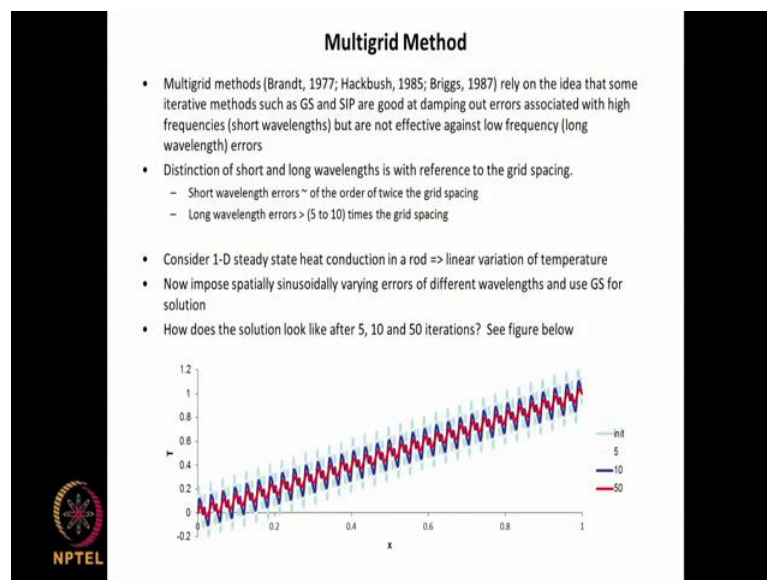


Computational Fluid Dynamics
Prof. Sreenivas Jayanti
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture - 53
Multigrid Method

We have come to the last lecture of this module on Advance method for the Solution of Ax equal to b . We have seen in this second part of this module, methods like successive over relaxation alternating direction implicit method and the strongly implicit procedures, all of which are attempts are trying to accelerate the rate of convergence of an iterative scheme multi grid methods are probably the best at making this oscillation.

(Refer Slide Time: 00:45)



These have been in development since mid 70s and 80s. And now this have reached a stage of fusion that these are now routinely being used for the for CFD type of problems. So, we would like to see what is the principle behind this and how it actually works, but in an academic context in the sense we will take the simplest possible example and we will try to understand how this whole thing works.

The idea of this multi grid methods is that in an iterative method we are trying to reduce the residual and residual represents the degree of non satisfaction of the current solution with respect of the original governing equation Ax equal to b and this satisfaction is non-

satisfaction is there because there is some error in the solution. So, we are trying to reduce the error.

Now, this error usually can be decomposed into many components and some methods like the Gauss-Seidel method are good at reducing error which is at the smallest scale of error and there are some other methods which are not so very effective in that. If you are reducing if you are using Gauss-Seidel method then that part of the error which is there in the smallest wavelength gets reduced very fast very efficiently, but the larger wavelength errors are not damped so easily you will have to work on it many many times before you can damp it.

And illustration of that is what we are we are going to begin with. So, we are saying that multigrid methods rely on the idea that some iterative methods such as Gauss-Seidel method and also the strongly implicit procedure are good at damping out errors associated with high frequencies or short wavelengths, but are not effective against low frequency a long wavelength errors. The distinction of short and long wavelengths is with reference to the grid spacing. So, if you have an error which varies as the wavelength of twice the grid spacing that is the shortest possible error, so that is considered as a short wavelength.

