particular column, and this value is also the largest value in this particular column. When we have in general an n by n matrix, we have to consider basically that the diagonal element, the absolute value of it should be greater than the sum of absolute values in the entire column for each and every diagonal element.

(Refer Slide Time: 17:26)



(No audio from 17:26 to 17:38)

So, we will draw a two axis - x and y, and the other line is x minus y equal to 4. So, that particular curve is perhaps going to be represented in this form, and will just extend it beyond. So, we will call this equation as x plus 2y equal to 1, and this particular equation is x minus y equal to 4. This point is essentially our (x 0, y 0). So, the first equation what we used was, we used x x 1 equal to 1 minus 2y, x 1 we said was equal to 1 minus 2y sorry 1 minus 2y 0. So, what what does 1 minus 2y 0 represent? 1 minus 2 y 0 represents a point on this particular line, we are going to keep the same value y 0, y 0 is going to be 0, so we are moving along this line to represent y 0, and we will reach essentially this particular point; this particular point is indeed (1,0) that is because it satisfies the equation of this particular curve. So, what we ended up doing is the first iterative step, we went from the point x 0, y 0 to the point x 1, y 0. So, x 1 is 1, y 0 equal to 0. So, this point is x 1, y 0.

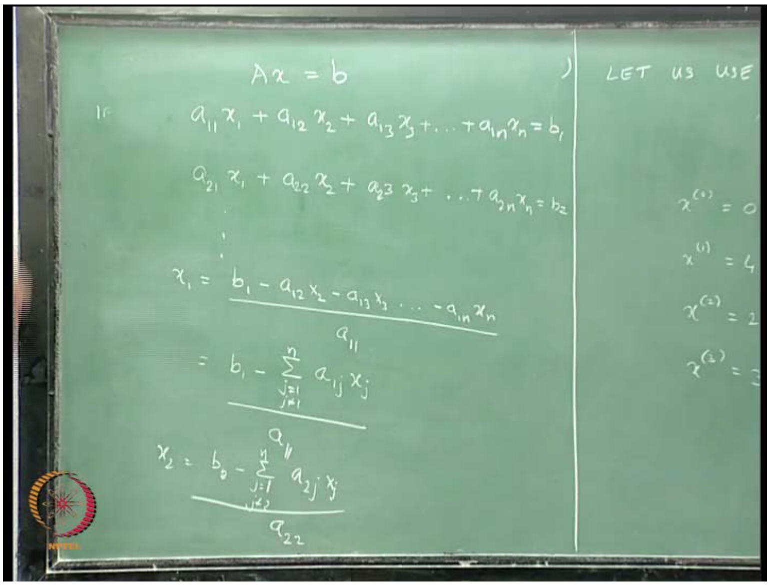
Then we have used the second equation, so that we had y 1 equal to x 1 minus 4; y 1 equal to x 1 minus 4, so we are keeping the same x 1, so we are going along this line, and we will essentially reach basically, this particular line; and we have now reached this particular point is essentially x 1, y 1. Why is it x 1, y 1? Because we know that the point x 1, y 1 lies on the line x minus y equal to 4, that is the way, that is the method in which we obtain the value of y 1; when the x 1 remains constant, we are going along this same vertical line; so this indeed is the point x 1, y 1.

We use this particular equation to get y 2; now y has to get sorry x 2. So, we will keep the same value of y 1, and we will go along; so along this horizontal line, the y 1 value remains the same. And the point where that horizontal line intersects this particular line that is where we will get the value of x 2. So, this particular point is going to be x 2, y 1. And now we keep the value of x 2 the same that means, we go along the vertical line, and find where this particular curve intersects; this the vertical line intersects this particular curve, and when we project will essentially reach over here, this point is nothing but x 2, y 2, we will keep the y 2 same, and we will reach this point x 3, y 2, and when we project along this point we get this x 3, y 3. We are starting from the origin, and we keep going in circles and that circle or that spiral is essentially, expanding spiral. So, the method that we are going to use if x equal to x plus 2 y is our first equation, and x minus y equal to 4 is the second equation. We are not going to reach the desired solution that is this particular point.

Now, an alternate way how we how we went about about this is we used this particular curve to get x, and we use this particular curve essentially, to get y. And how that is going to go is we again we start at the origin. The x-axis is the axis representing the curve curve where y 0 equal to 0. So, we will use this particular equation now in order to get x 1. So, the x 1 is going to be the point, where this horizontal line intersects this particular axis. So, this is going to be our x 1, y 0. Now, we project along this particular vertical vertical line; the vertical line again to remind at represents the line at which x equal to x 1, and where this intersects this line is y 1, so this point becomes x 1, y 1; this becomes x 2, y 1; this point becomes x 2, y 2; and as we keep iterating, we will spiraling towards desired solution.

So, if we are going to use this particular curve as the curve to get x, and this curve took as the curve of to get y, what happens is we starts spiraling essentially towards the center or to of the intersection of these two lines. So, this starts attracting essentially, all the trajectories of the Gauss Siedel iteration, if we ensure that our system of equations is indeed diagonal diagonally dominant. So, this is the graphical representation of how the Gauss Siedel iterations essentially work.

