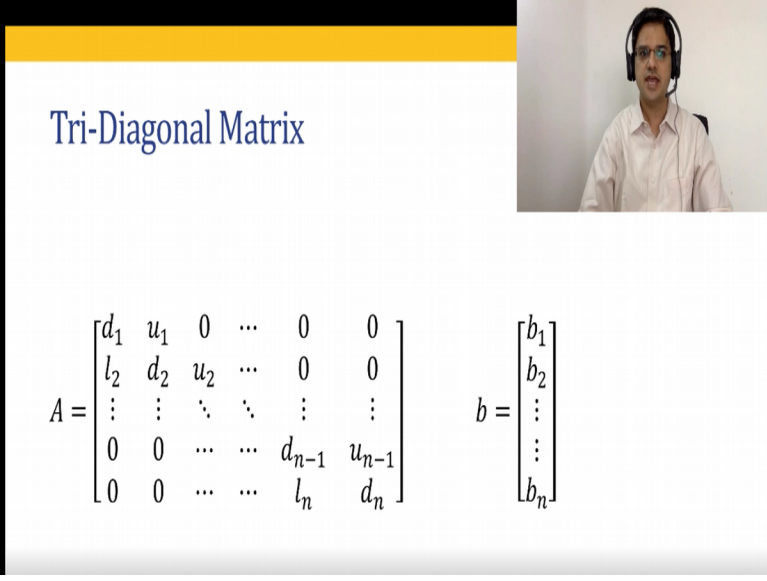


MATLAB Programming for Numerical Computation
Dr Niket Kaisare
Department of Chemical Engineering
Indian Institute of Technology, Madras

Module No. #04
Lecture No. #4.5
Linea Equations - Tri-Diagonal Matrix Algorithm

Hello and welcome to MATLAB programming for numerical computations. We are in module number 4, and this is the last lecture of this module. In module 4 we have been covering linear equations of the type $A(x) = b$ where, A is an n/n square matrix and b is an $n/1$ column vector. In this lecture, we are going to cover, a special type of algorithm known as the Thomas algorithm or Tri-Diagonal matrix algorithm.

(Refer Slide Time: 00:43)



Tri-Diagonal Matrix

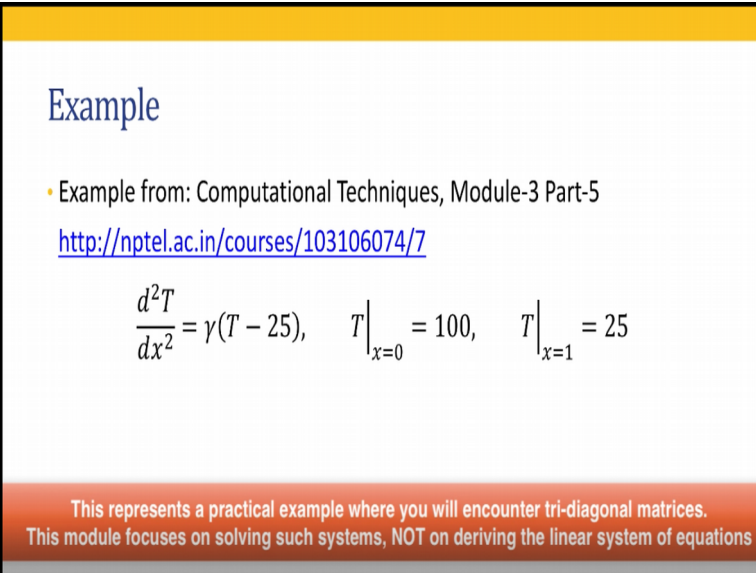
$$A = \begin{bmatrix} d_1 & u_1 & 0 & \cdots & 0 & 0 \\ l_2 & d_2 & u_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & d_{n-1} & u_{n-1} \\ 0 & 0 & \cdots & \cdots & l_n & d_n \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

So, let us look at how a Tri-Diagonal structure looks like. The matrix A shown over here is, what a Tri-Diagonal matrix will look like. So basically, there is diagonal elements d_1, d_2 up to the d_n okay, there is 1 sub diagonal row which is l_2, l_3 up to l_n , and 1 super diagonal that is 1 above the diagonal which is u_1, u_2 up to u_{n-1} okay. This is known as a Tri-Diagonal structure.

