Numerical Method

Lecture 2

Dejen Ketema

Department of Mathematics
Arba Minch University
https://elearning.amu.edu.et/course/view.php?id=279

Fall 2019

Accuracy and Precision I

- Measurements and calculations can be characterized with regard to their accuracy and precision.

Accuracy and Precision I

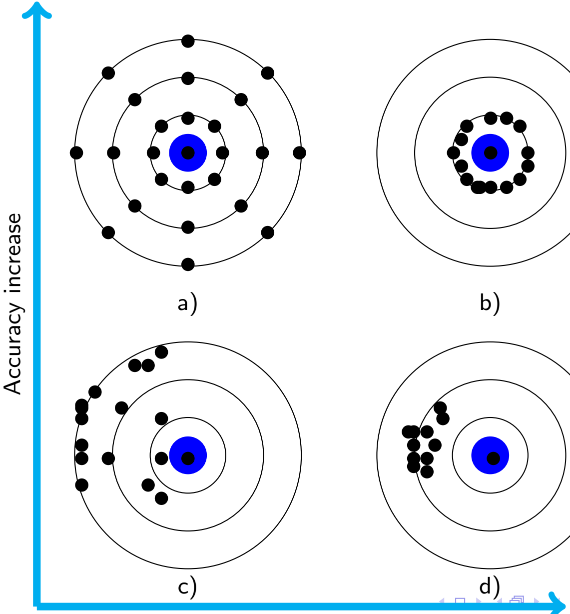
- Measurements and calculations can be characterized with regard to their accuracy and precision.
- Accuracy refers to how closely a value agrees with the true value.
- Precision refers to how closely values agree with each other.

Accuracy and Precision I

- Measurements and calculations can be characterized with regard to their accuracy and precision.
- Accuracy refers to how closely a value agrees with the true value.
- Precision refers to how closely values agree with each other.
- The term error represents the imprecision and inaccuracy of a numerical computation.
- The most fundamental feature of numerical computing is the inevitable presence of error.

Accuracy and Precision I

- Measurements and calculations can be characterized with regard to their accuracy and precision.
- Accuracy refers to how closely a value agrees with the true value.
- Precision refers to how closely values agree with each other.
- The term error represents the imprecision and inaccuracy of a numerical computation.
- The most fundamental feature of numerical computing is the inevitable presence of error.
- The result of any interesting computation will be only approximate,
- and our general quest is to ensure that the resulting error be tolerably small.


Accuracy increase
a)
b)
c)
d)

Description

A) inaccurate and imprecise

B) accurate and imprecise

C) inaccurate and precise

D) accurate and precise

In general, errors can be classified based on their sources as **non-numerical** and **numerical errors.**

INHERENT ERROR/NON-NUMERICAL

- These may be approximation errors in the **mathematical model**. The assumption is that simplification of the problem is worthwhile even if it generates an error in the model.

In general, errors can be classified based on their sources as **non-numerical** and **numerical errors.**

INHERENT ERROR/NON-NUMERICAL

- These may be approximation errors in the **mathematical model**. The assumption is that simplification of the problem is worthwhile even if it generates an error in the model.
- Another typical source of error is error in the **input data**. This may arise, for instance, from physical measurements, which are never infinitely accurate.' Thus, it may occur that after careful numerical solution of a given problem, the resulting solution would not quite match observations on the phenomenon being examined.

In general, errors can be classified based on their sources as **non-numerical** and **numerical errors.**

INHERENT ERROR/NON-NUMERICAL

- These may be approximation errors in the **mathematical model**. The assumption is that simplification of the problem is worthwhile even if it generates an error in the model.
- Another typical source of error is error in the **input data**. This may arise, for instance, from physical measurements, which are never infinitely accurate.' Thus, it may occur that after careful numerical solution of a given problem, the resulting solution would not quite match observations on the phenomenon being examined.
- At the level of numerical algorithms, there is really nothing we can do about such errors. However, they should be taken into consideration, for instance when determining the accuracy to which the numerical problem should be solved.

Such errors arise when an approximate formula is used in place of the actual function to be evaluated.

Theorem (Taylor's Series Theorem:)

Assume that $f(x)$ has $k+1$ derivatives in an interval containing the points x_0 and x_0+h. Then

$$f(x_0+h) = f(x_0)+hf'(x_0)+\frac{h^2}{2}f''(x_0)+\cdots+\frac{h^k}{k!}f^k(x_0)+\frac{h^{k+1]}}{(k+1)!}f^{k+1}f(\xi)$$

where ξ is some point between x_0 and x_0+h.

