Introduction to Scientific Computing

Steve Marschner

Cornell CS 322

Outline

Numerical Methods

Course mechanics

Course content

What CS 322 is about

The title says "scientific computing."

The course catalog copy says "numerical analysis."

I'll often call it "numerical methods."

"Scientific computing"

Using computers for scientific, medical, and engineering applications.
Some examples:

- X-ray crystallography for protein structure
- Structural analysis of machine parts
- Analysis of data from a clinical study

An application-focused description.

"Numerical analysis"

Solving problems in analysis (i.e., with real numbers) by numerical, rather than symbolic means. Some examples:

- Solving linear and nonlinear equations
- Computing values of definite integrals
- Solving differential equations

A problem-focused description.

"Numerical methods"

Algorithms (methods) for solving mathematical problems in the service of applications. Some examples:

- The method of conjugate gradients for linear systems.
- Gaussian quadrature for computing integrals.
- The Runge-Kutta method for initial value problems.

An algorithm-focused description.

In reality this area encompasses all three: based on our application, we formulate some mathematical problem, and find an algorithm for it.

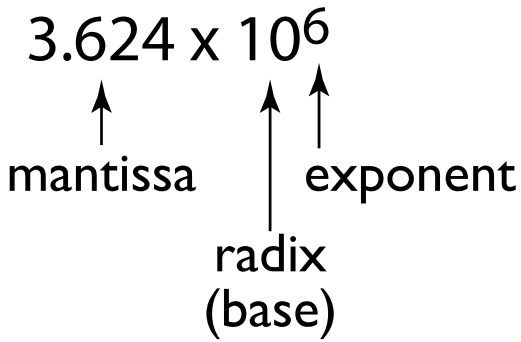
What are numerical methods?

The application of computer science to solving math problems.

- usually problems about real numbers
  - linear algebra
  - calculus/differential equations
  - geometry
- not always *arising from* problems with real numbers
  - PageRank
- computing using approximations
  - most interesting NA problems cannot be computed exactly
  - example of degree 5 polynomial

Floating point numbers

Numerical programs almost always use floating point arithmetic, and understanding how floating point works is crucial.



- fixed precision in mantissa
  $\longrightarrow$ too-close numbers can't be distinguished
- limited range of exponent
  $\longrightarrow$ too-small numbers can't be distinguished from zero
  $\longrightarrow$ too-big numbers can't be represented

Numerical methods vs. "standard" CS

Numerical methods:

- Deal with floating point approximations of real numbers
- Implement calculus, linear algebra, etc.
- Are approximate

Many other programs:

- Deal with integers and other discrete data
- Implement logic, finite algebra, etc.
- Are exact

Each is tricky to get right and has its own characteristic types of bugs.

Numerical programming seems hard at first because the bugs are unfamiliar.

Numerical methods vs. "standard" CS

an example

Binary search:

- Searching the phone book by binary search
  - Start with whole array (finiteness helps)
  - Compare search key with value at middle index (watch for off-by-one bugs)
  - Move endpoint one past compared value
  - Done when strings are equal or interval is empty
- Searching for the zero of a function by binary search
  - Start with bracketing interval (continuity helps)
  - Check sign of function value at midpoint (watch for rounding errors)
  - Move endpoint to midpoint
  - Done when function value is small enough or interval is small enough

Course Mechanics

See the web site for details on these topics.

- Meetings
- Homework
- Projects
- Grading system
- Exams
- Textbooks
- Web site

What you will learn in CS 322

For each of the classes of problems we'll study, I hope you will learn:

- How to recognize it
  - so that you know where to look for methods to solve it
- The workings of some basic methods to solve the problem
  - so you can code them up for easy problems
  - so you understand where the more advanced methods are coming from
- What can go wrong
  - so you know what to watch out for
- How to tell whether you have a hard or easy problem
  - so you know when you need a more advanced method
  - so you know how to choose an appropriate method

What you will learn in CS 322

Along the way I hope you will absorb:

- Debugging intuition for numerical code
  (like you already have for more discrete code)
  $\longrightarrow$ the "red flags" of floating point numbers
- Appreciation for working with numerical data
  $\longrightarrow$ how to get a handle on it
  $\longrightarrow$ how to explain it using plots and statistics
- The culture of numerical computing
  (names for things, recurring algorithmic features)
- Deeper understanding of the math

Course topics

- Getting started
  - linear systems I
  - fitting I
  - numerical programming
  - data visualization
- Low-dimensional methods
  - interpolation and approximation
  - root finding
  - quadrature
- Larger-scale methods
  - singular value decomposition
  - statistics of fitting problems
  - ordinary differential equations
  - linear systems II
  - fitting II

Preliminary schedule is on the course web site.

What you won't learn in CS322

- How to implement advanced methods (really good implementation is subtle!)
- Much about more advanced areas
  - $\longrightarrow$ optimization
  - $\longrightarrow$ partial differential equations
  - $\longrightarrow$ large-scale sparse problems

Lecture summary

- Numerical methods: programs that compute with approximations of real numbers
- Course content: innards of basic methods; using of numerical software
- Course mechanics: homeworks and projects; 5 point scale; starred problems.

Next time: applications of numerical methods, standard problem types, and what we want from our programs (never mind what they want from us).

`http://www.cs.cornell.edu/Courses/cs322/2007sp/`