(Refer Slide Time: 24:59)

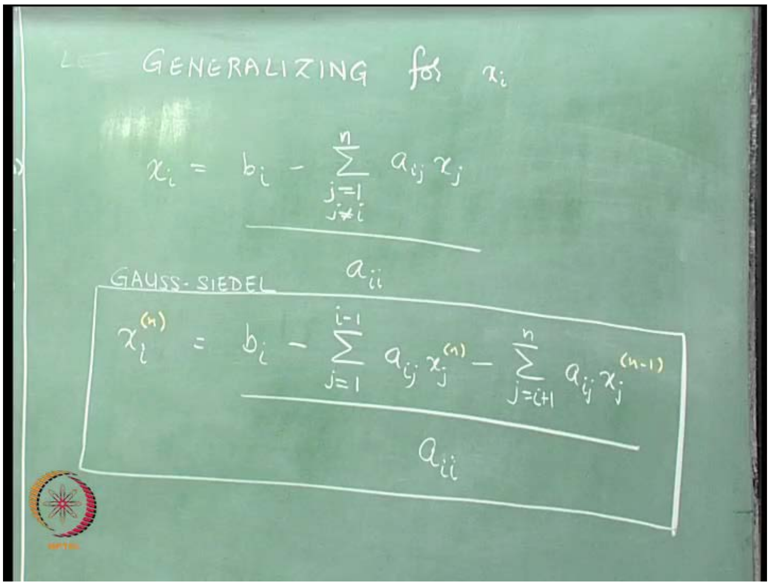


We have the equation A x equal to b, and we expand that equation we can write this as a 1 1 x 1 plus a 1 2 x 2 plus a 1 3 x 3…

(Non audio from 25:15 to 25:36)

And so on. So, we will essentially have n equations of this type. The first equation we will use in order to get, so this is, let us assume that we have gotten this equation after rearranging, such that the diagonal dominance condition is met. If the diagonal dominance condition is met, then will just go ahead and use the first equation to get x 1, the second equation to get x 2 and so on. So, we will take all these terms on to the right hand side and divide throughout by a 11. So, what we get is, x 1 equal to b 1 minus a 12 x 2 minus a 13 x 3 a 1n x n divided by a 11; this we can write it down as b 1 minus summation j equal to 1 to n, j not equal to 1 a 1 j x j divided by a 11. This is how we have we have written our our x 1 as… Like wise we can write our x 2 as b 1 b 2 minus summation j equal to 1 to n j not equal to 2 a 2j x j whole divided by a 22 and so on we will be able to write for each of the x i(s).

(Refer Slide Time: 27:26)



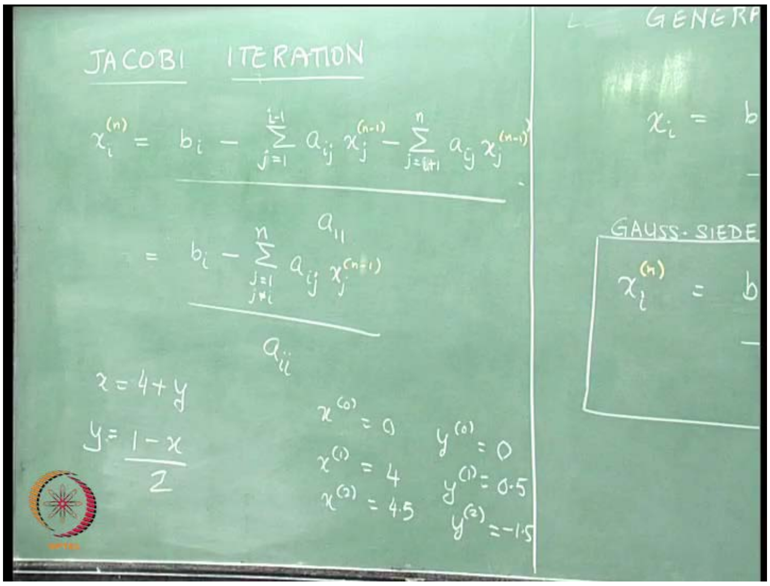
(No audio from 27:27 to 27:37)

I am sorry generalizing for x i, our x i is going to be written as b i; keep in mind, we have following the same pattern, when we go from x 1 to x 2 to x 3, we change from b 1 to b 2 to b 3 and so on. So, b i minus summation from j equal to 1 to n j not equal to i; j equal to 1 to n j not equal to i a i, j x j a i j times x j divided by a i i. This is just the diagonal term of the i eth equation, this is the right hand side of the i eth equation, and to the left hand side, we have taken all the non-diagonal terms in the i eth equation. And then we will go ahead, and solve this equation in order to get a 1 a 1 a 2 a 3 up to a n; and will keep repeating this until we get the desired convergence.

This equation of course, we can also will be able to write it as b i minus summation of j equal to 1 to i minus 1 a ij x j minus summation j equal to i plus 1 to n a ij x j divided by a ii, that is going to be our x i; what we will use to get x i value is the all the old values for the x i in the sub diagonal elements, and all the new values of x i in the super diagonal elements. So, x i from the at the iteration value n, we are going to get that equal to b i a ij x j n minus a ij x j n minus 1 represents the Gauss Siedel method.

So, the steps that we take in Gauss Siedel method is first to see if the equations can be put in diagonally dominant form; once the equations are put in the diagonally dominant form, decide on initial values of x x 0 y 0 and so on and so forth for all the variables; and then run this iterations step using this equation for i equal to 1 2 the to all the values of x that we have. And we keep doing this iteration multiple times, so that we get the solution to converge to the predefined tolerance value. So, this is the Gauss Siedel method.

