

Introduction to Finite Volume Methods –I
Prof. Ashok De
Department of Aerospace Engineering
Indian Institute of Technology, Kanpur

Lecture – 39
Error Analysis-II

(Refer Slide Time: 00:13)

Error analysis (Ax=b)

► When solving $Ax = b$ by Gaussian Elimination, we will see that a bound on $\|e_x\|$ such that this holds exactly

$$A(x_{\text{computed}} + e_x) = b \quad \leftarrow \text{bounded}$$

is much harder to find than bounds on $\|E_A\|$, $\|e_b\|$ such that this holds exactly

$$(A + E_A)x_{\text{computed}} = (b + e_b).$$

Note: In many instances backward errors are more meaningful than forward errors: if initial data is accurate only to 4 digits for example, then my algorithm for computing x need not have 10 digits of accuracy. A backward error of order 10^{-4} is acceptable.

INDIAN INSTITUTE OF TECHNOLOGY KANPUR Ashoke De 45

Ok so, welcome to the lecture of this Finite Volume Method and what we are discussing is that the properties of linear system like the metric system. So, we have discussed so far the basic properties of the matrix then, we looked at the norm and then the error. So, while doing the error, we have stopped in the last lecture by calculating the forward and backward error.

(Refer Slide Time: 00:47)

Error analysis

Example:

$$A = \begin{pmatrix} a & b \\ 0 & c \end{pmatrix} \quad B = \begin{pmatrix} d & e \\ 0 & f \end{pmatrix}$$

Consider the product: $fl(A.B) =$

$$\begin{pmatrix} (ad)(1+\epsilon_1) & [ae(1+\epsilon_2) + bf(1+\epsilon_3)](1+\epsilon_4) \\ 0 & cf(1+\epsilon_5) \end{pmatrix}$$

with $\epsilon_i \leq u$ for $i = 1, \dots, 5$. Result can be written as:

$$\begin{pmatrix} a & b(1+\epsilon_3)(1+\epsilon_4) \\ 0 & c(1+\epsilon_5) \end{pmatrix} \begin{pmatrix} d(1+\epsilon_1) & e(1+\epsilon_2)(1+\epsilon_4) \\ 0 & f \end{pmatrix}$$

► So $fl(A.B) = (A + E_A)(B + E_B)$

► Backward errors (E_A, E_B) satisfy:

$$|E_A| \leq 2u|A| + O(u^2) \quad ; \quad |E_B| \leq 2u|B| + O(u^2)$$

So, now, if you move ahead and see that we lastly took an example of 2 by 2 system and that was a matrix this is where we actually stopped, but before moving ahead just I would like to recall what has been discussed. So, two systems ABC 0 def 0 then you consider the floating point calculation of a dot B and you get the another 2 by 2 system like this.

And since this all the epsilons which are consider 1 2 3 4 5 all this epsilons are a small number. So, you recast this particular system in or rather split into two different matrices, A B into 1 plus epsilon into 1 plus epsilon 4. So, essentially what you are taking is that you take from the first component you split a and d into this, this is how it goes second component you take B 1 plus epsilon into epsilon 4 this one comes here other component like E 1 plus epsilon 2 epsilon 4 comes here then, 0 remains here then the first matrix written C 1 plus 5 and the last one is f. So, in a compact notation if you write it is a floating point calculation of A dot B is a plus epsilon A or E A E B these are essentially the representation of the errors.

So, the backward errors rather which satisfy the condition the magnitude of E A less than equals to the smaller number and this is the order magnitude of E B is this number less than equals to with this is the order. So, this is where we stopped with the forward and backward error calculation. Now we will try to see when so, slowly one having said all this information about norms errors and all these things we are slowly moving towards the system, which allows us to get an solution for a linear system through different

approach so for example, when we will be talking about this linear approaches or the linear system solver for this matrix, whether it is a direct or iterative approaches.

So, one of the approach is be the Gaussian elimination process when, you try to find out the solution for $Ax = b$ one very common approach is to get the Gaussian elimination and while solving through gauss elimination like $Ax = b$, we see that bound on that error norm would be like this $Ax_{\text{computed}} + e$. So, this should be bounded for this calculation. So, this is quite harder to find than the bounce on $E \|Ax - b\|$, such that this holds exactly this particular property $A + EA$ which is the backward error into x_{computed} plus ϵ . So, one please note here that in many instances the backward errors, which are shown here backward errors are more meaningful than forwarded errors.