Roundoff errors

- Roundoff error occurs because of the computing device's inability to deal with certain numbers.
- Such numbers need to be rounded off to some near approximation which is dependent on the word size used to represent numbers of the device.
- A number can be shortened either by **chopping off**, or discarding, the extra digits or by **rounding**.

Roundoff errors

- Roundoff error occurs because of the computing device's inability to deal with certain numbers.
- Such numbers need to be rounded off to some near approximation which is dependent on the word size used to represent numbers of the device.
- A number can be shortened either by **chopping off**, or discarding, the extra digits or by **rounding**.

Truncation Error

- Truncation error refers to the error in a method, which occurs because some series (finite or infinite) is truncated to a fewer number of terms.
- Such errors are essentially algorithmic errors and we can predict the extent of the error that will occur in the method.

Measured error

Absolute Error

Absolute Error is the magnitude of the difference between the true value x and the approximate value x_a. The error between two values is defined as

$$\epsilon_{abs} = \|x - xa\| \quad ,$$

where x denotes the exact value and x_a denotes the approximation.

Relative Error

The relative error is defined as

$$\epsilon_{rel} = \frac{\|x - x_a\|}{\|x\|} \quad .$$

which assumes $x \neq 0$; otherwise relative error is not defined.

Example

Assume Minilik measures a distance 9.99 meter out of 10 meter and Taytu measures 1 centimeter distance out of two centimeter.

1. Find absolute error of Minilik and Taytu
2. Find relative error of Minilik and Taytu
3. Find percentage error of Minilik and Taytu
4. Who one is made highest error

Example

Q In this lengthy example we see how discretization errors and roundoff errors both arise in a simple setting.
Find the approximate solution of sin(1.2)

Example

Q In this lengthy example we see how discretization errors and roundoff errors both arise in a simple setting.
Find the approximate solution of sin(1.2)

A A simple minded algorithm may be constructed using Taylor's series.

$$f(x_0+h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{3!}f'''(x_0) + \cdots \ .$$

Example

Q In this lengthy example we see how discretization errors and roundoff errors both arise in a simple setting.
Find the approximate solution of sin(1.2)

A A simple minded algorithm may be constructed using Taylor's series.

$$f(x_0+h)=f(x_0)+hf'(x_0)+\frac{h^2}{2}f''(x_0)+\frac{h^3}{3!}f'''(x_0)+\cdots\ .$$

- If we know $f''(x_0)$, and it is nonzero, then for h small we can estimate the discretization error by(Correction on handout)

$$\left|\ f'(x_0)-\frac{f(x_0+h)-f(x_0)}{h}\ \right|\approx\left|\frac{h}{2}f''(x_0)\right|.$$

Example

This approximation of $f'(x_0)$ using $h = 0.1$ is not very accurate. We apply smaller and smaller values of h. The resulting errors are as follows:

Example

This approximation of $f'(x_0)$ using $h = 0.1$ is not very accurate. We apply smaller and smaller values of h. The resulting errors are as follows:

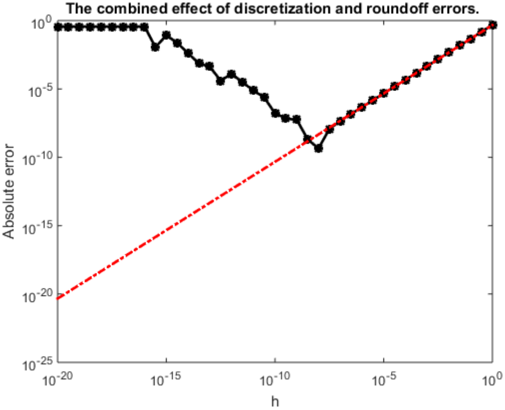
h	Absolute error
0.1	$4.716676e^{-2}$
0.01	$4.666196e^{-3}$
0.001	$4.660799e^{-4}$
$1.e^{-4}$	$4.660256e^{-5}$
$1.e^{-7}$	$4.619326e^{-8}$

Example

This approximation of $f'(x_0)$ using $h = 0.1$ is not very accurate. We apply smaller and smaller values of h. The resulting errors are as follows:

h	Absolute error
0.1	$4.716676e^{-2}$
0.01	$4.666196e^{-3}$
0.001	$4.660799e^{-4}$
$1.e^{-4}$	$4.660256e^{-5}$
$1.e^{-7}$	$4.619326e^{-8}$
$1.e^{-9}$	$5.594726e^{-8}$
$1.e^{-10}$	$1.669696e^{-7}$
$1.e^{-11}$	$7.938531e^{-6}$
$1.e^{-13}$	$6.851746e^{-4}$
$1.e^{-15}$	$8.173146e^{-2}$
$1.e^{-16}$	$3.623578e^{-1}$

The combined effect of discretization and roundoff errors.