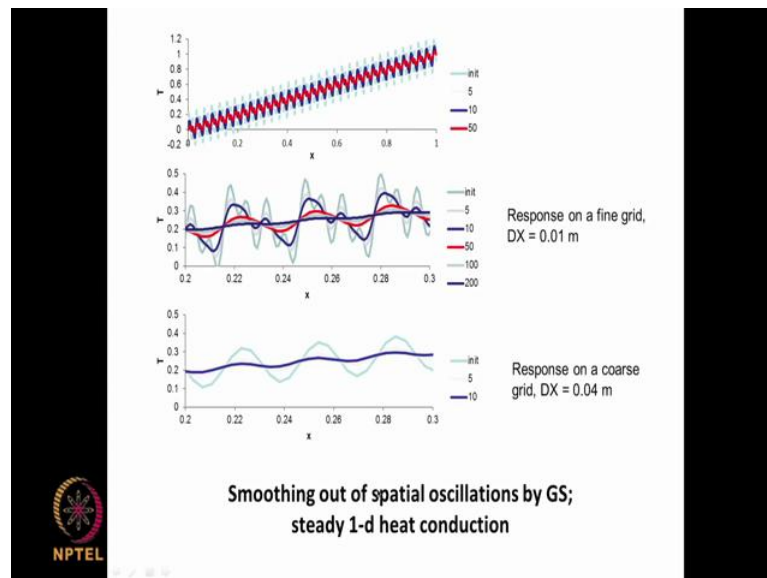
And long wavelength errors are anything of the order of five to ten times the grid spacing. So, we illustrate this error damping properties of the Gauss-Seidel method by considering the example of 1-D steady state heat conduction, in a rod so that you are expecting a linear variation and this is the kind of linear variation that we are looking at we have x going from 0 to 1 and T going from 0 to 1 as you go from x equal to 0 to x equal to 1, it is straight simple problem; 1-D steady state heat conduction with constant thermal conductivity, so $\alpha d^2 T / dx^2 = 0$.

So, that is the kind of thing without any source term. So, that will lead to a truly linear variation like this. Now what we are doing here is that we are imposing on this true linear variation we are introducing some error which is spatially distributed in the form of wavelengths, in the form of science order wave components and what you see very faintly here and we will see later is the error which is super post on this linear variation that is why you have this waviness that comes here.

And this waviness is composed of 3 4 different wavelengths, one which is short wavelength one which is bigger and one which is even bigger. So, a Gauss-Seidel method or any other method would try to damp out this errors, so as to give rise to the linear variation. So, right now this is the kind of variation, this is the kind of expected T that we have and we would like to go from the faint blue one to the blue one to the red one and that is happening over 50 iteration in this particular case.

So, we are considering 1-D steady state heat conduction with a linear variation and on top of this linear variation we are adding certain error of different wavelengths. So, we are saying that T at any point is that given by the linear variations between 0 and 1 for example, at midpoint x equal to 0.5 we expect T to be 0.5, but now we are saying that T is 0.5 plus some a sign x by lambda, lambda 1 plus b times sign x by lambda 2 like that. So, that is variation that is being imposed on this and we would like to see that if you take this wave v t as your initial guess then, after how many iterations the waviness disappears and we get a perfectly linear variation that is what we trying to see here. It is difficult to see in this.

(Refer Slide Time: 06:29)



So, we have a magnified version here. The one that we can see here is the initial solution you can see its going up like this and then going up, going down and then going up big wave coming down here and then like this, so this has a variation of this wavelength plus a variation of this wavelength. So, there are two variations one of what is eventually this

variation, the red variation and one which is of one quarter of that variation wavelength here, so the combination of these two is giving rise to these small wavelength variations plus large wavelength variation. So, this is what we have as the initial guess.

Now if you then put the Gauss-Seidel method for the solution of these equations this is with a Δx of 0.001. So, between this and this we have 100 grid points and 100 individual values, between this and this we have; it should be Δx of 0.01. So, it is not 10 its 0.001 I think. So, it should be large number of grid points here, so as to give you this. So, if you start with this then after 5 iterations it becomes like this very fine to gray line and after 10 iterations you see the blue line which is this one here, and after 50 iterations you see the red line. Now, what is there in red line and what is not there red line is of significance.

The red line here after 50 iterations has still retained the long wavelength variation, but it does not have the short wavelength variation that is exhibited in the initial guess. So, short wavelength variations it is going up and coming down over the small interval, again it is going up and coming down over this one. So, over a short variation small x interval it is already showing a sinusoidal variation. But this one is not showing any sinusoidal variation of that wavelength because it is all a single sinusoidal variation corresponding to wavelength from here to here.