If initial data is accurate to certain digit or let us say 4 5 digits for example, then the algorithm for computing x needs not have 10 digits of accuracy. A backward error of order 10^{-4} could be acceptable.

(Refer Slide Time: 05:13)


Error analysis

- ✎ Show for any x, y , there exist $\Delta x, \Delta y$ such that

$$fl(x^T y) = (x + \Delta x)^T y, \quad \text{with } |\Delta x| \leq \gamma_n |x|$$

$$fl(x^T y) = x^T (y + \Delta y), \quad \text{with } |\Delta y| \leq \gamma_n |y|$$
- ✎ (Continuation) Let A an $m \times n$ matrix, x an n -vector, and $y = Ax$. Show that there exist a matrix ΔA such

$$fl(y) = (A + \Delta A)x, \quad \text{with } |\Delta A| \leq \gamma_n |A|$$
- ✎ (Continuation) From the above derive a result about a column of the product of two matrices A and B . Does a similar result hold for the product AB as a whole?


INDIAN INSTITUTE OF TECHNOLOGY KANPUR
Ashoke De 46

(Refer Slide Time: 05:17)

Error analysis

Supplemental notes: Floating Point Arithmetic

In most computing systems, real numbers are represented in two parts: A (mantissa) and an (exponent). If the representation is in the base β then:

$$x = \pm (.d_1 d_2 \dots d_m)_\beta \beta^e$$

$$e = \{-v_e, \dots, +v_e, 0\}$$

- ▶ $.d_1 d_2 \dots d_m$ is a fraction in the base- β representation
- ▶ e is an integer - can be negative, positive or zero.
- ▶ Generally the form is normalized in that $d_1 \neq 0$.



So, this is an very very important note one needs to keep in mind. Now, these are something which one can show so, some more additional notes for floating point arithmetic. So, the some additional notes what it says that for the floating point arithmetic is that, in most of the computing system these are the real numbers which are actually represented in two parts one. So, this is where slowly we are moving towards the hand shaking between our hardware precision or the machine precision with our calculation. So, any real numbers are represented in two parts one called mantissa and the other called exponent.

So, if the representation is in the base beta then x is written as plus minus dot d 1 d 2 d 3 d m small meter to the power beta, where this dot product of or after decibel d 1 d 2 d 3 d m is a faction in the base b representation, this should be you can be able to appreciate this once we move ahead and see some example how you represent this mantissa and exponent component. Now epsilon is an integer which could be positive negative or 0. So, epsilon here would be anything plus, minus or 0 it can be anything. So, another thing is that the form is normalised in that where d 1 is not 0.

(Refer Slide Time: 07:14)

Error analysis

Example: In base 10 (for illustration) → Computing

1. 1000.12345 can be written as $0.100012345_{10} \times 10^4$

2. 0.000812345 can be written as $0.812345_{10} \times 10^{-3}$

► Problem with floating point arithmetic: we have to live with limited precision.

Example: Assume that we have only 5 digits of accuracy in the mantissa and 2 digits for the exponent (excluding sign).

.d ₁	d ₂	d ₃	d ₄	d ₅	e ₁	e ₂
				↑	↑	

INDIAN INSTITUTE OF TECHNOLOGY KANPUR Ashoke De 48

So, as I said we will pick some example to look at the previous calculation or the previous expression of mantissa and exponent for floating point arithmetic, now this is in terms of base 10 the number is 1000 dot 1 2 3 4 5, which can be written as point o point or 0.100012345, base 10 to the power 10 to the power 4. Ok, now similarly 0.000812345 can be written as 0.812345, base 10 to the power minus 3 ok. So, it is just this is I mean again those who are I mean any school going kid also know that these are two numbers and one can write these things

But what is important here we are we are trying to correlate these things, which are somehow thought in at the school level arithmetic calculations. How much that becomes important, when you actually do this large scale programming and numerical algorithm to be converted to the set of instruction for computers. So, this becomes important from the computing point of view. Now problem lies here with the floating point arithmetic and that is where we have to live with limited precision and what does that mean? It means you take an example again assume that, we have only 5 digits of accuracy in the mantissa and 2 digits for the exponent. So, excluding the sign, so now, if you look at that the 5 digit of accuracy. So, after decimal you go d 1 d 2 d 3 d 4 d 5 this actually is this 5 digits.