Round-off errors

- Consider the two nearly equal numbers $p = 9890.9$ and $q = 9887.l$.
- Use decimal floating point representation (scientific notation) with three significant digits in the mantissa to calculate the difference between the two numbers, $(p - q)$.
- Do the calculation first by using chopping and then by using rounding.

ROUND-OFF ERRORS

- Consider the two nearly equal numbers $p = 9890.9$ and $q = 9887.l$.
- Use decimal floating point representation (scientific notation) with three significant digits in the mantissa to calculate the difference between the two numbers, $(p - q)$.
- Do the calculation first by using chopping and then by using rounding.

SOLUTION

In decimal floating point representation, the two numbers are:

$$p = 9.8909 \times 10^3 \text{ and } q = 9.8871 \times 10^3$$

If only three significant digits are allowed in the mantissa, the numbers have to be shortened. If chopping is used, the numbers become:

$$p = 9.890 \times 10^3 \text{ and } q = 9.887 \times 10^3$$

Solution

Using these values in the subtraction gives:

$$p - q = 9.890 \times 10^3 - 9.887 \times 10^3 = 0.003 \times 1000 = 3$$

Solution

Using these values in the subtraction gives:

$$p - q = 9.890 \times 10^3 - 9.887 \times 10^3 = 0.003 \times 1000 = 3$$

If rounding is used, the numbers become:

$$p = 9.891 \times 10^3 \quad \text{and } q = 9.887 \times 10^3 (q \text{ is the same as before})$$

Using these values in the subtraction gives:

$$p - q = 9.891 \times 10^3 - 9.887 \times 10^3 = 0.004 \times 10^3 = 4$$

SOLUTION

Using these values in the subtraction gives:

$$p - q = 9.890 \times 10^3 - 9.887 \times 10^3 = 0.003 \times 1000 = 3$$

If rounding is used, the numbers become:

$$p = 9.891 \times 10^3 \text{ and } q = 9.887 \times 10^3 (q \text{ is the same as before})$$

Using these values in the subtraction gives:

$$p - q = 9.891 \times 10^3 - 9.887 \times 10^3 = 0.004 \times 10^3 = 4$$

The true (exact) difference between the numbers is 3.8. These results show that, in the present problem, rounding gives a value closer to the true answer.

Errors in Numerical Solutions

Example

Rounding 5-digit arithmetic

$$(0.96384.10^5 + 0.26678.10^2) - 0.96410.10^5 =$$

$$(0.96384.10^5 + 0.00027.10^5) - 0.96410.10^5 =$$

$$0.96411.10^5 - 0 : 96410.10^5 = 0.10000.10^1$$

Truncating 5-digit arithmetic

$$(0.96384.10^5 + 0.26678.10^2) - 0.96410.10^5 =$$

$$(0.96384.1065 + 0.00026.10^5) - 0.96410.^{1}05 =$$

$$0.96410.10^5 - 0.96410.10^5 = 0.0000.10^0$$

Rearrangement changes the result:

$$(0.96384.10^5 - 0.96410.10^5) + 0.26678.10^2 =$$

$$-0.26000.10^2 + 0.26678.10^2 = 0.67800.10^0$$

Example

Rounding 5-digit arithmetic

$$(0.96384.10^5 + 0.26678.10^2) - 0.96410.10^5 =$$
$$(0.96384.10^5 + 0.00027.10^5) - 0.96410.10^5 =$$
$$0.96411.10^5 - 0 : 96410.10^5 = 0.10000.10^1$$

Rearrangement changes the result:

$$(0.96384.10^5 - 0.96410.10^5) + 0.26678.10^2 =$$
$$-0.26000.10^2 + 0.26678.10^2 = 0.67800.10^0$$

Example

Rounding 5-digit arithmetic

$$(0.96384.10^5 + 0.26678.10^2) - 0.96410.10^5 =$$
$$(0.96384.10^5 + 0.00027.10^5) - 0.96410.10^5 =$$
$$0.96411.10^5 - 0 : 96410.10^5 = 0.10000.10^1$$

Truncating 5-digit arithmetic

$$(0.96384.10^5 + 0.26678.10^2) - 0.96410.10^5 =$$
$$(0.96384.1065 + 0.00026.10^5) - 0.96410.^105 =$$
$$0.96410.10^5 - 0.96410.10^5 = 0.0000.10^0$$

Rearrangement changes the result:

$$(0.96384.10^5 - 0.96410.10^5) + 0.26678.10^2 =$$
$$-0.26000.10^2 + 0.26678.10^2 = 0.67800.10^0$$