This is a wave length that is remaining after 50 iterations whereas, in the initial variation you have this long wave variation plus a short wave variation corresponding to this. So, the short wave variation got damped to within about 10 iterations, but the long wave is persisting after 50 and after 100 and even after 200. The amplitude is decreasing, but even that variation is still there. So, it is taken 200 iterations or more to bring it almost up to the linear variation.

But out of this the variation corresponding to the short wavelength is damped out within 10 iterations. Now, therefore, if you have an error which is composed of very short wavelength variation and a longer wavelength variation you will have to do many computations much iteration before you can reduce the long wavelength thing. That is what we will hold up your solution because your solution requests you to have very small residual. And variation like the red one here will have fairly large residual, and that residual will have to go down and down, and that will happen only as this approaches

this linear variation and that has taken 50 iterations to come up to this and then 100 iterations to come up to this and 200 iterations to come up to this.

Now, if we did not have the Δx of 0.01 or 0.002 I think, instead of that if you had 4 times the grid spacing then this same red variation is shown here, but you can see that this is like a piecewise linear, so you have one grid point here, one grid point here, one grid point here, like this. So, the Δx here is bigger 4 times as big as this. And if you start with this variation as the initial guess after 10 iterations you have come up to this. So, what has taken you 200 iterations to 100 iterations to come up to this variation you have achieved that in 10 iterations on a larger grid, because on this large grid this variation is like a short wavelength variation.

So, the same Gauss-Seidel method with the Δx spacing of 0.01 or 0.002 for here, is taking 100 iterations to reduce this wavelength error in the red thing. But it takes only 10 iterations on a grid spacing which is 4 times bigger that means, that a scheme like Gauss-Seidel can reduce the error associated with short wavelengths very efficiently and at the same time it is very inefficient in dealing with large wavelength errors. And what are these wavelength errors? We have seen in when we are looking at stability analysis of Fourier decomposition, Fourier decomposition often error into a number of wave components. So, if you have an error in the general case it can be decomposed into a large number of Fourier components each of different wavelengths in the x direction, in the space direction.

In the linear problem each of those errors needs to be damped out, each of those red variation wavelength variation should be damped out, so as to give rise to this linear variation. So, how what is the damping rate? The damping rate for a given scheme depends on what type of scheme it is and on also on the grid spacing what is a wavelength of the error wavelength and what is the Δx that that you have. So, if your Δx is very small compared to the wavelength then it will take large number of iterations. If your Δx is small compared to this then it will take only a few iterations to get down to this.

So, what do we do? We cannot choose the wavelength error because that is imposed by the problem and initial guess and how the error changes, and in the general case you have wavelengths of all wavelengths short wavelengths, long wavelengths, even longer

variations, long-long wavelength variations all of them are there. So, if you have only one grid then you can only damp out wavelengths of the shortest wavelength very fast. So, in multigrid the idea is that you do not have a single grid to deal with to get the solution, you use a short wavelength, fine grid to remove quickly the short wavelength variation and then you go on to a coarse grid, and then you remove again the short wavelength errors on the coarse grid fast.

These things were difficult to remove on the fine grid because for a fine grid these are long wavelength variation and that is illustrated here. The same the red variation here is a long wavelength variation for a Δx spacing a 0.01 and its short wavelength variation for a Δx spacing a 0.04. That is why it has taken 50 iterations to come from this amplitude to this amplitude; it has taken another 150 iterations to come from this amplitude to this amplitude.

But on this, on a coarse grid where it is like a short wavelength it has taken from here to here it has taken only 10 iterations, so that is the advantage that is efficacy of the Gauss-Seidel method in reducing short wavelength errors and that is also the inefficiency, weakness of the Gauss-Seidel method in dealing with long wavelength errors. So, one possibility is to have horses for coarse policy, for long wavelength use coarse grid, for short wavelength errors use fine grid.

