Lecture 1: Introduction to CS 3220

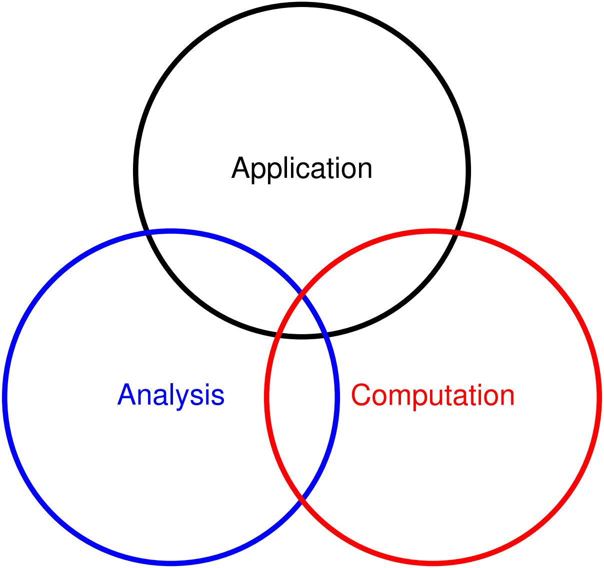
David Bindel

24 Jan 2011

Plan for today

- What is scientific computing about?
- What is this class about?
- Workload and prerequisites
- Logistics and course policies

The Big Computational Science & Engineering Picture



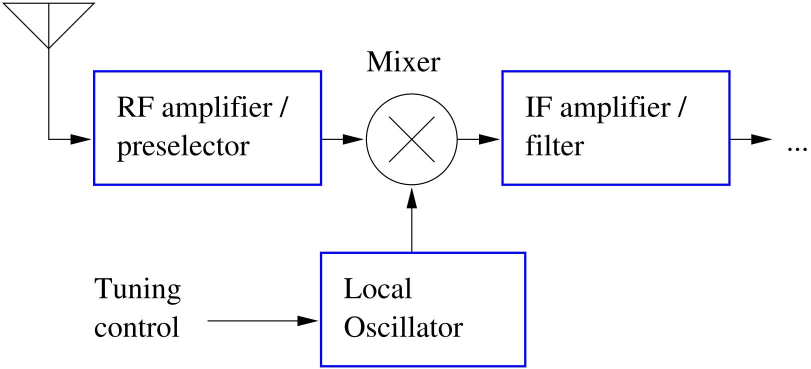
Applications Everywhere!

These tools are used in more places than you might think:

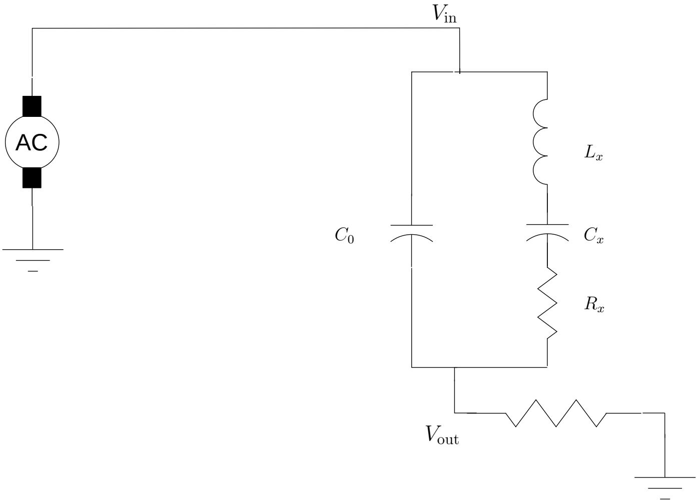
- Climate modeling
- CAD tools (computers, buildings, airplanes, ...)
- Control systems
- Computational biology
- Computational finance
- Machine learning and statistical models
- Game physics and movie special effects
- Medical imaging
- Information retrieval
- ...