(Refer Slide Time: 31:07)



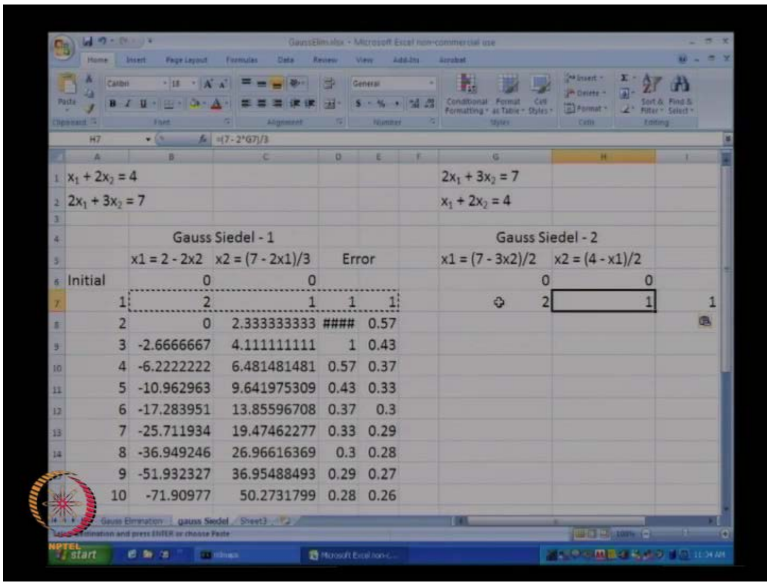
In Gauss iteration, we just keep the latest values of x 1 x 2 up to x n; in the Jacobi iteration, we do not keep only the latest value, we also keep the previous value, and we iterate based on only the previous values of x or not the most current values of x. So, the equation that we had for Gauss Siedel, we will just change it so, we have x i equal to b i minus summation i equal to 1 to n minus 1 a sorry j i equal to j to n minus 1 a ij x j minus summation j equal to i plus 1 to n. And what we had in Gauss Siedel iteration is this had an index n whereas, this had an index n minus 1.

The difference between Gauss Siedel iteration and the Jacobi iteration is both of them have the index n minus 1; which basically means that we are not going to use the latest values for the previous variables, but if we are going to use the values from the previous iteration or a more compact way of writing this is b i minus summation j equal to 1 j not equal to i up to n a ij x j this should be i minus 1 not n minus 1; and this essentially is the Jacobi iteration. And to go back to the previous example of x minus 4 x minus y equal 4 and x plus 2 y equal to 1, we will write those equations as x. So, what we did was we started with x 0 equal to 0, y 0 equal to 0, we substitute the value of y 0 over here and get x 1 - 4.

What we did in Gauss Siedel method is then substitute this value of 4 in this equation, and we got 1 minus 4 divided by 2 equal to minus 1.5. But what we do in the Jacobi iteration is instead use the x 0 and y 0 values itself in these equations, so we do not use the latest value, we use actually the previous value x 0 in this equations, so that y 1 we are going to get equal to 0.5. Now, we substitute the value of 0.5 in this equation to get the value of x, we substitute the value of 4 in this particular equation to get the value of y. So, our x 2, we will get it as 4.5; our y 2 that we will get we will get is essentially going to be 1 minus 4, that is minus 3 minus 3 divided by 2 and that is minus 1.5.

And we keep repeating this process over and over again until we reach convergence. What we will see actually is that the conditions for convergence of Gauss Siedel method as well as for the Jacobi iteration method remains the same only the procedure in which we do the Gauss Siedel method verses the procedure, and which we do the Jacobi iterations differ from each other in there details. So, that is about the Gauss Siedel iteration and the Jacobi iteration.

(Refer Slide Time: 35:45)



Now, let us take our Microsoft excel and we will use Microsoft excel to solve the two equation and two unknowns problem using the Gauss Siedel method; what I will do first is, I will use the Gauss Siedel method in the form that we have the original equations in that means, equation number 1, I will use it to obtain the value of x 1, and equation number 2, I will use it to obtain value of x 2. As we have said, there was nothing special about calling this particular equation as equation 1 and this equation as equation 2. So, what we have done is, we have interchange these two equations, and even after interchanging these equations, what we will end up with is the same point is going to be the point of intersection of the two lines.

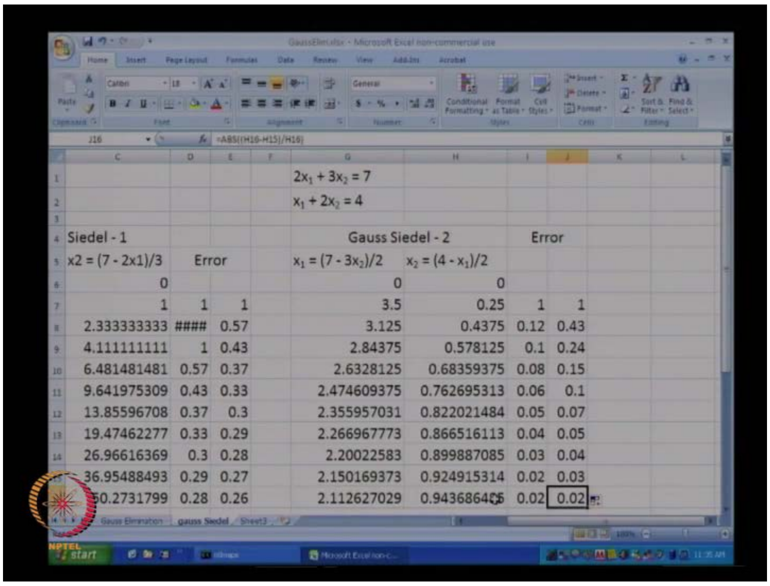
And what we will do in both these methods is that we will start these methods from the origin; so initial guess is x 1 is going to be equal to 0, x 2 is going to be equal to 0. So, because it is a Gauss Siedel method, what we do the at at the second iteration, our x 1 value is going to be equal to 2 minus 2 multiplied by the x 2 value from the previous iteration that is how we do the Gauss Siedel method, because in the Gauss Siedel method, the newest values are essentially going to be used in order to compute the next value of x 1 or x 2.