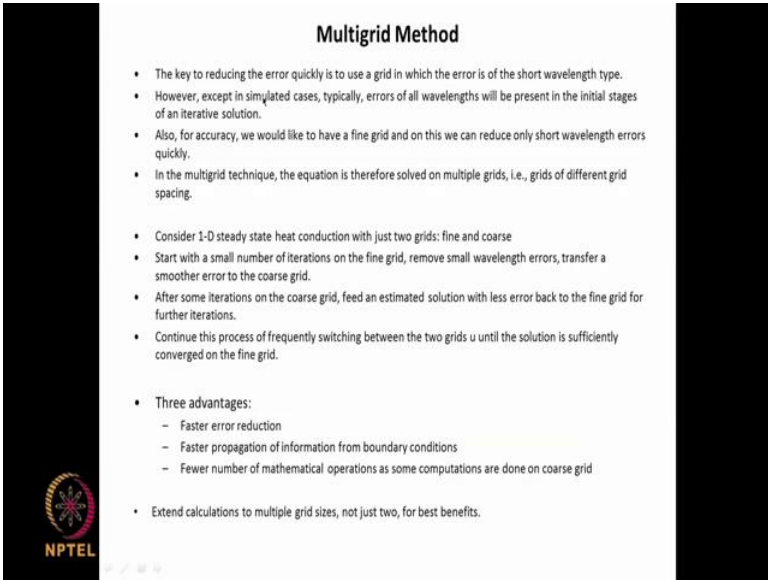
But it is not easy because you can have one grid that is a conventional thing. In multi grid method you flip, you go from fine grid to coarse grid, but if you ultimately get a solution coarse grid then we have too much of truncation error because truncation error is usually proportional to Δx we need to solve it on the finest grids that is possible and there your method is effective only in reducing the short wavelengths quickly and long wavelengths will take a huge number of iterations.

So, in multi grid method we start with computation on the fine grid we do a few number of iterations maybe 5 10 iterations, and then we take the error from the fine grid to coarse grid. There we work some more iterations and because it is no coarser grid what we are earlier long wavelength some of them would be damped out quickly and then we go to the next even coarser grid, so what the middle coarse grid is unable to damp out quickly this will be damped out in the even coarse grid. And then you go to the next coarse level. So, the error will also be damped out. So, in each grid you are doing only ten iterations

and then you go to the coarse grids and then you now have reduced error use this reduced error to improve the solution on the fine grid and then you do some more calculations.

So, finally, you are going from the fine grid to the coarse grid and then you are coming back to fine grid and each time you are trying to reduce error and eventually even the error on the fine grid becomes very small at that point you can say that what go to a converge solution. But always you are reducing most of the error in the coarse grids and not in the fine grid. So, fine grid is for final tune up. So, this is the idea of the multi grid and we will see how it actually gets implemented.

(Refer Slide Time: 18:00)



Multigrid Method

- The key to reducing the error quickly is to use a grid in which the error is of the short wavelength type.
- However, except in simulated cases, typically, errors of all wavelengths will be present in the initial stages of an iterative solution.
- Also, for accuracy, we would like to have a fine grid and on this we can reduce only short wavelength errors quickly.
- In the multigrid technique, the equation is therefore solved on multiple grids, i.e., grids of different grid spacing.
- Consider 1-D steady state heat conduction with just two grids: fine and coarse
- Start with a small number of iterations on the fine grid, remove small wavelength errors, transfer a smoother error to the coarse grid.
- After some iterations on the coarse grid, feed an estimated solution with less error back to the fine grid for further iterations.
- Continue this process of frequently switching between the two grids until the solution is sufficiently converged on the fine grid.
- Three advantages:
 - Faster error reduction
 - Faster propagation of information from boundary conditions
 - Fewer number of mathematical operations as some computations are done on coarse grid
- Extend calculations to multiple grid sizes, not just two, for best benefits.

Let us just try to paraphrase this. The key to reducing error quickly is to use a grid in which error is of the short wavelength type; however, except in simulated cases like what we have just now seen, typically errors of all wavelengths will be present in the initial stages of an iterative solution and also for accuracy we would like to have a fine grid and on this we can reduce only the short wavelength errors.