Application: Better Devices

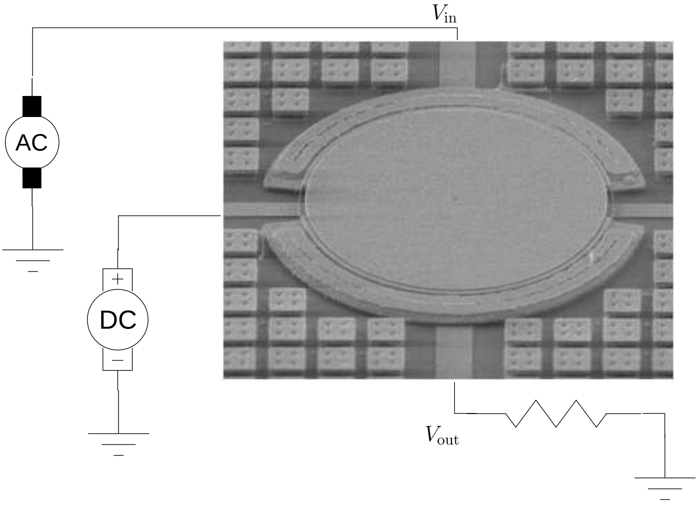
The Mechanical Cell Phone



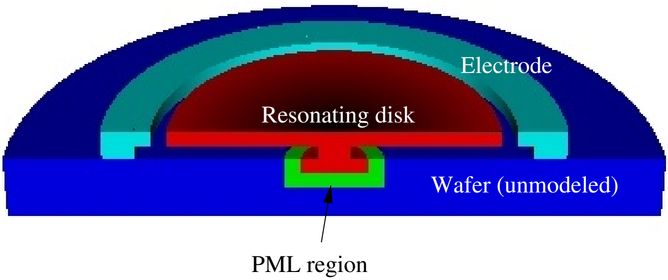
A Simple Circuit



An Electromechanical Circuit



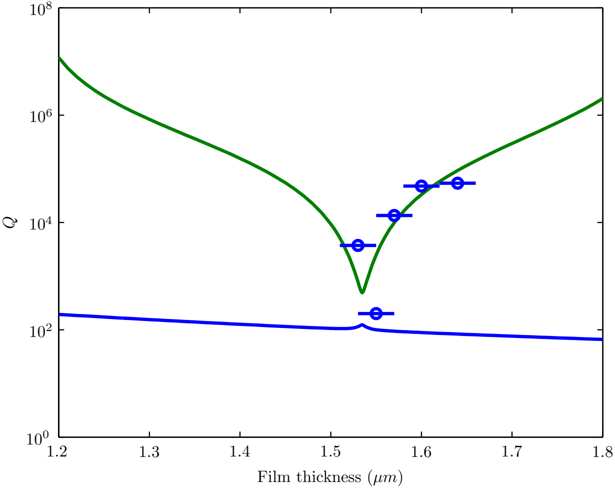
Modeling Damping and Radiation



Ingredients:

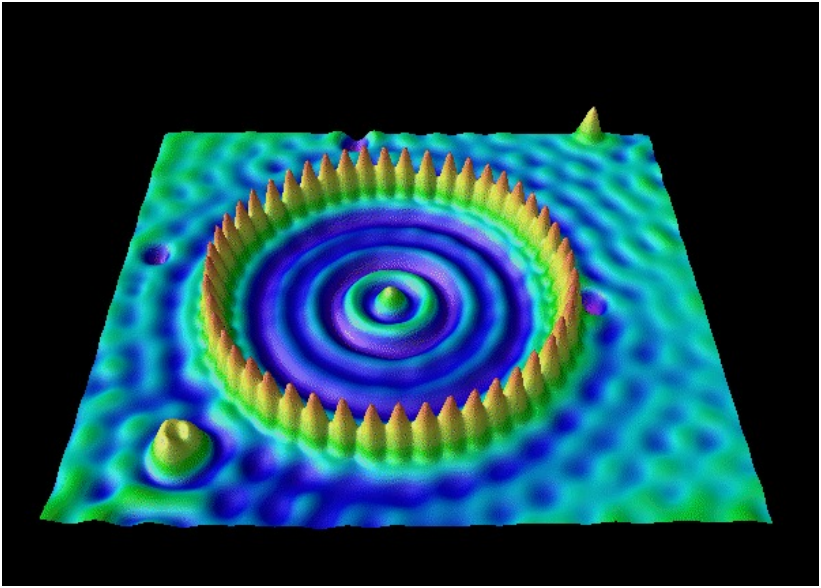
- Physics: Radiation, thermoelasticity
- Numerics: Structured eigensolvers, model reduction
- Software: HiQLab

Damping: Devil in the Details!