And then 2 digit for the exponent you go e 1 e 2.

(Refer Slide Time: 10:05)

Error analysis

Let us try to add 1000.2 and 1.07

$1000.2 = \overset{d_1}{.} \overset{d_2}{1} \overset{d_3}{0} \overset{d_4}{0} \overset{d_5}{0} \overset{d_6}{2} \overset{d_7}{0} \overset{d_8}{4} \overset{e_1}{.}$

$1.07 = \overset{d_1}{.} \overset{d_2}{1} \overset{d_3}{0} \overset{d_4}{7} \overset{d_5}{0} \overset{d_6}{0} \overset{d_7}{0} \overset{d_8}{1} \overset{e_2}{.}$


First task: align decimal points. The one with smallest exponent will be (internally) rewritten so its exponent matches the largest one:

$1.07 = 0.000107 \times 10^4$ *rewrite*

Second task: add mantissas:

$$\begin{array}{r} 0.10002 \\ + 0.000107 \\ \hline = 0.100127 \end{array}$$

- d₁ d₂ d₃ d₄ d₅ d₆ ⇒ 6 digits


INDIAN INSTITUTE OF TECHNOLOGY KANPUR
Ashoke De 49

Now let us try to add 1000.2 to 1.07. So, arithmetically one can do very easily like that this is 1000.2, 1.07 this should give me 1001.27 now if you represent them through this kind of mantissa and exponent representation 1000.2 or equals to .1000204 and 1.07 is .1070001. So, as we said these are 5 digits, which corresponds to d₁ d₂ d₃ d₄ d₅ and these are e₁ and e₂ these are exponents similarly here d₁ d₂ d₃ d₄ d₅ epsilon 1 or e₂.

So, what is the initial task you align the decimal points, the one with smallest exponent will be rewritten. So, its exponent matches the largest one this statement is very very crucial here, the one with smallest exponent will be rewritten. So, that it is exponent matches the largest one. So, we have to write 1.07 in such a fashion 0.000107 into 10 to the power 4, now second task is that we add the mantissas. So, mantissa was for this case it was 0.000107 other case .010002 and while doing this, what we have taken that the exponent part has been constant.

So, this number also the exponent would be the 10 to the power 4 here the multiplication now once we rewrite that number then the now it is a multiplication of 10 to the power 4.

(Refer Slide Time: 13:01)

Error analysis

Third task: round result. Result has 6 digits - can use only 5 so we can


▶ Chop result: $\overset{d_1}{.} \overset{d_2}{1} \overset{d_3}{0} \overset{d_4}{0} \overset{d_5}{1} \overset{d_6}{2}$;

▶ Round result: $\overset{d_1}{.} \overset{d_2}{1} \overset{d_3}{0} \overset{d_4}{0} \overset{d_5}{1} \overset{d_6}{3}$; ← rounded result

Fourth task: Normalize result if needed (not needed here)

result with rounding: $\overset{d_1}{.} \overset{d_2}{1} \overset{d_3}{0} \overset{d_4}{0} \overset{d_5}{1} \overset{d_6}{3} \overset{e_1}{0} \overset{e_2}{4}$;

▶ Redo the same thing with $1000.2 + 3000.8$

 INDIAN INSTITUTE OF TECHNOLOGY KANPUR Ashoke De 50

Once you add this mantissas, you get .1000127. So, you find out the result and result has 6 digits and can use only 5. So, we can chop the results. So, what you have got? You got $d_1 d_2 d_3 d_4 d_5 d_6$. So, the digits in mantissa are 6 digits, now our precision as per the problem description we have 5 digits of accuracy in the mantissa.