Lot of engineering problems, end up reducing to a Tri-Diagonal matrix structure, or sometimes what we have is, what is known as banded diagonal structure. Tri-Diagonal, is a specific example

of banded diagonal structure, with the band width=3. So, let us look at 1 particular example where, we get a Tri-Diagonal structure. This example is from the computational techniques course module 3 part 5 and the link for which is given over here okay.

(Refer Slide Time: 01:28)



Example

- Example from: Computational Techniques, Module-3 Part-5
<http://nptel.ac.in/courses/103106074/7>

$$\frac{d^2T}{dx^2} = \gamma(T - 25), \quad T|_{x=0} = 100, \quad T|_{x=1} = 25$$

This represents a practical example where you will encounter tri-diagonal matrices.
This module focuses on solving such systems, NOT on deriving the linear system of equations

If we have a rod, in through which, conduction is taking place, one end of the rod is held at 100 degree Celsius and the other end of the rod is at 25 degree Celsius, and this rod is losing heat to the surroundings, then the overall model for the system, can be derived, to obtain something like this okay.

(Refer Slide Time: 02:05)

Example

- Example from: Computational Techniques, Module-3 Part-5

<http://nptel.ac.in/courses/103106074/7>

$$\frac{d^2T}{dx^2} = \gamma(T - 25), \quad T|_{x=0} = 100, \quad T|_{x=1} = 25$$

- If we use central difference formula,

$$\left. \frac{d^2T}{dx^2} \right|_{x_i} = \frac{T_{i+1} - 2T_i + T_{i-1}}{h^2}$$

Now if we want to solve this set of equations, we can use the numerical differentiation that we covered in module 3 okay. And the central difference formula will be of this type. Now this is substituted to the left-hand side and right hand side becomes, gamma multiplied by $T_i - 25$. You write this for multiple locations that means at the initial point T_1 , the next point T_2 , T_3 and so on up to T_{n+1} okay. If you do that, you will end up reducing, this particular differential equation into, n linear equations in n unknowns.

(Refer Slide Time: 02:37)

Tri-diagonal System of Equations

Discretizing in 10 intervals, with $\gamma = 4$, results in following equations:

$$\begin{array}{rcl} T_1 & = & 100 \\ T_1 - (2 + \alpha)T_2 + T_3 & = & \beta \\ T_2 - (2 + \alpha)T_3 + T_4 & = & \beta \\ \vdots & & \vdots \\ T_{11} & = & 25 \end{array} \quad \begin{array}{l} \alpha = 0.04 \\ \beta = -1.0 \end{array}$$

And these, linear equations for example, the example that we covered in computational techniques lecture, consisted of discretizing in 10 intervals. If $\gamma=4$, then the first equation is going to be $T_1 = 100$, and for all the nodes within the rod, we get equations. If we do the derivation, will get the equations of the type, $T_1 - (2 + \alpha) T_2 + T_3 = \beta$ and so on and so forth okay.

So, what you can see over here. The first equation, the diagonal element is 1 and the non-diagonal elements are all 0. And for the vector b , the first element is 100. If you look at the next equation, the diagonal element is $(-2 + \alpha)$ okay. And the sub diagonal and the super diagonal elements are both equal to 1. So, let us go on to MATLAB, and start trying to solve this particular problem.

(Video Starts: 03:38) myTDMA, I have already created a skeleton of the structure. So, first step, that we are going to do is, we are going to create the problem matrices okay. (Video Ends: 03:53). If we go and look at power point again, okay what we are going to have is, basically the A matrix of the sort. We have the diagonal elements which we are going to store in vector d . The super diagonal elements where we are going to store in vector u , and the sub diagonal elements which we are going to store in vector l . That is what we are going to do okay.

(Video Starts: 04:18) So, our vector l , $l(1, 1)$ is going to be equal to 0 okay. Let us look at this again $l(1, 1) = 0$ because there is no element, in the first row for l . And in the last row, l_n is also equal to 0. So, we will do that as well okay. l_n , we should actually, let us just put this as $l_{n+1, n} = 0$ because 10 intervals and therefore we have 11 nodes so this is also equal to 0. We need to create n first. In this case, our n was equal to 10, our α was equal to 0.04 and our β was equal to -1. Let us just check those things, $\alpha=0.04$ $\beta=-1$ and $n=10$ okay.