And x 2, for x 2, what we will do is x 2 is going to be equal to 7 minus 2 times x 1; in case of a Jacobi iteration, we will always use the previous iterant value in order to compute the new value; however, in the Gauss Siedel method, we will use the newest value. So, x 1 the previous value was 0, but in the second iteration, we have replaced the value of 0 with the value of 2. So, that is the value that we are going to use, and this whole thing is going to be divided by 3, and that is going to be our new guess sorry the new solution value that we will get from this particular equation.

And we can then keep continuing this downwards and this, and let us considering this for 10 iterations, this should be iteration 1, this one initial is the 0 eth iteration, so this should be iteration 1 and 2. Just drag this see what have done is really clicked on the right edge, and then just dragging it down, and we have 10 iterations. And at each time what we also need to do is compute the relative error; and the relative error is going to be nothing but the absolute value of the difference between the current and the previous value divided by the current value. And this is the error in… So, that is the error in x 1, and likewise we will have error in x 2 is well; and for error in x 2, we are just going to drag this to the left hand side.

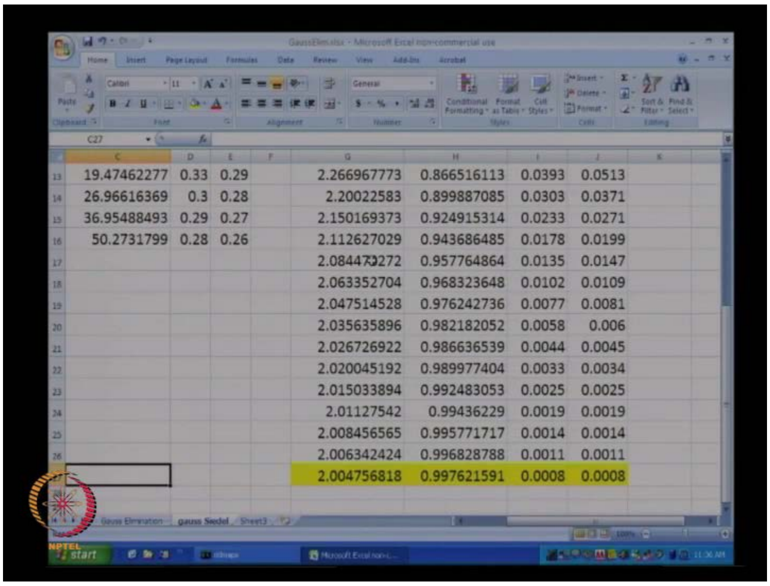
So, this is the error in x 1, this is the error in x 2, we will click f 2 and just confirm that this is what it ought to be; and we will just drag this and see how the error is changing. And when the error actually becomes less than a predefined tolerance value that is the time when we stop our iteration. So, what happens over here is that the error does not keep decreasing with the number of iterations, and what we are actually seeing over here is that the one of the values x 1 is going very negative, whereas x 2 is becoming a very large positive value. So, clearly what we see is that the solution is diverging; this is what we expected based on the analysis that we had done earlier we will do is we will use the two equations, but the 2x 1 plus 3x 2 equation we are calling this equation 1, and what was equation number 1 previously we are calling this as equation number 2.

(Refer Slide Time: 41:04)



And we will again have initial values as 0, we will start from the same initial values; and we will indeed continue in a very similar manner as before… Error. However we would not be using the same expression as before, instead we will be using 7 minus 3 times x 2 divided by 2, as I means to calculate x 1; format cells subscript; again this is in order to avoid confusion. And f our x 2 is going to be equal to 4 minus x 1, and again the x 1 is going to be the new value x 1 divided by 2. So, this, so from 0 our x 1 value has gone to 3.5, from 0 our x 2 value has going to 0.25; we will just select this and drag it to see where the solutions are leading us. If we recall our solutions, where 2, 1, so we are getting closer and closer to the solution, and we will need a few more iterations in order to reduce the error to the level that we desire; we need 10 to the power minus 3 at least as the accuracy.

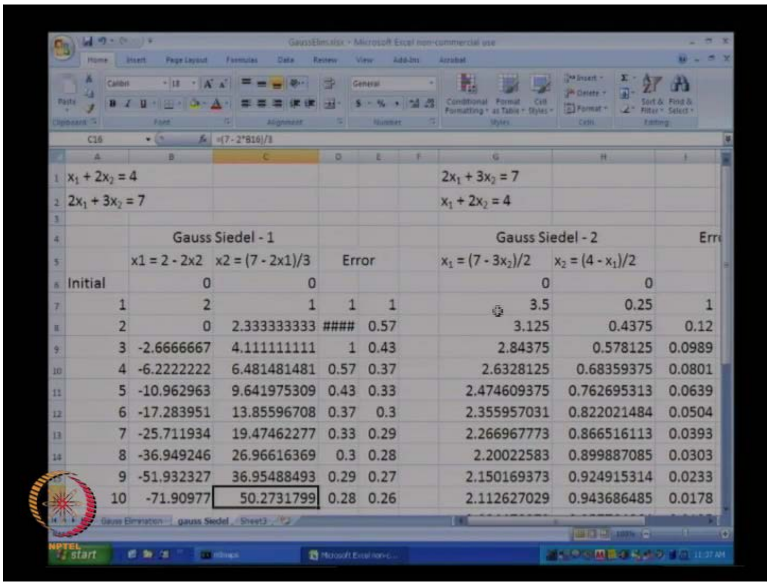
(Refer Slide Time: 42:41)