So, what it will happen even after the addition? You will get a chopped results up to 5 digits. So, $d_3 d_4 d_5$ and where it is getting rounded off it is rounded off.10013 again rounded up to 5 digit, because that is the accuracy that you have and if you look at 1 2 3 4 the fifth one is rounded off because the last 6 digit is greater than 5. So, it is rounded off to this. So, this is the rounded result. So, what you can do now, you can normalise the result if it is required, but particularly for this example you do not need to normalise anything. So, with rounding you can write these are my 5 digit accuracy mantissa $d_1 d_2 d_3 d_4 d_5$ and these are $e_1 e_2$. So, this should be the final results

And if you do that you have already rounded off. So, one can carry out similar exercise with this kind of numbers, but this what happens when you rounding of the numbers at the machine level because machine understands this kind of digits or the bits.

(Refer Slide Time: 15:24)

Error analysis

The IEEE standard

32 bit (Single precision) :

±

← 23 bits →

mantissa

8bits

exponent

(Sign)

POE →

Analysis

Computer

↓ program

↓

Language

↓

Computer

↓

hard

↓

Loss Precision

- ▶ In binary: The leading one in mantissa does not need to be represented. One bit gained. ▶ Hidden bit.
- ▶ Largest exponent: $2^7 - 1 = 127$; Smallest: $= -126$.
[‘bias’ of 127]

INDIAN INSTITUTE OF TECHNOLOGY KANPUR Ashoke De 51

Now when we say that, the machine precision now we bring back certain information from the computing hardware point of view what is the standard IEEE norms. So, if you have 32 bit precision, which is pretty much today not available in most of the desktop kind of computer because everything is more or less 64 bit system, which is a single precision system.

Some laptops still they are available with 32 bit precision, but preliminary now the hardware has been gone to that extent where you can get 64 bit precision of the high end precision, but what happens if you have a 32 bit precision or single precision system

? So, this is the sign of the number 23 bits are actually allocated to the mantissa and 8 bits are allocated to the exponent, 1 bit is allocated for the sign. So, this is how the distribution actually takes place. So, if you just think about as an engineer what you are doing? You are sending some comments to the computer, you are saying that hey you do this, but computer does not understand that thing.

So, that is what you write program. So, through program actually you send that in instruction to the computer, the ones you send that instruction to the computer every program could be written in some programming language, which includes the oldest one is the Fortran then C C plus plus, Java python anything these are all called programming language and programming language has some sort of an handshaking between the

hardware. So, your programming language has some sort of an equate some compilers and compilers actually has some hand shaking with the hardware.

So, what it does your programming language transform your set of instruction. So, you are actually working at this level, where you have taken a PD, you are converted your PD to the basically you converted your PD to a set of descritized equation, then you got a linear system, then you know how to solve the linear system, then you device some set of an series of instruction through programming language you fit that series of instruction to the computer. Computer literally does not understand that it only understand bits. So, those programming language has to be transformed of the machine understanding language.

So, that is where your compiler actually play the mediating role and it actually transform your set of instruction, which is written through the programming language and make it understandable for hardware. It is just like you may not understand this language of the machine hardware similarly machine hardware or the computer does not understand when I talk about computer it is not the everything it is essential, we are talking about pre processor or the processor of the computer, which understands only those machine language. So, your computer compiler actually transform things in the understandable of the hardware and the hardware understand this kind of precision level things

And where you actually carry on your arithmetic operation and feed everything back. So, the error you have committed at this location, this will have an top up cascading as effect at the end solution of your x and that is why we have been doing so much of discussion and on this. So, let us come back to that single precision. So, one bit assign for the sign of that digit, 23 bits allocated for mantissa, 8 bits allocated for exponent. So, this is in binary the leading one in mantissa does not need to be represented one bit gained. So, that is an hidden bit.

(Refer Slide Time: 20:44)

Error analysis

64 bit (Double precision) :

↗
Sign ↘

±

←
→
11 bits

mantissa

exponent

- Bias of 1023 so if e is stored exponent actual exponent is 2^{e-1023}
- $e + bias = 2047$ (all ones) = special use
- Largest exponent: 1023; Smallest = -1022.
- With hidden bit: (mantissa has 53 bits represented.)

