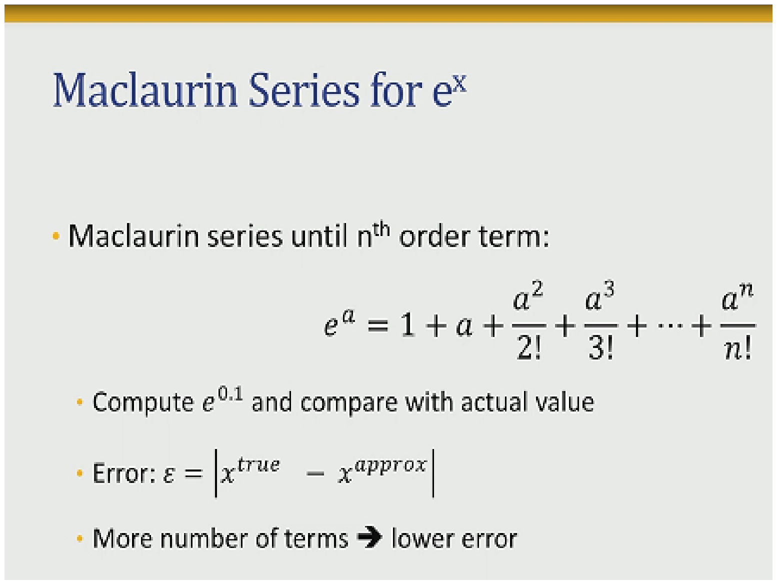
MATLAB Programming for Numerical Computation
Dr.Niket Kaisare
Department of Chemical Engineering
Indian Institute of Technology – Madras

Module No. #02
Lecture No. #2.1
Errors and Approximations – Errors in Numerical Methods

Hello and welcome to this week to of MATLAB programming for numerical computation course. In this module we are going to cover errors and approximations. This is lecture 2.1 and in this lecture we are going to look at where do errors originate in numerical methods. One of the things that happen always, when we talk about numerical methods.

When you use computers for using numerical techniques, you will always encounter errors of various forms. And these errors are going to have 2 basic origins. 1 is what is known as truncation errors and the other is known as the round of errors. Where do this error come from? And how to they affect the overall numerical methods is, something that we will introduce to ourselves in this particular lecture.
(Refer Slide Time 01:06)



Maclaurin Series for e^x

- Maclaurin series until nth order term:

$$e^a = 1 + a + \frac{a^2}{2!} + \frac{a^3}{3!} + \cdots + \frac{a^n}{n!}$$

  - Compute $e^{0.1}$ and compare with actual value
  - Error: $\varepsilon = \left| x^{true} - x^{approx} \right|$
  - More number of terms ➔ lower error

Ok so, let us look at the first term type of an error that is called the truncation error. And what are listed over here is, the Maclaurin series for calculating e to the power x. The Maclaurin series of e to the power x up to nth order term is return as 1 e to the power a. I am sorry, is return as 1 + a + a^2 / 2! + a^3 / 3! and so on.

So what we are going to do is, we are going to compute e to the power 0.1 using multiple number of terms in this Maclaurin series. And we will compare it with the actual value. The error we have, we will define as the difference between the true value and the approximate value. We do not care whether this difference is positive or negative. The error is therefore just the absolute value of the difference.

So it does not matter whether the approximate value is greater than or lesser than the true value as long as it is different than the true value. We want to capture how much that difference is and that is how we define the absolute error epsilon (02:08) okay.

And what we will show using MATLAB is, that if we introduce more number of terms in this Maclaurin series, we are going to get lower error okay. So let us head on over to MATLAB in order to compute this using the Maclaurin series approximation.

(Video Starts: 02:23) So, let us open script file. Let us call it maclaurinExp okay. We will create the file Maclaurin series for exp of 0.1 okay. So let us look at Maclaurin series with different number of terms. So let us say we want to look at terms up to n = 5. So we will say n = 5, our value of a was 0.1 okay and the result that is called as expVal = 1.0 why 1.0 is, the first term over here is 1. (Video Ends: 03:13)

The first order term that is a to the power 1 term is a, a to the power 2 term is a square divided by 2 factorial so on and so forth. So if you look at the first term and compare it with the second term, it is the first term is just multiplied by a/2. If you compare these 2 terms, is just multiplied by a/3 so on and so forth.

