September 2, 2014

Chapter 1: Numerical Algorithms

Uri M. Ascher and Chen Greif
Department of Computer Science
The University of British Columbia
{ascher,greif}@cs.ubc.ca

Slides for the book
A First Course in Numerical Methods (published by SIAM, 2011)
`http://bookstore.siam.org/cs07/`

Goals of this chapter

- To explain what numerical algorithms are;
- to describe various sources and types of errors and how to measure them;
- to discuss algorithm properties and explain in a basic manner the notions of conditioning and stability;
- to illustrate the potentially damaging effect of roundoff errors.

Outline

- Scientific computing
- Numerical algorithms and errors
- Algorithm properties

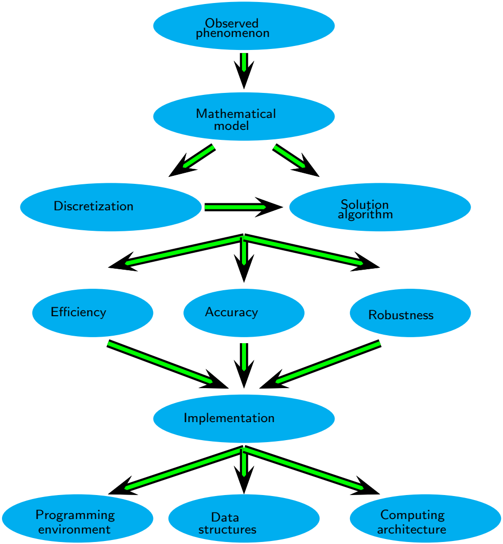


FIGURE: Scientific computing.

Methodology

- Approach the study of numerical algorithms by studying errors.
  Not really glorious, but useful.
- Problem solving environment: MATLAB

Outline

- Scientific computing
- Numerical algorithms and errors
- Algorithm properties

How to measure errors

- Can measure errors as absolute or relative, or a combination of both.
- The *absolute error* in v approximating u is $|u - v|$.
- The *relative error* (assuming $u \neq 0$) is $\frac{|u-v|}{|u|}$.

u	v	Absolute Error	Relative Error
1	0.99	0.01	0.01
1	1.01	0.01	0.01
-1.5	-1.2	0.3	0.2
100	99.99	0.01	0.0001
100	99	1	0.01

Source and type of errors

- Errors in the problem to be solved
  - In the mathematical model (an approximation to reality)
  - In input data
- Approximation errors (in the numerical algorithm)
  - Discretization errors
  - Convergence errors
  - Roundoff errors

Example

Given smooth function $f(x)$, approximate derivative at some point $x = x_0$:

$$f'(x_0) \approx \frac{f(x_0 + h) - f(x_0)}{h},$$

for a small parameter value h.

Discretization error:

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

Results

Try for $f(x) = \sin(x)$ at $x_0 = 1.2$.
(So we are approximating $\cos(1.2) = 0.362357754476674...$.)

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

These results reflect the discretization error as expected.

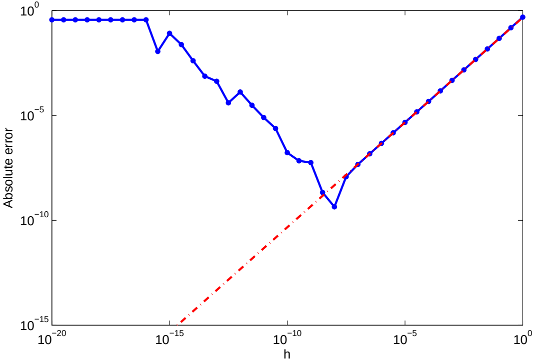
Results for smaller h

h	Absolute error
1.e-8	4.361050e-10
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

These results reflect both discretization and roundoff errors.

Run program Example1_3Figure1_3.m

Results for all h



The solid curve interpolates the computed values of $|f'(x_0) - \frac{f(x_0+h)-f(x_0)}{h}|$ for $f(x) = \sin(x),\ x_0 = 1.2$. Shown in dash-dot style is a straight line depicting the discretization error without roundoff error.

Outline

- Scientific computing
- Numerical algorithms and errors
- Algorithm properties

Algorithm Properties

- Accuracy
- Efficiency (surprisingly hard to measure)
- Robustness

Problem conditioning and algorithm stability

Qualitatively speaking:

- The problem is ill-conditioned if a small perturbation in the data may produce a large difference in the result.
  The problem is well-conditioned otherwise.
- The algorithm is stable if its output is the exact result of a slightly perturbed input.

Next, we'll see how bad roundoff error accumulation can be when an unstable algorithm is used.

Unstable algorithm: an extreme example

Problem statement: evaluate the integrals

$$y_n = \int_0^1 \frac{x^n}{x+10} dx$$

for $n = 1, 2, \ldots, 30$.

Algorithm development: observe that analytically

$$y_n + 10y_{n-1} = \int_0^1 \frac{x^n + 10x^{n-1}}{x+10} dx = \int_0^1 x^{n-1} dx = \frac{1}{n}.$$

Also

$$y_0 = \int_0^1 \frac{1}{x+10} dx = \ln(11) - \ln(10).$$

Algorithm:

- Evaluate $y_0 = \ln(11) - \ln(10)$.
- For $n = 1, \ldots, 30$, evaluate $y_n = \frac{1}{n} - 10\, y_{n-1}$.

Run program Example1_6.m

Roundoff error accumulation

- In general, if E_n is error after n elementary operations, cannot avoid linear roundoff error accumulation

$$E_n \simeq c_0 n E_0.$$

- Will not tolerate an **exponential** error growth such as

$$E_n \simeq c_1^n E_0 \quad \text{for some constant} \quad c_1 > 1$$

  – an **unstable algorithm**.