INDIAN INSTITUTE OF TECHNOLOGY KANPUR
Ashoke De 52

So, largest exponent could be 2 to the power 7 minus 1127 or smallest one is minus 126. So, this is the biased which, now when you go to the 64 bit precision or rather double precision system, 1 bit again assigned for the sign, 52 bits assigned for mantissa, 11 bits for exponent. So, the bias is 1023. So, if c needs to be stored the exponent actual exponent is 2 to the power c minus 1023. So, which in a mathematical term e plus bias is 2047 that is a use. So, the largest exponent which is possible is 1023, smallest is minus 1022 and there is one hidden bit. So, mantissa has 53 bits represented.

(Refer Slide Time: 21:46)

Error analysis

☑ Take the number 1.0 and see what will happen if you add $1/2, 1/4, \dots, 2^{-i}$. (Do not forget the hidden bit!)

↓
Hidden bit
↘
exponent ↙

←

←
→
52 bits

1	1	0	0	0	0	0	0	0	0	0	0	0	e	e
1	0	1	0	0	0	0	0	0	0	0	0	0	e	e
1	0	0	1	0	0	0	0	0	0	0	0	0	e	e
.....														
1	0	0	0	0	0	0	0	0	0	0	0	1	e	e
1	0	0	0	0	0	0	0	0	0	0	0	0	e	e

➤ Conclusion

$f1(1 + 2^{-52}) \neq 1$ but: $f1(1 + 2^{-53}) == 1 !!$

INDIAN INSTITUTE OF TECHNOLOGY KANPUR
Ashoke De 53

So, this is how your machine understands all this information in that way. So, we can again take an example and then look at these things. So, you take number 1 dot 0 and see

what will happen if you add half 1 by 4 and 2 to the power minus i, do not forget the hidden bit to be accounted for. So, this is the point now hidden bit, this is the hidden bit these are mantissa these are exponent ok. So, that is how you define. So, the conclusion here is that floating point of 1 plus 2 to the power minus 52 not equals to 1, but floating point 1 plus 2 to the power minus 53 should be equals to 1 that is the message.

(Refer Slide Time: 22:59)

Error analysis

ERROR AND SENSITIVITY ANALYSIS FOR SYSTEMS OF LINEAR EQUATIONS

- Conditioning of linear systems. }
- Estimating errors for solutions of linear systems }
- Backward error analysis
- Relative element-wise error analysis }

INDIAN INSTITUTE OF TECHNOLOGY KANPUR Ashoke De 54

So, if you make plus minus this mistakes here that is going to hit you back. Now, we can move back with our discussion in some sort of an error and sensitivity analysis for the linear system. So, upon discussing of all these properties norm, then precision or rather rounding of error, where it comes from and the machine level precision then we look at the sensitivity analysis and error analysis and what you would like to look at is the conditioning of the first linear system, then estimating errors for solution of the linear system. So, though this is going to be a detailed discussion, when you look at different kind of solvers and try to find out the solution for the linear system.

(Refer Slide Time: 24:16)

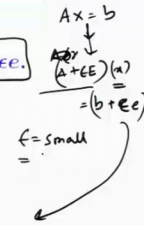
Error analysis

Perturbation analysis for linear systems ($Ax = b$)

Question addressed by perturbation analysis: determine the variation of the solution x when the data, namely A and b , undergoes small variations. Problem is **ill-conditioned** if small variations in data cause very large variation in the solution.

- ▶ Let E , be an $n \times n$ matrix and e be an n -vector.
- ▶ "Perturb" A into $A(\epsilon) = A + \epsilon E$ and b into $b + \epsilon e$.
- ▶ Note: $A + \epsilon E$ is nonsingular for ϵ small enough.
- ☒ Why?
- ▶ The solution $x(\epsilon)$ of the perturbed system is s.t.

$$(A + \epsilon E)x(\epsilon) = b + \epsilon e.$$



But here would like to touch upon the error related analysis for those linear system and then some other error analysis. So, you have a linear system like this $Ax = b$. So, we do some sort of an perturbation analysis for this linear system and how would one do that? You determine the variation of the solution x , when the data undergoes small variations. So, the problem is ill conditioned if small variation in data cause very large variation in the solution. So, this is what leads to the perturbation analysis of the system. So, let us say E be an n by n matrix and small e be an n vector. So, you perturb A into small epsilon

So, this should be equals to A plus epsilon E and b into b plus epsilon e . So, I am perturbing my $Ax = b$ to $(A + \epsilon E)x = b + \epsilon e$, this is the perturb system that we are looking at an; obviously, this is going to be non singular for small enough epsilon. So, epsilon is a very small number for that this would be now solution of $x(\epsilon)$ of the part of system is such that you get this system which I have written here essentially, $(A + \epsilon E)x = b + \epsilon e$ and this solution what you get this you get for the perturbation of amount epsilon.