(Video Starts: 03:40) So that is what we are going to do. We will call currentTerm as 1.0 and we will use a for loop for i = 1 to n okay. And currentTerm = currentTerm multiplied by a/i okay. So let us see how exactly this works. (Video Ends: 04:06)

So we have e to the power a, that we need to compute approximately. When i is going to be equal to 1, at that time our value of currentTerm is going to be 1 multiplied by a/1. That is a, in the next loop it is going to be a multiplied by a /2. That is a square by 2 factorial. The next term will be a multiplied by a^2/ 2 multiplied by a / 3 that would be a^3 / 3!, so on and so forth okay.

(Video Starts: 04:41) And our expVal = expVal + currentTerm, okay. And I will end this loop over here. So that is going to be the value of expVal. Let us say trueVal = exp (0.1) and error = abs (trueVal – expVal) okay. So we save this and will run. Let us go to MATLAB and run it and we get this result. So, we actually see that the error is 1.4 multiplied by e to the power 10 minus 9. That is the error expVal is 1.1052 and trueVal is also 1.1052 because the error is of the order of n ^ - 9 okay.

Now what let us do is, let us change the particular code. So, that we store all the various values of expVal. So let us say expVal(i) = expVal(i – 1). Let us put it in this way expVal(i +1) = expVal(i + currentTerm). So, what does the first expVal contain? It is it is the 0 ordered approximation (Video Ends: 06:34).

That is just 1. So, that is not really an approximation, expVal 2 is going to have 1 + a, expVal 3 is going to have 1 + a + a square by 2 factorial so on and so forth. When we calculate this error, trueVal is a scalar and expVal is a vector so, we can subtract a scalar from a vector or a vector from a scalar. And we will get a result in vector so this should not be any problem for us. The error will actually end of the 6 dimensional vector, a 1 /6 vector okay.

(Video Starts: 07:06) So let us go and clear okay. And let us now run Maclaurin exp okay. So now we have the error and as you can see the error is 10 ^ -3, 10 ^ -4 and as we go to higher order terms the error keeps decreasing quite significantly.
So what we see over here is, that as we introducing more and more terms, our error keeps decreasing quite significantly okay. Let us look at error 4 which is basically when n = 3, our error is 10 ^ -6. When n = 4 our error is going to be 10 ^ -8. And we have seen already that when we

introduce all 5 terms of the Maclaurin series, our error drops even further okay. (Video Ends: 08:04)

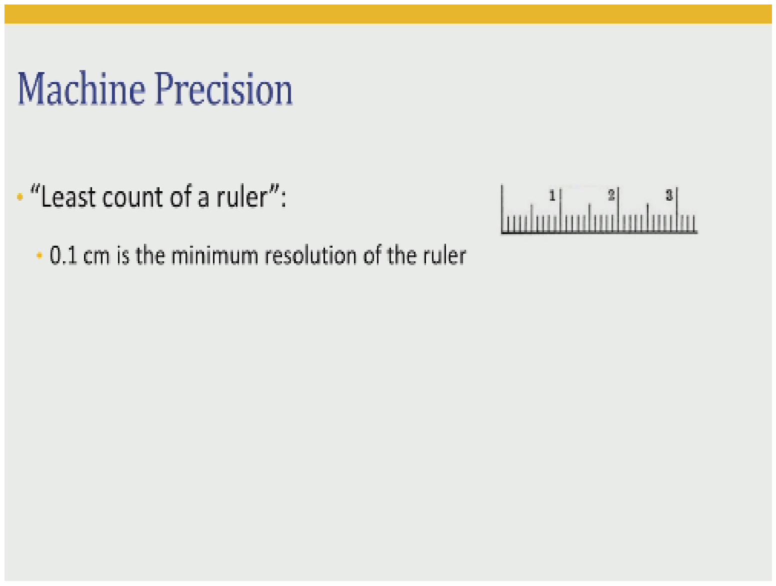
So this is how we have computed, the e to the power a, approximation using Maclaurin series. So what we have seen over here is that as we truncate the overall series to nth order terms we realized that the error keeps on decreasing. So more the number of terms of Maclaurin series we introduce, the smaller is going to be the error okay. Why does the error keep changing or why the error in the first place?