	C	D	E	F	G	H	I	J
13	19.47462277	0.33	0.29		2.266967773	0.866516113	0.0393	0.0513
14	26.96616369	0.3	0.28		2.20022583	0.899887085	0.0303	0.0371
15	36.95488493	0.29	0.27		2.150169373	0.924915314	0.0233	0.0271
16	50.2731799	0.28	0.26		2.112627029	0.943686485	0.0178	0.0199
17					2.0844[illegible]272	0.957764864	0.0135	0.0147
18					2.063352704	0.968323648	0.0102	0.0109
19					2.047514528	0.976242736	0.0077	0.0081
20					2.035635896	0.982182052	0.0058	0.006
21					2.026726922	0.986636539	0.0044	0.0045
22					2.020045192	0.989977404	0.0033	0.0034
23					2.015033894	0.992483053	0.0025	0.0025
24					2.01127542	0.99436229	0.0019	0.0019
25					2.008456565	0.995771717	0.0014	0.0014
26					2.006342424	0.996828788	0.0011	0.0011
27					2.004756818	0.997621591	0.0008	0.0008

So, let us go and do this little bit more, may be I will just expand this cells a little bit further. So, our accuracy is not yet met, we are looking for our the error epsilons to go below a tolerance value of 10 to the power minus 3; that is when we will stop our iterations. This is when the solution the error value has gone below 10 to the power minus 3, and this is what we will take as an approximate solution from the iterative method.

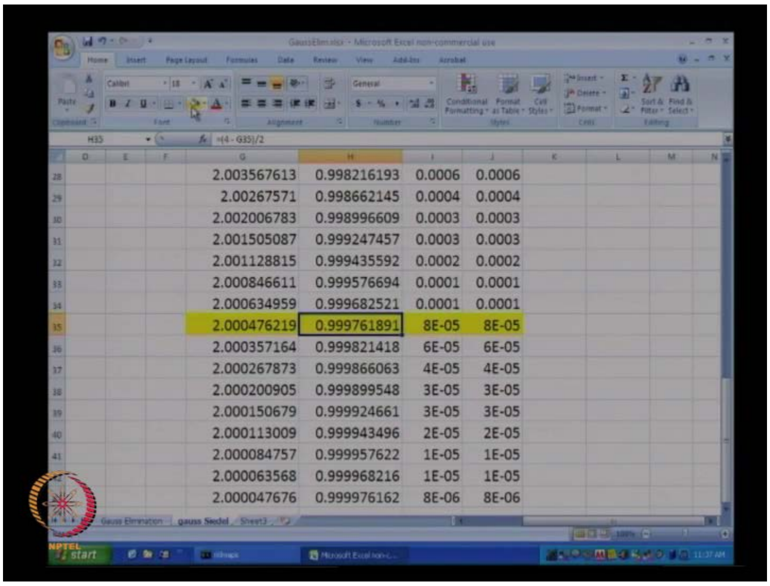
(Refer Slide Time: 43:30)



	A	B	C	D	E	G	H	I
1	$x_1 + 2x_2 = 4$					$2x_1 + 3x_2 = 7$		
2	$2x_1 + 3x_2 = 7$					$x_1 + 2x_2 = 4$		
3								
4		Gauss Siedel - 1				Gauss Siedel - 2		Err
5		x1 = 2 - 2x2	x2 = (7 - 2x1)/3	Error		$x_1 = (7 - 3x_2)/2$	$x_2 = (4 - x_1)/2$	
6	Initial	0	0			0	0	
7	1	2	1	1	1	3.5	0.25	1
8	2	0	2.333333333	####	0.57	3.125	0.4375	0.12
9	3	-2.6666667	4.111111111	1	0.43	2.84375	0.578125	0.0989
10	4	-6.2222222	6.481481481	0.57	0.37	2.6328125	0.68359375	0.0801
11	5	-10.962963	9.641975309	0.43	0.33	2.474609375	0.762695313	0.0639
12	6	-17.283951	13.85596708	0.37	0.3	2.355957031	0.822021484	0.0504
13	7	-25.711934	19.47462277	0.33	0.29	2.266967773	0.866516113	0.0393
14	8	-36.949246	26.96616369	0.3	0.28	2.20022583	0.899887085	0.0303
15	9	-51.932327	36.95488493	0.29	0.27	2.150169373	0.924915314	0.0233
16	10	-71.90977	50.2731799	0.28	0.26	2.112627029	0.943686485	0.0178