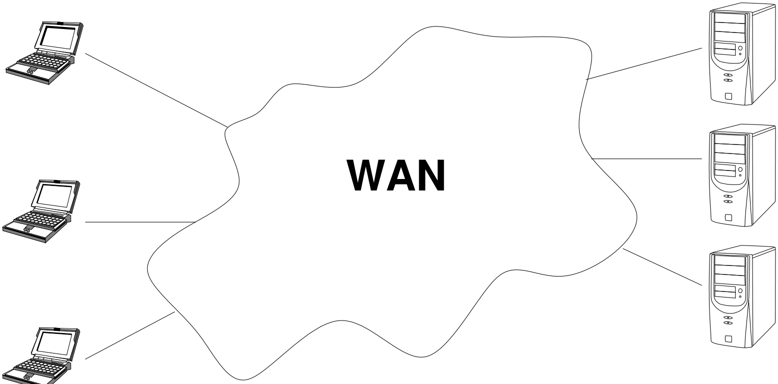


Simulation and lab measurements vs. disk thickness

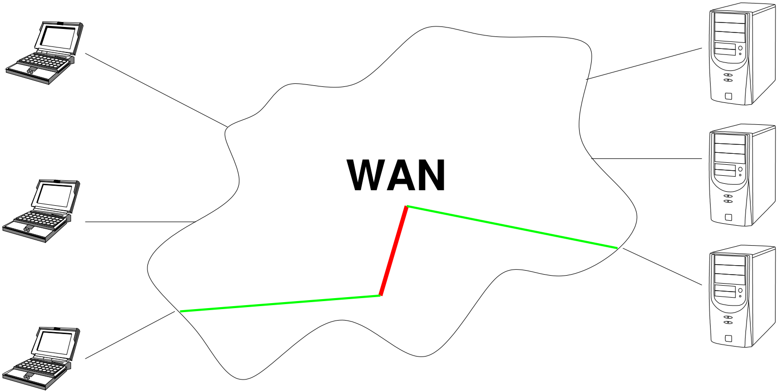
Application: Resonance and Metastable Behavior



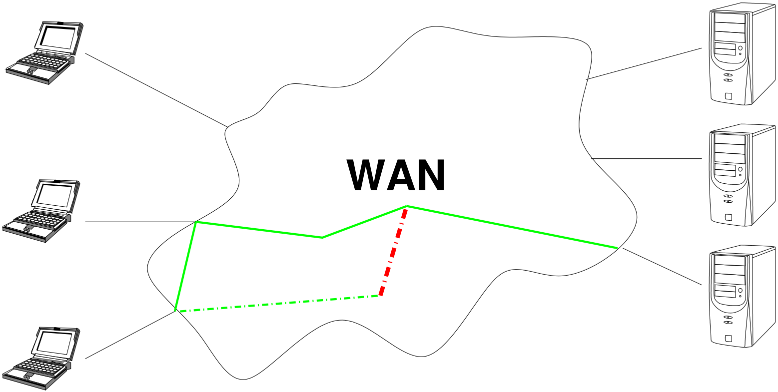
Application: Computer Network Tomography