The reason why there is an error in the first place is, because we are truncating this Maclaurin series. So this series is being truncated at the first term or the second term or the third term and so on and so forth. If we truncated just the first order term, we are going to get a greater amount of error. The error was of the order of 10 ^ -3. When we truncated it to second ordered term, we got error of 10 ^ -4.

When we truncated it to the third ordered term, we got the error of the order of 10 ^ -6 so on and so forth. So what we saw over here is that once we introduce greater number of terms, we are going to reduce the error. This is going to be something that we are going to see consistently in this particular course. So if I can summarize these results. What these results are? When you have an infinite series the greater number of terms that you retain in the infinite series, the greater is the accuracy or conversely smaller is the error okay.

In the next lecture we will expand this particular idea to Taylor's series expansion. Taylor's series expansion is indeed where we get this Maclaurin series from. So that particular idea of introducing more number of terms in order to reduce the overall truncation error is going to be a common motive that we are going to follow in the rest of this particular lecture series. Ok so this was the first type of an error which is the truncation error.

(Refer Slide Time: 10:06)



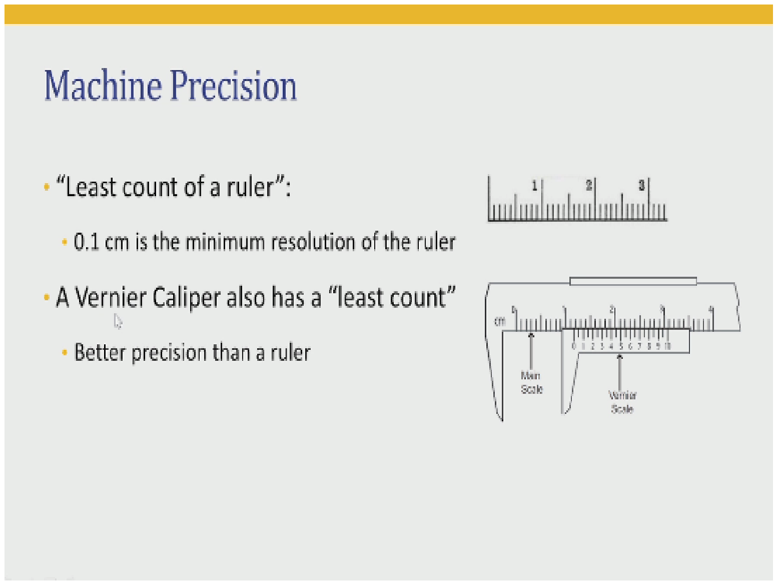
Machine Precision

- “Least count of a ruler”:
  - 0.1 cm is the minimum resolution of the ruler

The second type of an error turns out because of what is known as machine precision. We like to think of a computer as having what is known as an infinite precision. However that is not really true. The machine, the computer also has a machine precision which in some way is like a least count. So let us consider a ruler that I have shown over here. The least count of this ruler is 1 millimeter or 0.1 centimeter.

So that is kind of the minimum resolution of that ruler. You cannot measure length that is less than 1 millimeter. Ok if you want to measure, then with the precision greater than 1 millimeter or a precision value of sub millimeter.

(Refer Slide Time: 10:52)



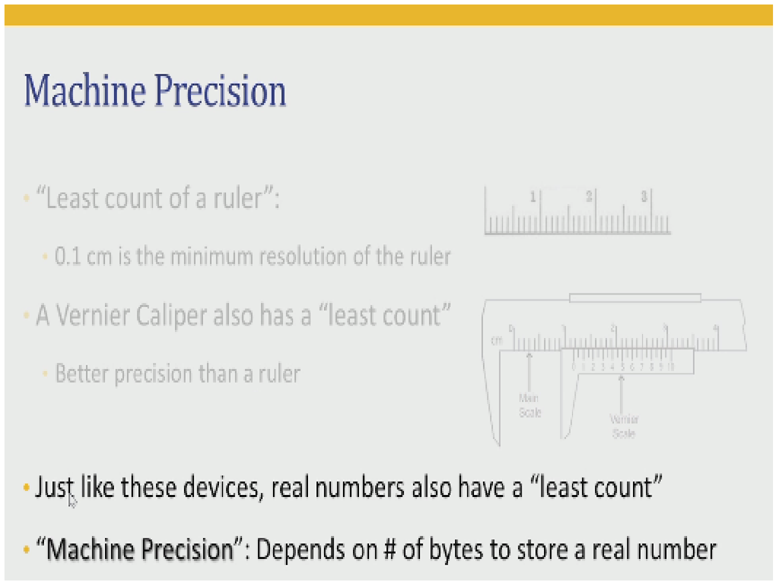
Machine Precision

- “Least count of a ruler”:
  - 0.1 cm is the minimum resolution of the ruler
- A Vernier Caliper also has a “least count”
  - Better precision than a ruler