In the multigrid technique the equation is solved on multiple grids, grids of different grid spacing consider 1-D steady state heat conduction equation with just two grids - one fine grid and coarse grid. You start with the small number of iterations on the fine grid, removes small wavelength errors; transfer a smoother error onto the coarse grid. After some iteration on the coarse grid, you have reduced the large wavelength error somewhat

and then you have now a different error distribution than what you had at the end of fine grid solution.

So, from this error you estimate what would be the true solution, and then you give this estimated solution back to the fine grid. So, you feed an estimated solution with less error back to the fine grid for further iterations and in the estimation process you have also introduce some small wavelength errors, you damp them out and then you reduce error and then you bring it back to the coarse grid. So, you continue this process frequently switching between the two grids until the solution is sufficiently converged on the fine grid.

So, when you do this kind of computation we still at this stage do not fully understand this, we will do that shortly. But this is the idea that you have different grids, for the same domain different grids of different Δx and you do part of the computations on the fine grid and then you go to the coarse grid and then you go to the next coarse grid and then you come back to the fine grid and then you keep on doing this; and reducing part of the error in the fine grid part of the error in the coarse grid and part of the error in even coarse grid.

So, by doing this you gain 3 advantages you get faster error reduction because each error is being reduced efficiently on grid of different sizes and we know that the grid size does matter when we look at reducing error using something like Gauss-Seidel method. Faster propagation of information from boundary condition, when we look at typical elliptic equation the interior points need to know what the boundary condition is. So, that information propagates one step one Δx at a time. So, if you have small Δx then it takes a large number of steps before the interior point gets to know what is the boundary point.

If you have a coarse grid that boundary point information gets propagated into the interior faster, in fewer number of iterations. And finally, we know that the number of arithmetic operations which are required for the solution of $Ax = b$, where A is $n \times n$ square where n is the number of grid points if you have coarse grid the number of grid points is less, that means, that the solution will take fewer number of mathematical operations. So, the number of mathematical operations that are required on the coarse grid in order to get a solution for the error is far less than what is required for a solution

on the fine grid. So, since you are doing part of the computations on the coarse grid your overall number of mathematical operations, arithmetic operations is reduced.

So, because of all the 3 advantages you can get much faster solution on the multi grid, this multi grid can be extended to multiple grid sizes not just two and the more the number of grids that you have the faster; the better will be the benefits.

(Refer Slide Time: 22:29)

Multigrid Method

- Simple case of 1-d, steady state heat conduction

$$d^2T/dx^2 = f(x) \quad , \quad T_{i=0} = T_0 \text{ and } T_{i=L} = T_L$$

- For point i on the fine grid, this can be written as $(T_{i+1} - 2T_i + T_{i-1})/\Delta x^2 = f_i$
- After k iterations on the fine grid, the solution satisfies the governing equation with a residual ρ^k :

$$(T_{i+1}^k - 2T_i^k + T_{i-1}^k)/\Delta x^2 = f_i - \rho_i^k$$
- The error, e_i^k , defined as $e_i^k = (T_i - T_i^k)$, evolves as per the following equation

$$(e_{i+1}^k - 2e_i^k + e_{i-1}^k)/\Delta x^2 = \rho_i^k$$
- Since error is already smoothed, it can be solved on a coarse grid. The error equation therefore needs to be extrapolated or "prolongated" to the coarse grid.

So, let us just see how it works for a 1-D problem. We are considering now 1-D steady state heat conduction equation with $d^2T/dx^2 = f(x)$. So, with the boundary condition that T at x equal to 0 is T_0 and T at x equal to l is T_l , these two are specified T_0 and T_l is specified, so that you have an elliptic problem. And we have a domain 1-D domain, so it is going this is the x direction, increase in x direction and you have two grids in this way case small i and small i minus 1, small i plus 1 these are all the grid points on the fine grid. And capital I , capital I minus 1, capital I minus 2, I plus 1 are the grids in the coarse grid. So, you can have a Δx which is coarse grid spacing which is twice the grid spacing on the fine grid. So, we are looking at two grids - fine grid and a coarse grid here.