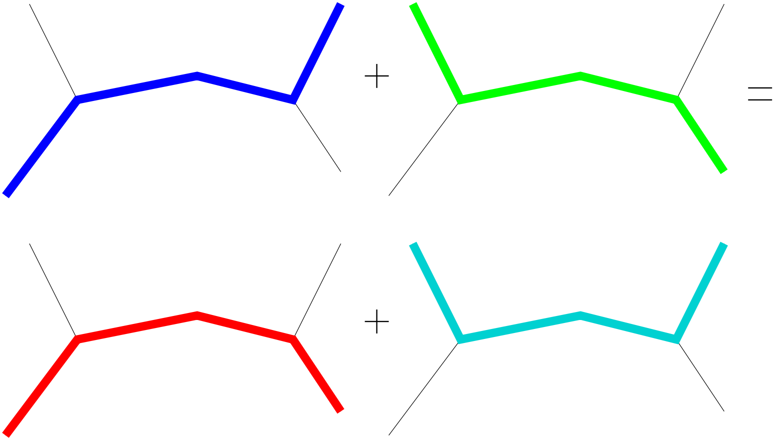
A Possible Problem



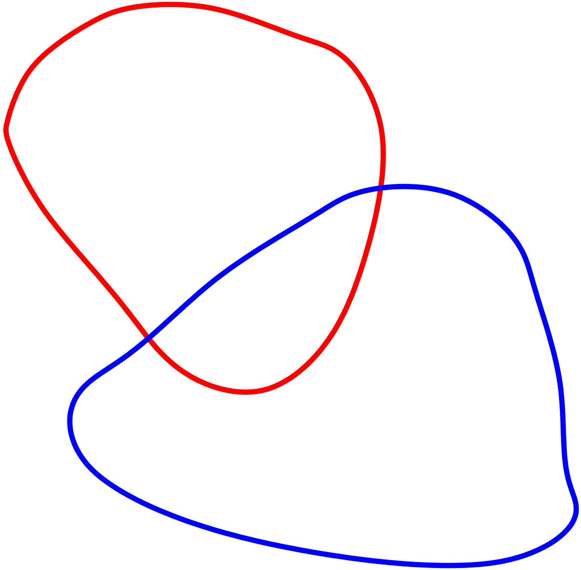
Find and Fix or Route Around?



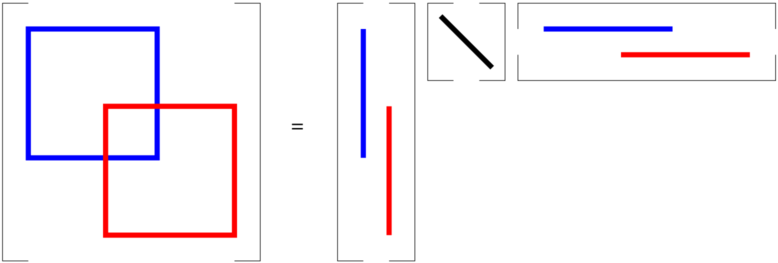
Linear Algebra of Paths



Application: Detecting Overlapping Communities

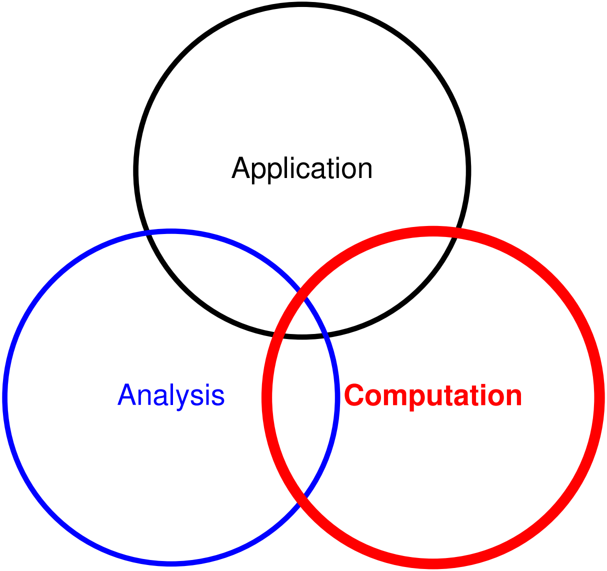


Linear Algebraic View



$$\hat{A} \approx CDC^T$$

What is this class about?



What is this class about?

Our goal: Use numerical methods wisely

- Numerical methods = algorithms for *continuous* problems
- What about applications and analysis?
  - Need applications to ask meaningful questions
  - Need analysis to understand method properties
  - We will focus on numerical methods, but these still matter
- The ideal: *fast*, *accurate*, *robust* methods for cool problems

Topics covered

I plan to cover (roughly):

- Intro and basic concepts
- Numerical linear algebra
- Nonlinear equations and optimization
- Interpolation, differentiation, and integration
- Solving differential equations
- Random numbers and simulation

To follow, you will need:

- Basic multivariable calculus
- Some linear algebra
- Familiarity with MATLAB

Logistics

You will be responsible for information from:

- Lecture: MW 10:10-11 – mix of slides, board
- Section: misc times Th-F – questions, practice problems
- Readings:
  - Heath, *Scientific Computing: An Introductory Survey*
  - Moler, *Numerical Computing with MATLAB* (online is fine)

If you miss a class, get someone's notes!