Likewise, for u also we are going to get, $u(1, 1) = 0$. Remember, the first equation was just, $T_1 = 100$. So, $u(1, 1)$ was 0 and $u(n+1, 1)$ was also equal to 0. Again, recall the last equation, was also $T_{11} = 0$ sorry, $T_{11} = 25$ which meant that, the non-diagonal elements were all 0. And the entire set of diagonal elements, that we have seen is the first guy, and the last guy, are both equal to 1 whereas, everything else is $-2 + \alpha$ okay.

So we will write $d(1,1)=1$ and $d(n+1,1) = 1$ again and likewise $b(1,1)$ was equal to 1 and sorry, was equal to 100 and $b(n+1,1)$ was equal to 25. So, that is the structure that we have created. The

first and last row are taken care of. Now we want to take care of all the middle rows, so $l(2:n,1)$ equal to, we will decide this, $u(2:n,1)$ again, we need to decide this. I will just put question marks here, again this is an error but we will change that immediately. $(2:n,1)=??$ And $b(2:n,1)$ again=?? So, let us see what we need to put in our vector l okay. (Video Ends: 07:12)

So, our vector l , as we had said was, $1 * T1$ that gets l . And $1 * T3$, that is in u . So, u $l3$ is going to be sorry, $l2$ is going to be $=1$, $l3$ is going to be also equal to 1 . $u2=1$, $u3=1$ and so on. (Video Starts: 07:38) So, all the middle elements, of both l and u matrices are all equal to 1 . So, we will put this, the diagonal elements. If we check the diagonal elements are $(-2 + \alpha)$, so we will put that $(-2 + \alpha)$.

And our b elements are equal to β . So, we will put that β . And with this, we have done with creating the problem matrices. We have saved this. (Video Starts: 08:11) Let us go to MATLAB and run myTDMA. To ensure that, we do not have errors at this stage okay. We do it right. So, let us say, were okay.

So here, if you look at this particular code, what I have done is, I have made a typo, instead of putting a comma over here, I have put a dot, which is type of typo. So, let me go and change that, and save, and let me run it, and see whether we get errors. So, does not look like. So, `clc`, what I had also done was create a small function called `show matrix`.

So if I call the `showMatrix` and give (l, d, u) okay. I will get our A matrix over here okay. This is just so for our ease of discussion, and I will just print, first 5 elements of this okay. So, these are the 5 elements, and this looks exactly as we wanted, our first guy is going to be 1 , our second guy is $(1 - 2 + \alpha)$ and 1 , the third guy is $0 \ 1 \ (-2 + \alpha \ 1)$ so on and so forth okay. And if we were to print out the entire A matrix. We will get the desired result as well okay. And let us also print out the b vector. And so, this is the b vector, and this is the A matrix okay.

Now at this stage, what we can also do, solve it using the methods that we have seen in lecture 4.1 which means our x . We will call this, as $x1 = A \setminus b$ and this is the solution okay. So, this is the overall temperatures, in the rod going from one end to the other, from the 100-degree Celsius end to the 25 degrees Celsius end. This is the result. What we want to do is, reproduce this result not using the slash but using the TDMA algorithm. (Video Ends: 10:18)

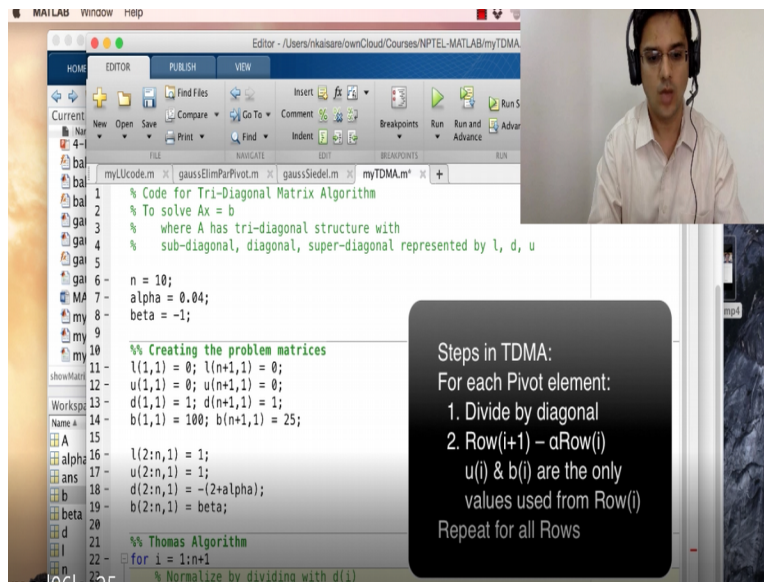
So, how does the TDMA algorithm work? You can go again to this link that is shown over here and you can see this video to try to understand how the TDMA or the Thomas algorithm works? We are going to just use that, it is kind of similar to the gauss elimination, except we account for the fact that, all the other elements except d , l and u are 0. So, we do not need to do all the computation, that gauss elimination does, that is the one.