(Refer Slide Time: 26:26)

Error analysis

$\delta \equiv \text{difference}$

- ▶ Let $\delta(\epsilon) = x(\epsilon) - x$. Then,

$$(A + \epsilon E)\delta(\epsilon) = (b + \epsilon e) - (A + \epsilon E)x = \epsilon(e - Ex)$$

$$\delta(\epsilon) = \epsilon(A + \epsilon E)^{-1}(e - Ex)$$

$(A + \epsilon E)\delta = \epsilon(e - Ex)$
 $\delta = \epsilon(A + \epsilon E)^{-1}(e - Ex)$
- ▶ $x(\epsilon)$ is differentiable at $\epsilon = 0$ and its derivative is

$$x'(0) = \lim_{\epsilon \rightarrow 0} \frac{\delta(\epsilon)}{\epsilon} = A^{-1}(e - Ex)$$
- ▶ A small variation $[\epsilon E, \epsilon e]$ will cause the solution to vary by roughly $\epsilon x'(0) = \epsilon A^{-1}(e - Ex)$
- ▶ The relative variation is such that

$$\frac{\|x(\epsilon) - x\|}{\|x\|} \leq \epsilon \|A^{-1}\| \left(\frac{\|e\|}{\|x\|} + \|E\| \right) + O(\epsilon^2)$$
- ▶ Since $\|b\| \leq \|A\| \|x\|$:

$$\frac{\|x(\epsilon) - x\|}{\|x\|} \leq \epsilon \|A\| \|A^{-1}\| \left(\frac{\|e\|}{\|b\|} + \frac{\|E\|}{\|A\|} \right) + O(\epsilon^2)$$

$\frac{\epsilon}{2} < 1$
 Relative variation

INDIAN INSTITUTE OF TECHNOLOGY KANPUR Ashoke De 56

So, that is the solution for epsilon. So, now you assume the difference. So, delta here actually represents the difference in the solution. So, when you get a perturb solution and the actual solution. So, that is the difference which is represented to delta epsilon then, one can write A plus epsilon E into delta epsilon equals to b plus epsilon e minus A plus epsilon E x. So, which will be we rearrange as epsilon into small e minus e x. So, that is what one can write and here delta epsilon is epsilon A plus epsilon E inverse into e minus epsilon E x. So, this is where you get from. So, you got an system and from here A plus epsilon E into delta equals to epsilon into e minus E x.

So, you get an delta equals to epsilon into A plus epsilon E to the power inverse multiplied with e minus E x. So, that is what you get. So, what you find out next is that the x epsilon is differentiable and it is derivative is going to be x prime 0 with the limit epsilon tends to 0, then this delta epsilon by epsilon is like that. So, whatever delta you have obtained here, now in the limiting sense when epsilon tends to 0, this guy should get me back A multiplied with e minus E x. So, a small variation will cause the solution to vary roughly epsilon in to x prime 0, which is nothing, but epsilon A inverse e e minus x

So, that is what it does and one can estimate by a small perturbation to the system how much your final solution will vary. So, the relative variation must be such that x epsilon minus x the magnitude of that divided by x magnitude of the original solution. So, that is the relative variation must be less than equals to epsilon and the magnitude of A inverse then the normalised e by x and e magnitude of e so, that is a order of.

So, it just come from this particular expression finding the relative variation and one condition which satisfies here is that $\|b\|$ the magnitude of the b is less than equals to $\|a\|$ and $\|x\|$ or the norm of that. So, I can rewrite this particular expression the relative variation is less than equals to ϵ , $\|a\|^{-1} \epsilon$ by $\|b\|$ ϵ by $\|a\|$ order of ϵ^2 . So, you can see once you perturb the initial variation by ϵ how much solution will vary and you can find out the relative variation through this. So, thank you we will discuss or other things in the next lecture.