You can ask questions via the newsgroup or in office hours:

Bindel	TBA
TA	TBA

Workload

- Graded work:

Homework	$8 \times 5\%$ (lowest dropped)
Projects	$4 \times 7\%$
Prelims	$2 \times 8\%$
Final	$1 \times 16\%$

- Late project grading:

$<$ 2 days late	10% off score
$<$ 1 week late	Pass/fail – pass receives 50%
$>$ 1 week	No credit

- Homework may not be late.
- Regrade requests must be within one week of return.

Course policies

- Collaboration
  - Projects can be done in pairs; all else is solo work!
  - *Do* talk to each other about general ideas
  - *Don't* claim work of others as your own
  - *Cite* any online references, hints from the TA, or tips from classmates that you use
- Academic integrity
  - We expect academic integrity from everyone
  - When in doubt on details: talk to us (and *cite*!)
- Emergency procedures
  - If there's a major campus emergency, we'll adapt
  - Information will be posted on the web page and newsgroup

Detailed syllabus

`http://www.cs.cornell.edu/~bindel/class/cs3220-s11/syllabus.html`

Onward!

Let's get a taste of approximation and error analysis.
Let the games begin!

The name of the game

- I have a problem for which the *exact* answer is x.
- I compute, but with finite precision, and get $\hat{x}$.
- I want $\hat{x}$ to have small *relative error*:
  - Absolute error is $(\hat{x} - x)$.
  - Relative error is $(\hat{x} - x)/x$.

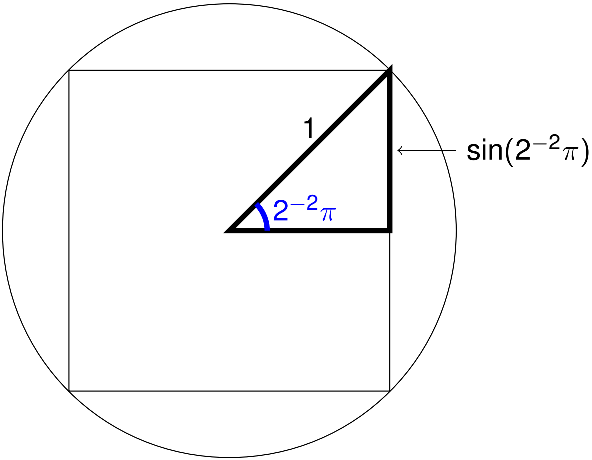
An arithmetic overview

- IEEE floating point arithmetic: like scientific notation, but with 53 binary digits $\approx$ 16 decimal digits).
- *Exact result, correctly rounded* for basic floating point operations (add, subtract, multiply, divide, square root).
- This means we do basic operations to high relative accuracy. For example,

  $$\mathrm{fl}(x + y) = (x + y)(1 + \delta),$$

  where $|\delta| < \epsilon_{\mathrm{machine}} \approx 10^{-16}$.
- What happens when I do a sequence of steps?

Archimedes method



- Inscribe a 2^k-gon in the unit circle.
- Approximate $\pi \approx 2^k \sin(2^{-k}\pi)$.

Archimedes method

Remember the double-angle identity for sines?

$$\sin(2\theta) = 2\sin(\theta)\cos(\theta)$$

Therefore:

$$\sin^2(2\theta) = 2\sin^2(\theta)(1 - \sin^2(\theta))$$

Algorithm to compute $x_k = \sin^2(2^{-k}\pi)$:

- Start from $x_2 = 1/2$
- Compute $x_3, x_4, \ldots$ with the quadratic formula via

$$x_k^2 - x_k + \frac{1}{4}x_{k-1} = 0.$$

- $s_k = 2^k\sqrt{x_k}$ approximates π

Archimedes method

```
% s = lec01pi(kmax)
%
% Compute semiperimeters s(k) of 2^k-gons for k = 2:kmax.

function s = lec01pi(kmax)

  x = zeros(1,kmax);
  x(2) = 0.5;
  for k = 3:kmax
    x(k) = ( 1 - sqrt(1-x(k-1)) )/2;
  end
  s = 2.^(1:kmax) .* sqrt(x);
```