Main Scale
Vernier Scale

We can for example, use a vernier caliper. This particular example of vernier calipers has the least count of 0.01centimeter, so we can say that this vernier calipers has a better precision than a ruler okay. So precision is basically a term which we can approximately, we can use it in view of the term least count okay. So it is kind of like least count. Now in the same manner in which these 2 devices have a least count our computer is also has a least count.

(Refer Slide Time: 11:27)



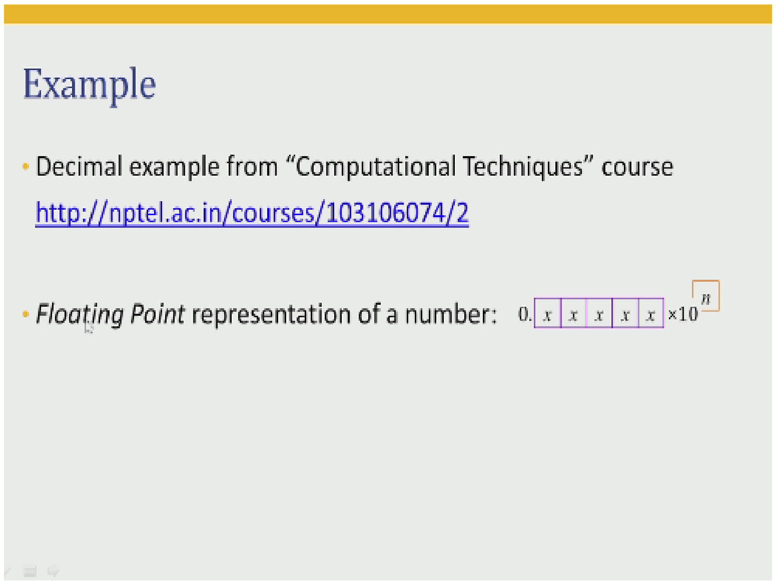
Machine Precision

- "Least count of a ruler":
  - 0.1 cm is the minimum resolution of the ruler
- A Vernier Caliper also has a "least count"
  - Better precision than a ruler
- Just like these devices, real numbers also have a "least count"
- "Machine Precision": Depends on # of bytes to store a real number

1
2
3
cm
Main Scale
Vernier Scale

Just like this devices the real numbers in a computer representation have a least count. So between a number a, and a+epsilon, that does not exist any number and that is what the machine precision is about. It is about the least count of that particular computer okay and that least count is dependent on the number of bytes that we used in order to store a real number. So for example, a standard real number is, uses 4 bytes which is basically 32 bits so that particular real number has a certain precision. A double precision real number which uses 8 bytes or 64 bits has a better machine precision okay.

(Refer Slide Time: 12:16)



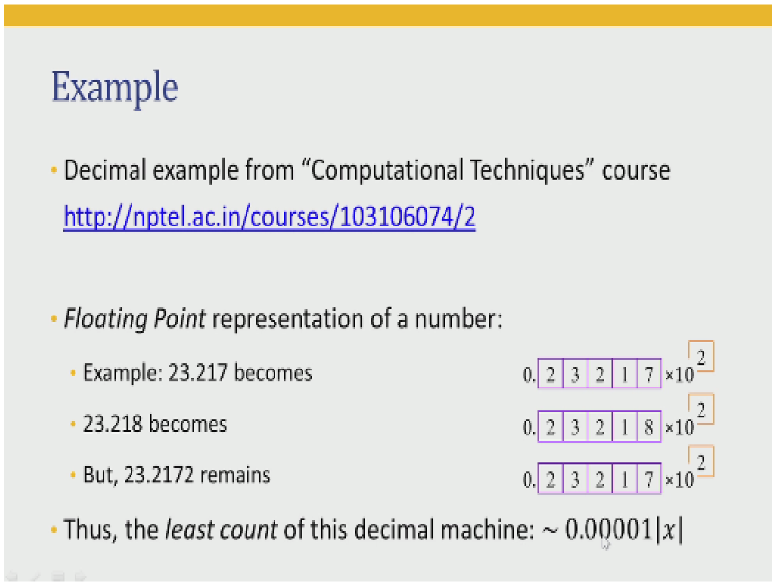
Example

- Decimal example from "Computational Techniques" course http://nptel.ac.in/courses/103106074/2
- *Floating Point* representation of a number: $0.\boxed{x}\boxed{x}\boxed{x}\boxed{x}\boxed{x}\times 10^{\boxed{n}}$