So, for point i which is there in the fine grid and also in the coarse grid this equation can be written as $T_{i-1} - 2T_i + T_{i+1} = \Delta x^2 f_i$. This is our conventional finite difference formula at point i with a grid corresponding to the small i 's. So, you have $i - 1$ this value, $i + 1$ this value and this

value. So, that gives you a central difference approximation for this and if you do this all the grid points you get at equal to b and then you can solve it.

But you are solving the aT equal to b in an iterative way and we have seen the Gauss-Seidel method. So, after k iterations on the fine grid the T_k that is the solution for T_i^k satisfies the governing equation with the residual. So, that residual is such that it is equal to $T_i^{k-1} - 2T_i^k + T_{i+1}^k$ divide by Δx^2 is equal to f_i it is not exactly equal to f_i , but $f_i - \rho_i^k$. So, this is the residual and this is the amount which is if this is equal to 0 then you have an exact solution, but if it is not equal to 0 then you have these. So, if you say that T_i^{hat} is the exact solution then you can say that error ϵ_i^k is the exact solution minus T_i^k and then the error evolves as $\epsilon_i^{k-1} - 2\epsilon_i^k + \epsilon_{i+1}^k$ divide by Δx^2 and that is equal to ρ_i^k .

Now, we can solve this, this has already been solved for k iterations on this, and in the process all the short wavelength contribution of this error has been already damped out because the Gauss-Seidel method is very effective in removing short wavelength errors and if the error distribution, the spatial distribution has a combination of short wavelength and long length of different wavelengths then the short wavelength contribution is already removed. So, this has only the long wavelength things.

So, instead of solving it on the fine grid which is a small $i - 1$ small i plus small $i + 1$ we would like we can solve it on the coarse grid. So, since error is already smoothed it can be solved on a coarse grid. The error equation therefore, needs to be extrapolated or prolonged to the coarse grid, as it is this error equation is expressed in terms of small i with the grid spacing of small Δx . So, this is not post for the coarse grid. If we can extrapolate it in such a way that we have $\epsilon_{I-1}^{\text{capital}}$, $\epsilon_I^{\text{capital}}$, $\epsilon_{I+1}^{\text{capital}}$ that will be a like solving the error equation on the coarse grid.

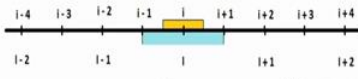
So, how do we use this error, how do we extrapolate this onto the coarse grid and that process of extrapolation is known as propagation in the multi grid literature.

(Refer Slide Time: 27:22)

Multigrid Method


- Derive coarse-grid error equation at I by adding the error equations for nodes $(i-1)$ and $(i+1)$:

$$\frac{1}{2}(\epsilon_{i-2}^k - 2\epsilon_{i-1}^k + \epsilon_i^k) + (\epsilon_{i-1}^k - 2\epsilon_i^k + \epsilon_{i+1}^k) + \frac{1}{2}(\epsilon_i^k - 2\epsilon_{i+1}^k + \epsilon_{i+2}^k) \Delta x^2 = \frac{1}{2}\rho_{i-1}^k + \rho_i^k + \frac{1}{2}\rho_{i+1}^k \equiv \rho^k$$
 or $(\epsilon_{i-1}^k - 2\epsilon_i^k + \epsilon_{i+1}^k)\Delta x^2 = \rho^k$



- Solve error equation on coarse grid for K iterations to get ϵ_i^{k+K} at all grid nodes
- Update T_i^{k+K} as $T_i^{k+K} = T_i^k + \epsilon_i^{k+K}$ and interpolate to get T_i^{k+K} on fine grid:

$$T_{i-1}^{k+K} = \frac{1}{2}(T_i^{k+K} + T_{i-1}^{k+K}); T_i^{k+K} = T_i^{k+K}; T_{i+1}^{k+K} = \frac{1}{2}(T_i^{k+K} + T_{i+1}^{k+K})$$
- Use these new values of T_i^{k+K} on fine grid and do further k iterations on fine grid, and repeat.
- Note that we solve T_i on fine grid and ϵ_i on coarse grid
- Can be extended to 3-d, unstructured, non-linear problems
- Studies show that arithmetic operations $\sim n^{1.1}$ where n = no of grid points