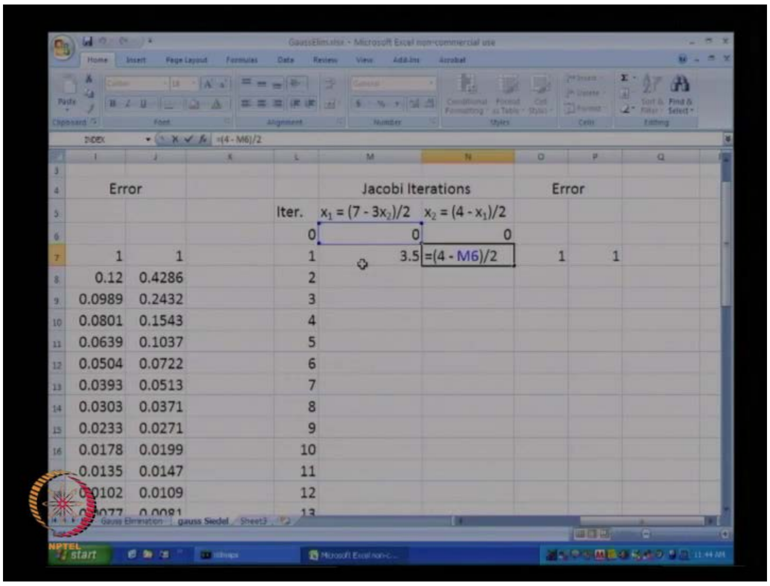
So, to recap what we have done so far, we took two equations and two unknowns, we wrote them down in this particular form as well as in this form, the difference over here is that the equation 1 in this case is taken as equation number 2 in this case; equation 2 in the first case is taken as equation number 1 in this particular case. What we saw is that when the in the first example the diagonal dominance condition was not met, as a result our solution diverged; whereas in the second example, the diagonal dominance condition was met and what happens is that the indeed finally, reach the desired solution. The solution is accurate to to the accuracy that we have asked our solver, if we say we need a greater accuracy, we need to go ahead and do a few more iterations, and I can show you the results.

(Refer Slide Time: 44:30)



And if let us say we we were to ask for an accuracy of 10 to the power minus 4, this is this solution that we will get. And let us see the number of iterations that are required, so this is the initial guess; so that is 0 eth iteration; this is the first iteration. And I will just drag this, and we require 29 iterations if the error tolerance is 10 to the power minus 4, and if the error tolerance is 10 to the power minus 3, we require 21 iterations.

(Refer Slide Time: 45: 22)



And just the way we did for the Gauss Siedel method, we will do a, we will solve Jacobi method in a similar manner. I will change the font size. So, we have iteration number x 1 and x 2, we will compute in the same manner, and then errors again we will compute in the same manner. So, what again what I am going to do is, I will go back and copy paste this rather than having to redo it myself again, because nothing really changes over here. And I am getting this hash signs, because the error I am defining as the difference between the two divided by this particular value and this value, because no value is specified over here is taken as 0. So, this is equation 0 divided by 0 as a result of which we get hashes.

As soon as we populate these results, we will not get hashes, and it will be the replaced by the appropriate numbers over there. So, we will have number of iterations, and I will just drag to say 15 iterations; and we start with x 1 equal to 0 and x 2 equal to 0. x 1 the way we calculate does not change, because x 1 is the newest value all the time. So, the calculation of x 1 does not change. The calculation of x 2 is going to change. So, I will copy paste it, and then show you how we change the way, we compute x 2 using the Jacobi iteration.

So, this is, so the pasted value is for the Gauss seidel method, it is not for the good Jacobi method. So, in Gauss Siedel method, we have 4 minus the latest value of x 1 divided by 2, this is not how the Jacobi iteration works. In Jacobi iteration, we do not use any of the values in this particular row; we will use only the values in the upper row. So, what I will do is, I will just take this particular cell, look where my cursor is, my cursor is at edge of the cell, it changes from we know a plus sign like a white cross type of a sign to a cross arrows type of a sign over here.

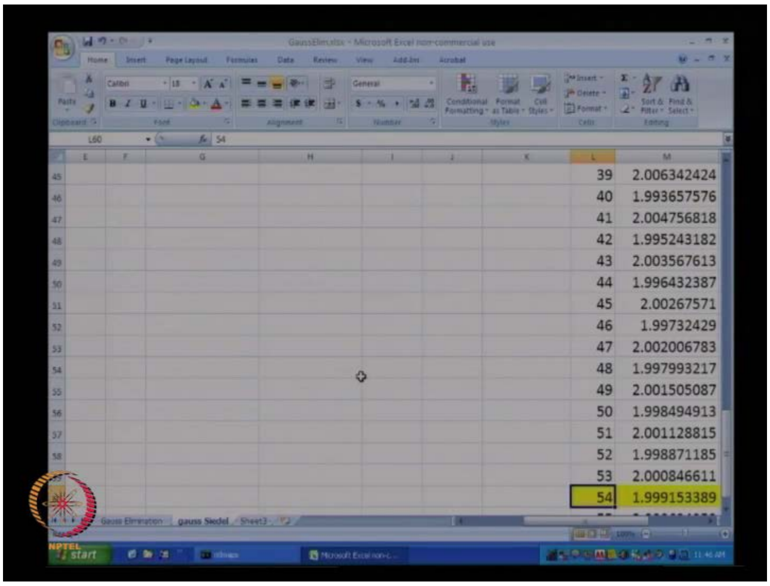
When it changes to this type of an arrow, this type of an arrow this type of a sign the cursor, I will click the left key with the left mouse key depressed that particular plus arrow sign has disappeared. Now I can move this anywhere. So, as I move this my mouse along you can see this blue color square moving; and as the blue star color square is moving, this particular formula is changing. So, I will move this blue color square from the new value of x 1 to the old value of x 1, and that should give me the Jacobi iteration, and I will click enter, and this is the result from the Jacobi iteration.

So, to recap what we do in Gauss seidel method is we use only the latest value; in Jacobi iteration, we use only the values from the previous iteration and not the latest value. So, in this particular case, the value from previous iteration is 0 and 0. As a result for the for the Jacobi iteration, we compute 4 minus m 4 m sorry 4 minus m 6, and not 4 minus m 7 as was done in the Gauss Siedel method. We will just highlight this entire row, and drag it for 15 iterations and see what we are getting. In 15 iterations, our solution has not yet converged; let us go to say 40 iterations as before.

