Numerical Methods

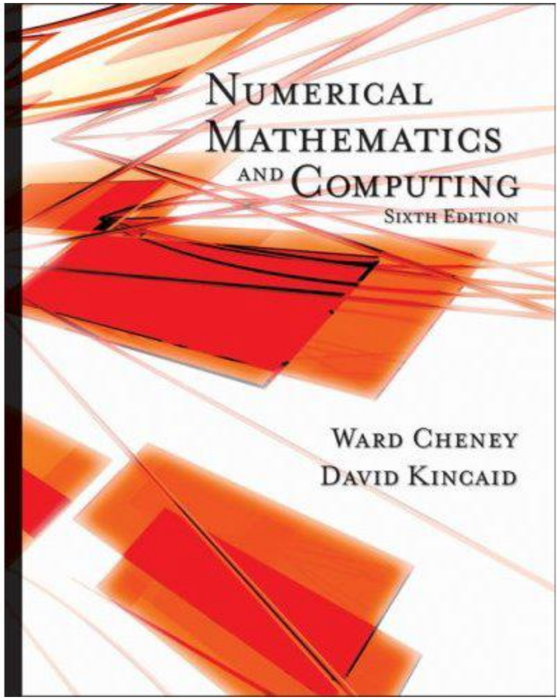
Lecture 1

CS 357 Fall 2013

David Semeraro Ph.D.

Course Information

- Text – *Numerical Mathematics and Computing.* Cheney and Kincaid
- Course Work
  - Weekly homework
  - Two midterm exams
  - Final exam
- Weekly office hours – See Piazza.



Course Information

- Class Website
  – http://courses.engr.illinois.edu/cs357/
  – Contains general information, lecture notes, links to resources
- Class Piazza Page
  – https://piazza.com/illinois/fall2013/cs357/home
  – Discussion forum
    - Ask questions here.
    - TAs, Instructor, and possibly fellow students will answer.
    - Check to see if your question has already been asked.
    - Don't ask questions like "what is the answer to problem 2 on this weeks homework". (Don't answer those sorts of questions either)
    - Don't post answers to the homework on the forum.
  – Announcements
  – Office hour schedules and locations

Assessment

- Grades
  - Homework 25%
  - Midterm 1 25%
  - Midterm 2 25%
  - Final 25%
- Homework
  - Distributed Tuesdays, Due by 5pm following Tuesday in TA dropboxs.
  - 20% penalty for being turning in up to 24 hours late.
  - Submitted in writing. No email.
  - Source code upon request.
  - No hand graphics
  - No dropped sores
  - Collaborate but **do not copy**
    - Cite collaborators at top of paper
    - Hand in your own work in your own words
    - Cheating will not be tolerated.

Programming Environment

- Python
  - Numpy and Scipy packages
  - Information on Python on class webpage.
  - Python workshop
    - Installation
    - Basic usage
    - Intro to Numpy , Scipy, and Matplotlib

Why Study Numerical Methods

Numerical problems are pervasive in many areas of computation.

Science and Engineering

- Climate modeling
- Molecular dynamics
- Structural modeling
- Robotics – computer vision