The second difference is that, at in each step, we are going to divide by the pivot element, d_1 $A(1, 1)$, $A(2, 2)$ so on and so forth, before doing a typical gauss elimination step. So, let us go and start working that out. (Video Starts: 11:07) In the Thomas algorithm the first step, is going to be, normalized by dividing okay. Let us put this in a loop, for $i = (1 : n+1)$ the loop should not be $(n+1)$ we will come to that later, but for now, I have, let us just try to do it for the entire loop okay.

So, the first row is going to be divided by, d_i , or any row is going to be divided by d . So at whenever, you know, $A(i, i)$ is the pivot element, everything to the left of the pivot element is already 0. So, we do not have to worry about dividing anything in the l vector. We only have to look at the d vector, the u vector, and the b vector. But keep in mind that because we are dividing by d itself, we do not have to do anything with respect to d vector either.

So, what we are going to do is, $u(i)$ is going to be nothing but $u(i)$ divided by $d(i)$, and same thing with the b also $b(i) = b(i) / d(i)$ and $d(i)$ is going to be equal to 1. Why because $d(i)$ is going to be old $d(i)$ divided by itself so therefore it is equal to 1 and that is the only thing that we need to do. Let us also put end over here okay. So, we ended up normalizing it by dividing with the $d(i)$. So, that is the step that we are going to take. So, that is the first step with respect to normalizing.

(Refer Slide Time: 12:25)



So, what we did now was, with $A(1, 1)$ as the pivot element, row 1 was divided by that pivot element. So, that in the first row. Now we have 1, u divided by d 0, 0, 0 and so on. And the last guy is b divided d okay. Now we are going to use that pivot element, to make zeroes in the first column.

Now where do we make to need to make 0? We only need to make 0 in the row number 2, or row number $(i+1)$. Why because, that is the only non-0 element. So, $l(i+1)$ is the only non-0 element in that pivot column, using pivot element for elimination okay. So, our alpha is going to be nothing but $l(i+1)$. Why, because our d(i) that means, $A(i, i)$ is already equal to 1, because of this step okay.

So if we want to make 0, in the $i+1$ th row, i th column, then we need to just multiply row i with $l(i+1)$ and subtract it from row $(i+1)$ and that is essentially what we are going to do. Your alpha is just going to be $l(i+1) / d(i)$ but $d(i)$ is already 1. So, you do not need to make that division okay. So, $\alpha = l(i+1)$, $l(i+1)$ is going to be equal to $l(i+1) - \alpha$ okay which is basically going to be equal to 0. So, why we should do this extra calculation? We do not need to do this extra calculation. We can directly put that equal to 0 and that is essentially what we have done okay.

Now $d(i+1)$ is also going to change. And $d(i+1)$ is going to change, because $d(i+1)$ is going to be $d(i+1) - \alpha * u(i) - \alpha * u(i)$ okay. And the same thing with b also $b(i+1)$ is going to be

equal to $b(i+1)$ okay. What is above $b(i+1)$? Above $b(i+1)$ is exactly $d(i)$. So, it is going to be $-\alpha * b(i)$ okay. So, that is the thing.

So, what about the $u(i+1)$? The element above $u(i+1)$ is 0. The element above this guy is 0. So, if we do any kind of a computation, it is still going to result in $u(i+1)$. So, we do not need to change $u(i+1)$. So, we are done over here. So, that is all we need to do with respect to gauss elimination okay.

This gauss elimination is applied to a special structure, the banded structure, of the Tri-Diagonal matrix. So, let us now run this okay. I expect an error, and I will come to that, when we actually get an error. I expect an error, and I will explain that in a minute. Let me just run it, okay and let us see what error we get okay. Error is, the attempt to access $l(12)$ index out of bounds. So, what is happening over here is, this is when $i = (i: n+1)$ with which means when $i=11$ okay we have done this normalization, that is fine, however this cannot continue. Why, because there is nothing called, $l(12)$ or $d(12)$ or $b(12)$, our matrix ends, at row number 11, column number 11 okay.