And I will I will select this particular row; and instead of dragging the row downwards, if the row adjoins another row, which at the column adjoins another column, which is full, we can actually just double click at the right handed the Excel will fill itself. So, I have double clicked over here, and excel has fill the values; and we want our overall error to be 10 to the power minus 3 at least. So we need to drag this a little bit further. We are still not reached yes.

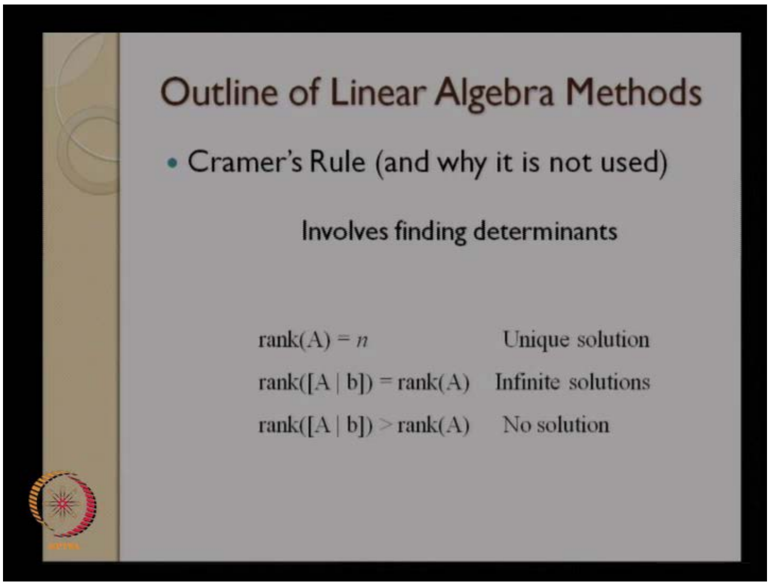
So, finally, after 54 iterations we have reached the solution; whereas, in the Gauss Siedel method, we reach we have reached the solution in only 21 iterations, the reason why this happens in the Gauss Siedel method against the Jacobi method is because the Gauss Siedel method uses the latest values of the variables; whereas, the Jacobi iteration uses the previous iterate values of the variables.

(Refer Slide Time: 51:05)



However, both these methods have linear rate of convergence, so although in this particular example, we were able to see such a big difference in the number of iterations required for the Jacobi iteration to converge compared to the Gauss Siedel iteration. In general experience has been that the Jacobi iteration is able to converge fairly rapidly, I mean as rapidly as a typical Gauss Siedel iteration. In general, I find that using a Gauss Siedel method to be a more preferred method; and the reason for this is that when we are solving this overall equation by the Gauss Siedel method, we only need to keep in n number of values, when we have n variables. So, every time we recomputed the variable, we do not need to store the previous value.

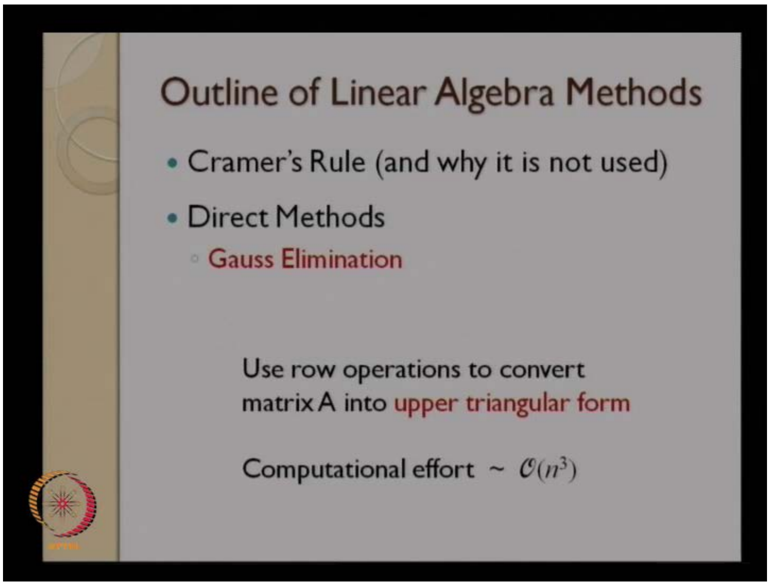
(Refer Slide Time: 52:14)



So, what we have done in today's lecture - the lecture 6 of module 3 is to look at the iterative methods for solving linear equation a of the type A x equal to b. And now what I will do is use the next few minutes to summarize what we have done in this module. In this particular module that is module 3, we have primarily covered computational methods for solving linear equations, linear equations of the type A x equal to b.