Subtraction Error

Consider the MatLab computation near x = 1 of

$$y = x^7 - 7x^6 + 21x^5 - 35x^4 + 35x^3 - 21x^2 + 7x - 1$$

compared to $y = (x-1)^7$

$$y = x^7 - 7x^6 + 21x^5 - 35x^4 + 35x^3 - 21x^2 + 7x - 1 = (x-1)^7$$

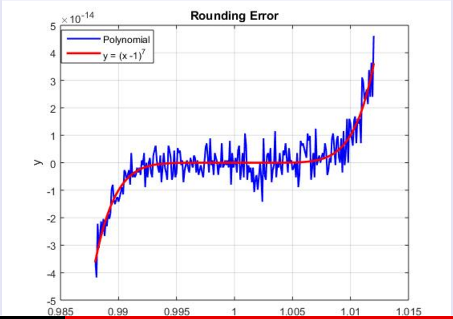
Subtraction Error

Consider the MatLab computation near x = 1 of

$$y = x^7 - 7x^6 + 21x^5 - 35x^4 + 35x^3 - 21x^2 + 7x - 1$$

compared to $y = (x-1)^7$

$$y = x^7 - 7x^6 + 21x^5 - 35x^4 + 35x^3 - 21x^2 + 7x - 1 = (x-1)^7$$



Algorithm

An algorithm is a procedure that describes, in an unambiguous manner, a finite sequence of steps to be performed in a specific order.

ALGORITHM

An algorithm is a procedure that describes, in an unambiguous manner, a finite sequence of steps to be performed in a specific order.

STABILITY

An algorithm is said to be stable if small changes in the input, generates small changes in the output.

Algorithm

An algorithm is a procedure that describes, in an unambiguous manner, a finite sequence of steps to be performed in a specific order.

Stability

An algorithm is said to be stable if small changes in the input, generates small changes in the output.

Convergence

Suppose the sequence $\underline{\beta} = \{\beta_n\}_{n=0}^{\infty}$ converges to zero, and $\underline{\alpha} = \{\alpha_n\}_{n=0}^{\infty}$ converges to a number α. If there exists $K > 0 : |\alpha_n - \alpha| < K\beta_n$, for n large enough, then we say that $\{\alpha_n\}_{n=0}^{\infty}$ converges to α with a Rate of Convergence $O(\beta_n)$ ("Big Oh of β_n"). We write

$$\alpha_n = \alpha + O(\beta_n)$$

STABILITY CONDITIONS

- In general, it is impossible to prevent linear accumulation of roundoff errors during a calculation, and this is acceptable if the linear rate is moderate (i.e., the constant c_0 below is not very large).
- But we must prevent exponential growth! Explicitly, if E_n measures the relative error at the n^{th} operation of an algorithm, then

Stability conditions

- If $E_n \simeq CnE_0$ (for a constant C, which is independent of n), then the growth is **linear.**

Stability conditions

- If $E_n \simeq CnE_0$ (for a constant C, which is independent of n), then the growth is **linear.**
- If $E_n \simeq C^n E_0, C > 1$, then the growth is **exponential** in this case the error will dominate very fast(undesirable scenario).

Stability conditions

- If $E_n \simeq CnE_0$ (for a constant C, which is independent of n), then the growth is **linear.**
- If $E_n \simeq C^n E_0, C > 1$, then the growth is **exponential** in this case the error will dominate very fast(undesirable scenario).
- Linear error growth is usually unavoidable, and in the case where C and E_0 are small the results are generally acceptable. Stable algorithm.

Stability conditions

- If $E_n \simeq CnE_0$ (for a constant C, which is independent of n), then the growth is **linear.**
- If $E_n \simeq C^n E_0$, $C > 1$, then the growth is **exponential** in this case the error will dominate very fast(undesirable scenario).
- Linear error growth is usually unavoidable, and in the case where C and E_0 are small the results are generally acceptable. Stable algorithm.
- Exponential error growth is unacceptable. Regardless of the size of E_0 the error grows rapidly. Unstable algorithm.

Stability conditions

- If $E_n \simeq CnE_0$ (for a constant C, which is independent of n), then the growth is **linear.**
- If $E_n \simeq C^n E_0, C > 1$, then the growth is **exponential** in this case the error will dominate very fast(undesirable scenario).
- Linear error growth is usually unavoidable, and in the case where C and E_0 are small the results are generally acceptable. Stable algorithm.
- Exponential error growth is unacceptable. Regardless of the size of E_0 the error grows rapidly. Unstable algorithm.
- One property of chaos in a dynamical system is the exponential growth of any error in initial conditions leading to **unpredictable behavior**