And that representation is based on what is known as the floating point representation of a number for more information on this. You can go to the computational techniques course the link for which is given over here and look at the machine precision and floating point representation. So let us look at floating point representation of a decimal number. Let us say we have 5 boxes in order to store a number and 1 box in order to store an exponent.

The number is always going to be represented as 0 point x x x x x multiplied by 10 ^ n okay. So this is where you we put in those 5 digits of our decimal number so let us see.

(Refer Slide Time: 13:02)



Example

- Decimal example from "Computational Techniques" course http://nptel.ac.in/courses/103106074/2
- *Floating Point* representation of a number:
  - Example: 23.217 becomes $0.\boxed{2}\boxed{3}\boxed{2}\boxed{1}\boxed{7}\times 10^{\boxed{2}}$
  - 23.218 becomes $0.\boxed{2}\boxed{3}\boxed{2}\boxed{1}\boxed{8}\times 10^{\boxed{2}}$
  - But, 23.2172 remains $0.\boxed{2}\boxed{3}\boxed{2}\boxed{1}\boxed{7}\times 10^{\boxed{2}}$
- Thus, the *least count* of this decimal machine: $\sim 0.00001|x|$

How we represent a number of 23.217. The number 23.217 becomes 0.23217 * 10 to the power 2. So what is the next number? That comes in the next number is when the lowest digit over here is incremented. So that becomes the next number becomes 23218 * 10 ^ 2 that is the next number okay. So what if we had a number of the form.

23.2172, the problem is the number 23.2172 cannot be represented with 5 digits of what is known as mantissa and 1 digit exponent. The reason is that number will be represented as 0.23217 multiplied by 10 ^ 2. There is no space to store this trailing digit. As a result this number and this particular number have the exact same representation.

The reason why we are not able to represent a number with a 6 significant digit is, because we do not have enough precision in our decimal machine okay. So the least count of this particular decimal machine turns out to be 10 ^ -5 multiplied by the absolute value of x. So that is the 10 ^ -5 is what is known as the machine precision.

(Video Starts: 14:32) In case of MATLAB, the machine precision is given by a keyword eps and the machine precision in MATLAB is 2 multiplied by 10 ^ -16 okay. This machine precision is 2 ^ -52. (Video Ends: 14:54). The reason is because just the way we had 5 digit representation of this mantissa in this decimal number. Likewise the double precision real number that MATLAB uses which is the IBM standard uses a 52 bit mantissa. So the mantissa has 52 bits in it. So the precision is 2 ^ -52.

Note over here that the precision was 10 ^ -5 okay. So let us look at the case where we have (Video Starts: 15:25) let us call a = 1+ 2 ^ - 52. That is the value of our a and if you subtract a - 1, we are going to extract this particular value okay. So what is happening over here is when we use the value of this precision, that particular number a 1 + 2 * 10 ^ -6 is indeed represented as 1.00015 times followed by a2.

However what if we write a = 1-2 ^ -53, this number is going to be indistinguishable from 1. If we say y a – 1, we are going to get a result of 0 okay. So what does this mean is, this means that our MATLAB has a least count of 2 ^ -16. And that is what is known as the machine precision okay. (Video Ends: 16:25)

So with this, we come to the end of this particular lecture. In this lecture we covered 2 important concepts. The 2 concepts that we covered in this lecture were is that the truncation error because we are going to truncate an infinite series to a finite number of terms. There is going to be an inherent error that is associated with that truncation, the second thing that we realized was that there is a machine precision.

We cannot represent any real number in any form that we want there is going to be a least count with respect to representing those real numbers. So, what does this imply when it comes to numerical techniques, something we are going to cover in the next lecture. Thank you and see in the next lecture.