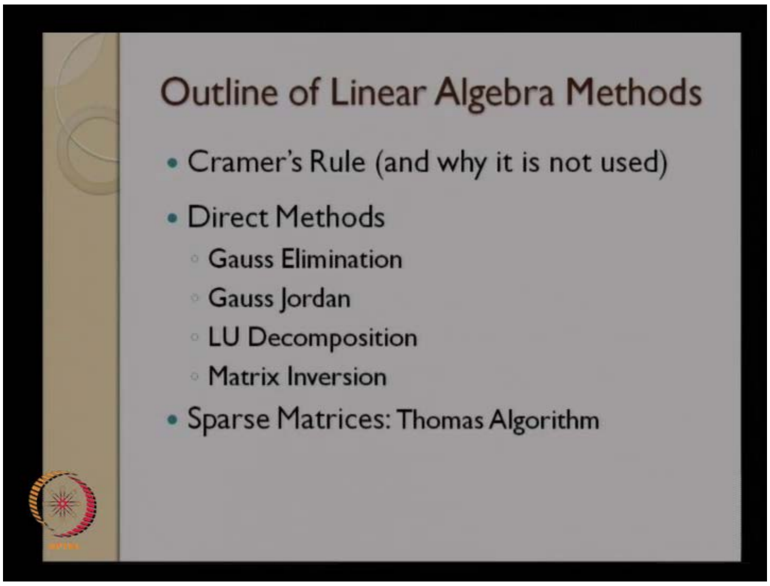
The first thing that we started off with was Cramer's rule; Cramer's rule involves finding determinants, and finding determinants is extremely computationally complex. As a result Cramer's rule is not used beyond say the 4 or 5 dimensional systems. Using this Cramer's rule idea, we motivated a few things further we said that if rank of the matrix A in A x equal to b equal to is equal to n, where n is the size of the vector x. Then we will get a unique solution. If rank of the matrix is not equal to n, if the rank of matrix A is less than n, then we need to check the rank of matrix A b, and the rank of matrix A; if both these ranks are equal, then we have infinite number of solution; if rank of this guy is greater than rank of A, is going to be no solution. And then we gave a geometric interpretation of this also, in terms of vector spaces.

(Refer Slide Time: 53:11)



Next, we covered at the direct method, the first method are that we covered very extensively was the Gauss elimination method. The idea of Gauss elimination is to use row operations to convert in the matrix A into an upper triangular form. What we also saw in the previous lecture of this module is that a computational effort required in Gauss elimination is of the order of n cube. The next method we talked about was Gauss Jordan method; in Gauss Jordan method, we use row operations to convert the matrix A, not into upper triangular matrix, but into an identity matrix. And whatever is left of the matrix b in the just suppose matrix A b is going to be the solution x.

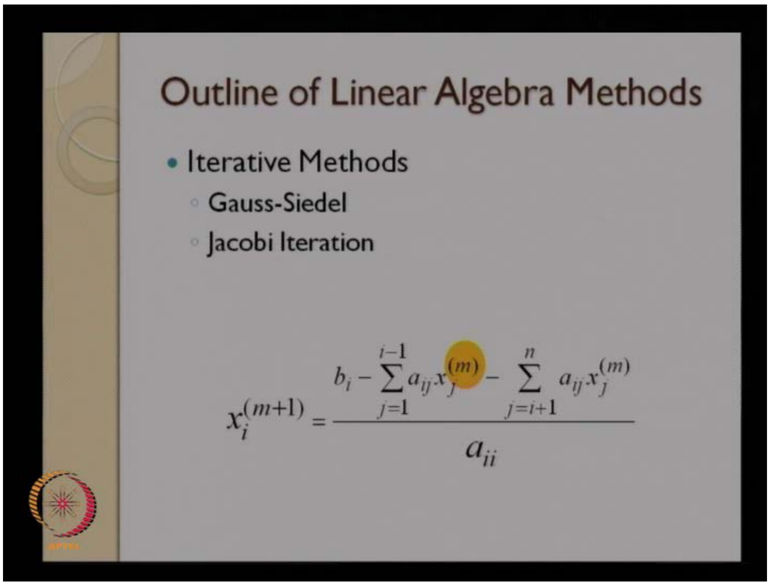
(Refer Slide Time: 53:59)



We then also talked about the LU decomposition method. The first part of LU decomposition is exactly the same as Gauss elimination. We use Gauss elimination to get the upper triangular matrix; and all the coefficients that are used to in each pivot row is are going to actually form the lower triangular matrix in the LU decomposition method. Then in towards the end of the lecture 5 in this module, we covered sparse matrices specifically, we covered tridiagonal matrix, and that were to solve the tridiagonal matrix we covered an algorithm called the Thomas algorithm or the tridiagonal matrix algorithm.

The Thomas algorithm is similar to Gauss elimination, but it exploits the special structure of the sparse matrix, because of which the overall effort required in computing the solution using Thomas algorithm is of the order of n to the power 1 rather than it being of the order of n to the power 3. So, that was what we discuss about the Thomas algorithm or the TDMA.

(Refer Slide Time: 55:05)



And finally, in this particular lecture, we covered iterative methods and the geometric interpretation of the iterative methods. The first iterative method we covered was the Gauss Siedel method, and this was the overall expression for Gauss Siedel method. What we did was for each equation, we took all the elements from the left hand side overall to the right hand side, and used whatever is left as an iterative equation. So, we started with some initial guess of x values, and we keep kept improving the guess by iterating it over this particular equation; this is an equation we saw of few moments ago.

In Gauss Siedel, we use the most recent values of x that means for computing x i, we use the m plus 1 eth values of a x 1 up to x i minus 1; and we use the m eth value from x i plus 1 up to n, because we do not have the m plus 1 eth values of x j at this point. So, we used the latest value in computing x i in Gauss Siedel method, whereas in Jacobi iteration method, we do not use the latest value, but we use the value values from the previous iteration; and the change between Gauss Siedel and the Jacobi iteration have just highlighted over here; in Gauss Siedel, we got m plus 1 as the iteration number over here whereas, in Jacobi iteration we get this iteration number m at itself.

So, these were the two main classes of methods; the direct methods and the iterative methods, when solving the linear algebra. So with that, we come an come to an end to module 6 sorry with that we come to an end to module 3, and from next lecture onwards, we will start with module 4, which will cover solving non-linear algebraic equations.