Math 298
Fundamental Concepts in Computational and Applied Mathematics

Juan Meza
School of Natural Sciences
University of California, Merced

Course Goals

- Introduce fundamental concepts in computational and applied mathematics.
- Highlight some of the classic NA papers and algorithms.
- Demonstrate use of these concepts in real world applications.
- We will not go into any one area in depth.

Learning Outcomes

- Be familiar with key mathematical concepts used in developing numerical algorithms.
- Understand some of the basic skills and resources necessary to start research in computational and applied mathematics.
- Be aware of basic communications skills needed to present mathematics clearly to a broad audience in writing and in speaking.

Ground Rules

- Class participation is critical to getting the most out of this course
- Some assigned readings
- Presentations at the end of the semester

Theory vs. Practice

- In theory there is no difference between theory and practice.

Theory vs. Practice

- In theory there is no difference between theory and practice.
- In practice there is.

Phil Colella's 7 Dwarfes

1. Dense linear algebra
2. Sparse line algebra
3. Spectral methods (Fast Fourier transform)
4. N-body methods (Particle)
5. Structured grids
6. Unstructured grids
7. Monte Carlo

Defining Software Requirements for Scientific Computing, P. Colella, DARPA presentation, 2004.

13 Motifs – Patterson et al.

1. Dense linear algebra
2. Sparse line algebra
3. Spectral methods
4. N-body methods
5. Structured grids
6. Unstructured grids
7. MapReduce
8. Combinational Logic
9. Graph Traversal
10. Dynamic Programming
11. Backtrack and Branch-and-Bound
12. Graphical Models
13. Finite State Machines

Nick Trefethen's 13 Classic Numerical Analysis Papers.

1. Cover some of the most important NA papers in the last 40 years
2. Cover most of the areas mentioned above
3. You should have a passing familiarity with all of them
4. Note: Copies have all of these have been uploaded into CROPS

Top 10 Algorithms of the Century

1946 Monte Carlo

1947 Simplex method for LP

1950 Conjugate gradient (Krylov subspace iteration methods)

1951 Decompositional approach to matrix computations

1957 Fortran compiler

1959 QR for computing eigenvalues

1962 Quicksort

1965 Fast Fourier transform

1977 PSLQ (integer relation detection algorithm)

1987 Fast Multipole

Finite Precision

- Difference between infinite precision and computer arithmetic
- You will always make an error when approximating a mathematical object on a computer
- One of the major types of errors is discretization(truncation), which is the difference between the solution of the discrete problem and the exact solution of the mathematical object
- Example: Suppose you have a function defined by an infinite series and you "truncate" it at some point, e.g.

$$\begin{aligned}\exp(x) &= 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots \\ &\approx 1 + x\end{aligned}$$

Finite Precision

- Example

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

- How do we pick h to get the best approximation?

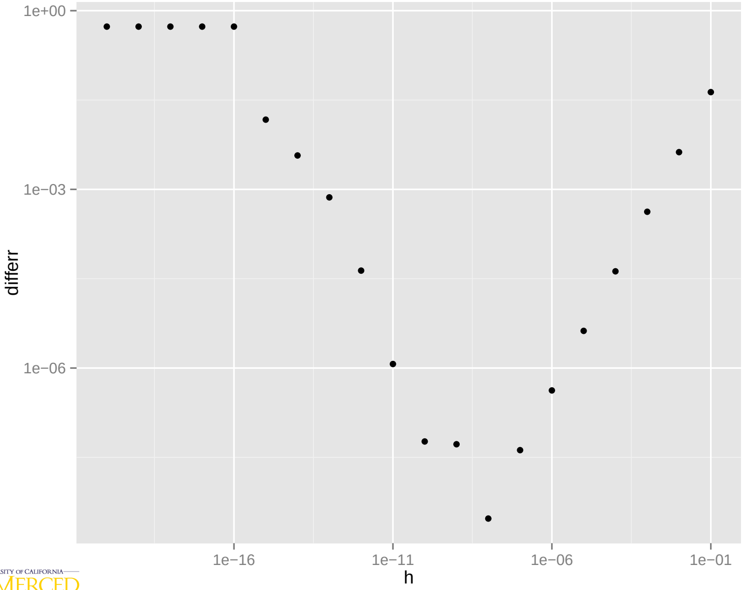
Finite Precision

- Example

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

- How do we pick h to get the best approximation?
- Discussion

Finite Precision Example



Finite Precision

- Example

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

- Lesson: Don't solve an approximate problem too exactly
- Rule of Thumb: Choose h to perturb about half of the digits in x.

Finite Precision Real World Examples

Applications requiring extra precision

- Climate Modeling
- Supernova Simulations
- Coulomb N-Body Atomic System Simulations
- Computational Geometry and Grid Generation
- Many, many more, . . .

Reference: **High-Precision Floating-Point Arithmetic in Scientific Computation**, David H. Bailey, Computing in Science & Engineering, IEEE, May/June 2005.

Conditioning

- Suppose we would like to know how perturbing x will affect the value of $y = f(x)$
- Define
$$\kappa_f(x) = \frac{|x| \cdot |f'(x)|}{|f(x)|}$$
- Then
$$\frac{|\hat{y} - y|}{|y|} \approx \kappa_f(x) \cdot \frac{|\hat{x} - x|}{|x|}$$
- Fact: $-\log_{10}(\text{relative error}) \approx \#$ decimal digits to which two numbers agree!

Rule of thumb: You will lose approximately $\log_{10}(\kappa_f(x))$ decimal digits

Stability

- Informal definition of stability. An algorithm for computing $f(x)$ is *stable* if it returns a $\hat{y}$ that satisfies:

$$\frac{|\hat{y}-y|}{|y|} \approx \kappa_f(x) \cdot \frac{|\hat{x}-x|}{|x|}$$

- Question: What do you think of the following algorithm for computing $\exp(x)$?

Stability

- Informal definition of stability. An algorithm for computing $f(x)$ is *stable* if it returns a $\hat{y}$ that satisfies:

$$\frac{|\hat{y} - y|}{|y|} \approx \kappa_f(x) \cdot \frac{|\hat{x} - x|}{|x|}$$

- Question: What do you think of the following algorithm for computing $\exp(x)$?

$$\exp(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

Stability

- Informal definition of stability. An algorithm for computing $f(x)$ is *stable* if it returns a $\hat{y}$ that satisfies:

$$\frac{|\hat{y} - y|}{|y|} \approx \kappa_f(x) \cdot \frac{|\hat{x} - x|}{|x|}$$

- Question: What do you think of the following algorithm for computing $\exp(x)$?

$$\exp(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

- Discussion

Summary

- Beware catastrophic cancellation - know limits of precision
- Ask about conditioning of a problem
- Choose stable algorithms so as to not introduce any more loss of precision than is necessary
- Conditioning is fundamentally a characteristic of the problem, while stability is related to algorithms.

References

- **High-Precision Floating-Point Arithmetic in Scientific Computation**, David H. Bailey, Computing in Science & Engineering, IEEE, May/June 2005.
- **Numerical Computing with IEEE Floating Point Arithmetic**, Michael L. Overton, SIAM, 2001.
- **What Every Computer Scientist Should Know About Floating Point Arithmetic**, David Goldberg, Computing Surveys, ACM, March 1991.