So, we derive the coarse grid error at error equation at for capital i here which happens to be the same i here by saying that this total Δx is composed of this small Δx at i plus half of the error equation at i minus 1 plus half of the error equation at i plus 1. So, this total error here is composed of the error here plus half of the error here and half of error here.

So, we can say that this whole thing is equal to half of the error equation written at i minus 1. So, that is error at i minus 2 k i minus 2 i minus 1 plus epsilon i here. So, that represents error equation at i minus 1 and error equation at small i is epsilon i minus 1 2 epsilon i plus epsilon i plus 1. And the error equation at this point small i plus 1 is epsilon i epsilon i plus 1 and epsilon i plus 2. So, all of this is equal to rho i minus 1 because this is rho i minus 1 k and this is rho i k plus this is rho i plus 1 k . So, this is the i think.

Now, when you look at this at the end of k iterations we do not have this epsilons, but what we definitely have our rho i minus 1 and rho i and rho i plus 1. So, these are known. So, the right hand side is known and we can say that the sum of all these things is now the residual at capital I at the end of small k number of iterations. So, this is the residual at on the coarse grid and you can now write all these thing as on the coarse grid as epsilon capital I minus 1 k and 2 epsilon i capital I k and epsilon i plus 1 capital I k plus 1.

Now, if you look at this you have error expressed at coarse grid points T_{I-1} T_{I+1} with a corresponding residual and for this can be return for the all the points. So, this will give us coefficient matrix A times error on the coarse grid equal to the residual. So, this equation can be solved because right hand side is known and right hand side is derived from the residual which is computed by solving this equation for each of this for k number of iterations.

So, with the known right hand thing we solve for error at each of this points on the coarse grid. So, now, we solve for k number of iterations this coarse grid error equation with residual extrapolated or prolonged from the fine grid. So, we solve this k iterations and then we get error and then we say the true value is the value at the under small k number iterations plus the estimated error at this particular point. If you go back to our definition here we are saying that error is the true solution minus this one here. So, the true solution is this plus the error. So, we had this solution and we have now a revised estimate of error. So, we have the estimated new solution after small k number iterations on the fine grid and k number of iterations on the coarse grid is the value at the coarse grid that we had here plus the error here.

So, with this we can say now we can go to the fine grid we know have an improved solution here and we can go to the fine grid. But the problem is that these estimated values are now available immediate at the coarse grid points that is T_{I-1} T_{I+1} like this; whereas, if you want to solve this on the fine grid we need to know at this point and also on T_{i-1} T_{i+1} T_{i+2} . So for this simple grid we can say that T_{i+2} is equal to T_{I-1} which is obtained here and this value is also known.

And for the intermediate points we evaluate we take interpolate from here. So, that is T_{i-1} is equal to half of T_{I+1} plus T_{I-1} by 2. Similarly T_{i+1} at this point is the average of these 2 values. So, and that way you can get the interpolated value and the computed value here the estimated value here.

So, by doing this interpolation you now have estimated better solution of T , why we say its better solution? Because you have run some more number of iterations k number of k iterations and reduce the error by solving this residuals equation here and therefore, it has less error and therefore, it is better approximation. Now with this small T_{i+k} T_{i-k}

as the initial guess you solve this equation here for another small k number of iterations and that will reduce the error further and at the end of that you again calculate residuals and then prolongate these things to the coarse grid and then you come back to the grid and then you reduce the error further.

So, that is how you go forward. So, you solve the original equation in terms of the true variable in this case temperature on the fine grid. And after doing certain number of iterations you estimate the residuals and then you solve the error evaluation equation on the coarse grid.