The errors of Archimedes

The code implicitly computes

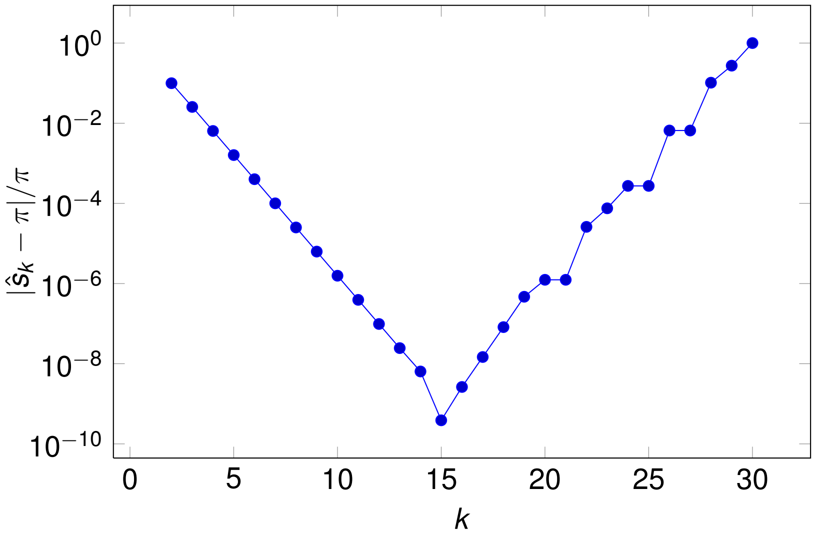
$$s_k = 2^k \sin(2^{-k}\pi)$$

Absent rounding, what is the relative error $|s_k - \pi|/\pi$?

Hint:

$$\sin(x) = x - \frac{1}{6}x^3 + O(x^5)$$

Archimedes, despair!



The problem

Note that

$$x_{29} \approx 2^{-58}\pi^2 \approx 3.4 \times 10^{-17}$$

What happens when we compute $\hat{x}_{30}$?

(We analyze the problem in class – you'll fix it in homework!)

Cancellation

Suppose bold digits are correct:

$$\begin{array}{rl} & \mathbf{1.093752}543 \\ - & \mathbf{1.093741}233 \\ \hline = & \mathbf{0.000011}310 \end{array}$$

Inputs have six correct digits. Output has only one!

Another example

The integrals $E_n := \int_0^1 x^n e^{x-1}\,dx$ can be evaluated by

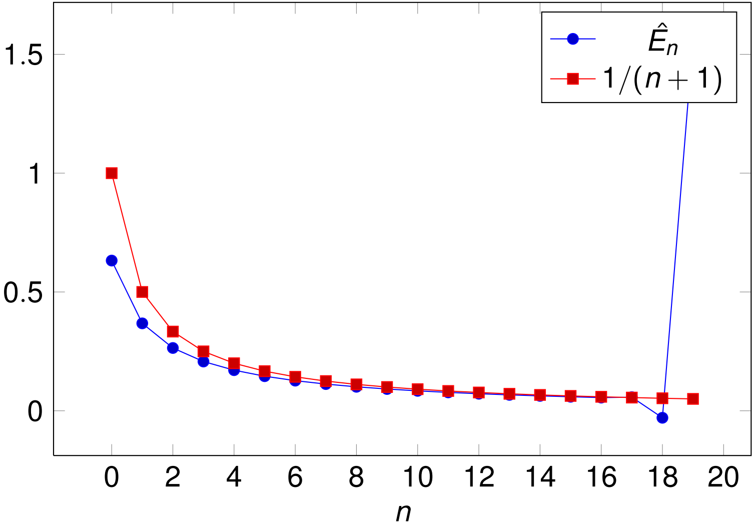
$$\begin{aligned} E_0 &= 1 - 1/e \\ E_n &= 1 - nE_{n-1}. \end{aligned}$$

True values satisfy

$$\frac{1}{e(n+1)} < E_n < \frac{1}{n+1}$$

and $E_n \approx 1/(n+1)$ for n large.

Disaster!



The moral

Scientific computing is not all about error analysis.

... but clearly error analysis matters!