So, what we needed to do really was, we need to do only 10 row operations and not 11th row operation because $r(11)$ is not going to be used to change anything below because $r(11)$ is the last row. So we need to use this, to go from $i = (1: n)$ only okay. So, let us do that. Let us go to MATLAB, you clear, clear all, clc and run myTDMA okay. And we have our results over here okay.

Now, what we want to do is, we want to check our A matrix. So A is showMatrix, and then we want to display $A(1:5)$ and see how they look like okay. So, this is how our A matrix is going to look. So, A matrix has now, an upper triangular structure and a special kind of upper triangular structure. Only the u elements in the upper triangular structure are non-zero, everything else is zero

So, we have your $A(i, i)$ as non-zero. $A(i, i+1)$ is non zero and $A(i, i+1)$ is nothing but equal to $u(i)$ okay. So, when we do back substitution okay. $A(i, i)$ is already equal to 1. When we take this guy, to the right hand side, we are going to get $b(i) - u(i) * x(i+1)$ okay. And that is our $x(i)$. We do not need to divide by $A(1, 1)$, because $A(1, 1)$ is already equal to 1. So, back substitution is also significantly simplified okay.

So, our first guy is, our first step is going to be, we are going to say $x = \text{zeros}(n+1, 1)$ and $x(n+1, 1)$ and x is going to be equal to $b(n+1) / d(n+1)$ okay. Why do we need to do divide $d(n+1)$, is because all these things, we have done it, only until row number 10, row number 11? We have not yet divided by the pivot element. So, we need to do that, in the back-substitution step okay. And then we have the for loop for back substitution for $i=n$ in steps of $(-1: 1)$ okay.

And our $x(i)$ we said was going to be equal to $b(i) - u(i)$ multiplied by $x(i+1)$. Keep in mind, $u(i)$ is nothing but $A(i, i+1)$. $A(i, i+1)$ is multiplying with $x(i+1)$. We take that to the right-hand side, subtract it from $b(i)$ and divide by 1. We do not need do that division because, dividing by 1 is going to give us the same result and end okay. And that is the back substitution that we needed. So, let us see whether we are going to get any errors or this is going to run like a chance.

So, let me click the run, and let us go back to our command prompt, and see what solution x we are getting okay. So, this is what we are getting as x . Let us check back, with $x1$ and compare the 2 results okay. So, as we can see, the results using the gauss elimination that MATLAB used when it did backslash. And the results using our TDMA algorithm, they are matching exactly okay.

So TDMA algorithm, is nothing different, it does nothing different from gauss elimination with back substitution, except for the fact that exploits the structure of the Tri-Diagonal matrix. That means, there are only 3 elements in any row which are non-zero and therefore, we need to take significantly lesser number of computations, compared to a standard gauss elimination algorithm okay. (Video Ends: 20:48)

(Refer Slide Time: 20:50)

Tri-Diagonal Matrix



$$A = \begin{bmatrix} d_1 & u_1 & 0 & \cdots & 0 & 0 \\ l_2 & d_2 & u_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & d_{n-1} & u_{n-1} \\ 0 & 0 & \cdots & \cdots & l_n & d_n \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

We will solve the above problem using Thomas Algorithm

(For Theory, see: <http://nptel.ac.in/courses/103106074/7>)

So, that is what I wanted to cover with respect to Thomas algorithm. What I would put as an exercise for you guys is, to do the same problem but with a larger number of examples. And that is going to be first problem and the second problem, what it is going to be is, to solve the exact same problem that we have done in this lecture. To solve this exact same problem but using the gauss elimination method.

So here you will need to construct your A matrix and your b vector and compute this using gauss elimination. So, what you will need to do is basically you create this A matrix and the b matrix. I have not shown you my code, which is the show matrix code, which is, which I used in order to calculate a matrix from l, d and u that is the main crux of your second problem of this assignment

So, with that I come to the end of lecture 4. 5, and indeed to the end of module 4. In module 4, we have covered solving linear equations of the type $Ax=b$, in lecture 4.2 we covered naïve gauss elimination, lecture 4.3 we change the gauss elimination to gauss elimination with partial pivoting, as well as we saw very quickly how the LU-decomposition works.

In lecture 4.4, we moved on to what is what are known as iterative methods and covered gauss Seidel method, and in today's lecture that is the final lecture we covered a special method known as Thomas algorithm for solving Tri-Diagonal system of equations. Thank you and I will see you next week where we are going to start working on solving nonlinear equations. Thank you and see you in the next week.