So, once you get the error estimate on the coarse grid then you estimate the solution on the fine grid and then with this solution you do further more error reduction and then after that you come back here. So, a switch between the fine grid and the coarse grid and after that it come fine grid and coarse grid. You solve the original equation involving the true variable T on the fine grid and then you estimate the error and then you solve the error equation with a prolonged residual on the coarse grid and from the coarse grid you have the estimated error. Based on this you estimate the error at each of these points here and then you can get an improved estimate of the solution after having done some more iteration on the coarse grid, and then you come back to the fine grid. Then you can repeat this process and if you do that over a number of such things you get an error reduction which is faster than if you at to use only a single grid.

So, that is the idea of the multigrid method it can readily be extended to three dimensions and it can be used for not just the diffusion kind of problems. It can be used for other type of problems also, but its best effectiveness in terms of the diffusion type problems. It can also be extended to unstructured problems, it can be extended to non-linear problems, but each time you go for these more complications lot more issues need to be solved; lot more issues needs to be addressed. And if you go through a literature you will see some suggested solutions for each of this and the state of the art is that you can do these multi grid solutions for real world fluid flow problems.

And when you do that, studies show that the number of arithmetic operations for the multi grid method varies as n raise power 1.1 where n is the number of grid points. So, this needs to be compared with n cube number of operations for a Gaussian elimination n square number of operations for a Gauss-Seidel method, n to the power 1.5 for the

optimal XOR method, and there are methods like conjugate gradient method which have as a theoretical limit n to the power 1.25 for the conjugate gradient thing and here you have n to the power 1.1.

So, this represents a significant improvement over the Gauss-Seidel method and Gauss elimination method and essentially we are making use of the same Gauss-Seidel method, but we are doing it on different grids. So, there is obviously more overhead, more amount of computation, more difficulty in writing programs and all that is there. But once you overcome those you can get much faster rate of error reduction and much faster turn out of the solution. Because it takes only n to the power 1.1, so that means, that it also easily scale up to larger problems. So, this is often used to advantage when you are dealing with large scale problems involving huge number of grid points. So, there are problems when you do non-linear problems and those types of things those are to be expected anyway, but we can expect some significant improvement over the simple Gauss-Seidel method.

So, with this we can close this module and we can close this module which is specially on how to solve $Ax = b$. And we have seen that there is a vast plethora of models and methods and these methods can be like direct methods or iterative methods, this can be very basic methods that have been known at the turn of the century or in early 1900s, but over the 20th century they have been study improvement in this rate of convergence by bringing in vast number of methods that are not to mathematicians that have been developed mathematicians. And these are now common place in terms of the recurrence in CFD literature. So, these are the kind of methods that make CFD what it is today where you can solve a million grid points on your desktop overnight and get a solution. So, something like this would not be possible if this method were not there.

We are finished 5 modules and we can say that we now have a bird's eye view of the entire process of the CFD, right from formulating the problem terms of identifying the equations, boundary conditions, and in terms of discretization, and in terms of putting together a method of solution for coupled equations and converting them into this $Ax = b$ type of things and efficient methods.

In the next module we are going to look at essentially the remaining part which is how to do the discretization, how to do the grid generation and especially when we have an

irregular geometry all these problems become even more. So, when you are dealing with irregular geometry which is often the case imagine you have a car, car cannot be described in cylindrical coordinates and any car component cannot be described in terms of Cartesian coordinate system, we will have these step kind of things. So, when you are dealing with those kind of bodies of irregular shape and you have to look at fluid flow over those kind of things or through those kind of things, you need to have more sophisticated methodology and we are going to look at how to deal with these kind of things.

In the first part of the last module we look at the finite volume method and we look at how we can generate grids for this odd shaped flow (Refer Time: 40:51) computational domains. And then in the second part, we look at another aspect of reality we have so far been assuming laminar flow, but turbulent flow is quite common and for turbulent flow this simplistic approach cannot be used and we have to do something about it and that will be done in the second part of the second of the last module.