Computational Finance and Economics

- Commodity pricing
- Economic modeling

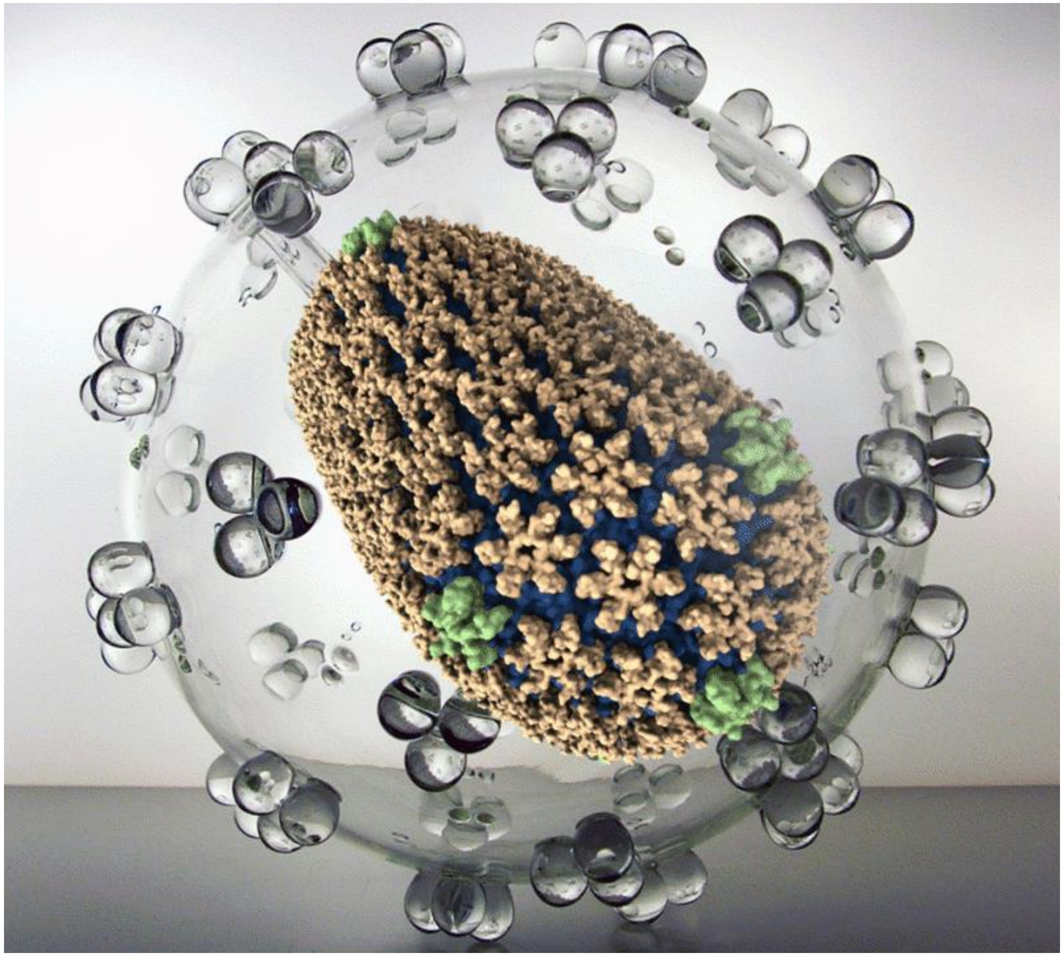
Entertainment

- Game physics
- Digital effects

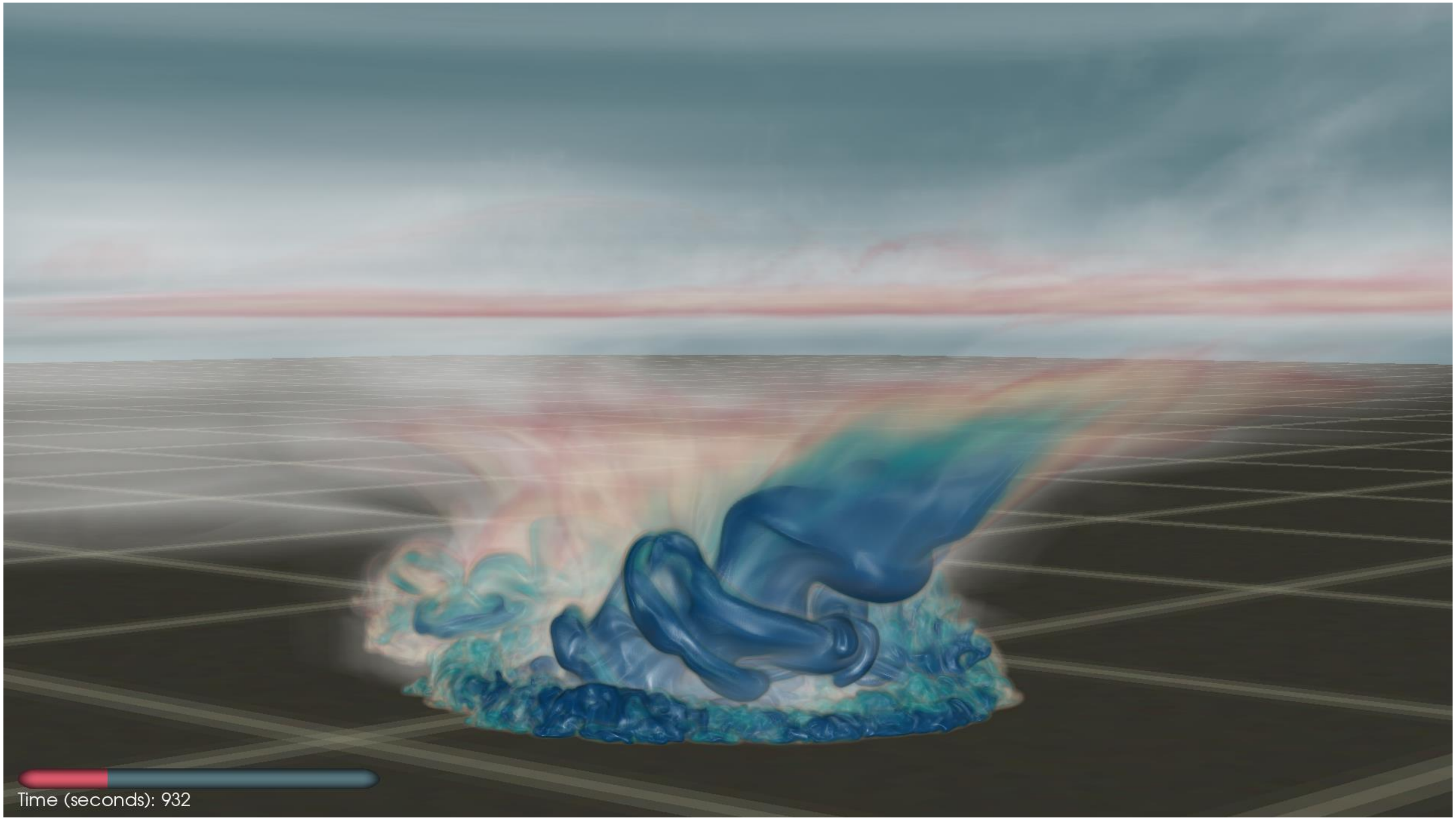
Informatics

- Google page rank
- Amazon recommendations

HIV Capsid



Thunderstorm



Motion Picture Effects

Motion Picture Effects

“To create realistic simulations of fluids, the software has to mimic the underlying physical processes as accurately as possible, by implementing the mathematical equations that describe these processes. But these equations are incredibly complex — in fact there is no complete mathematical description of fluid flow. So the challenge facing programmers is not only to understand the math, but also to figure out how to turn it into programs that don't take ages to run. It's a struggle between accuracy and efficiency.” -Alexis Wajsbrot

What are Numerical Methods?

Numerical methods are techniques for solving mathematical problems on digital computers.

- Key Concepts
- Approximation: An approximate solution is sought. How close is this to the desired solution?
- Efficiency: How fast and cheap (memory) can we compute a solution?
- Stability: Is the solution sensitive to small variations in the problem setup?
- Error: What is the role of finite precision of our computers?

Formulation and Solution

Physical process

- Observation
- Experiment

Mathematical description

- Derived from observed physical relations
- Complex analytical expression

Numerical solution

- Discrete mathematical description
- Numerical approximation

Methods, Algorithms, and Implementations

- Method: a general (mathematical) framework describing the solution process
- Algorithm: a detailed description of executing the method
- Implementation: a particular instantiation of the algorithm
- Is it a “good” method?
- Is it a robust algorithm?
- Is it a fast implementation?

Analytic vs. Numerical

- Analytic Solution (a.k.a. symbolic): The exact numerical or symbolic representation of the solution may use special characters such as , e, or tan (83)
- Numerical Solution: The computational representation of the solution entirely numerical

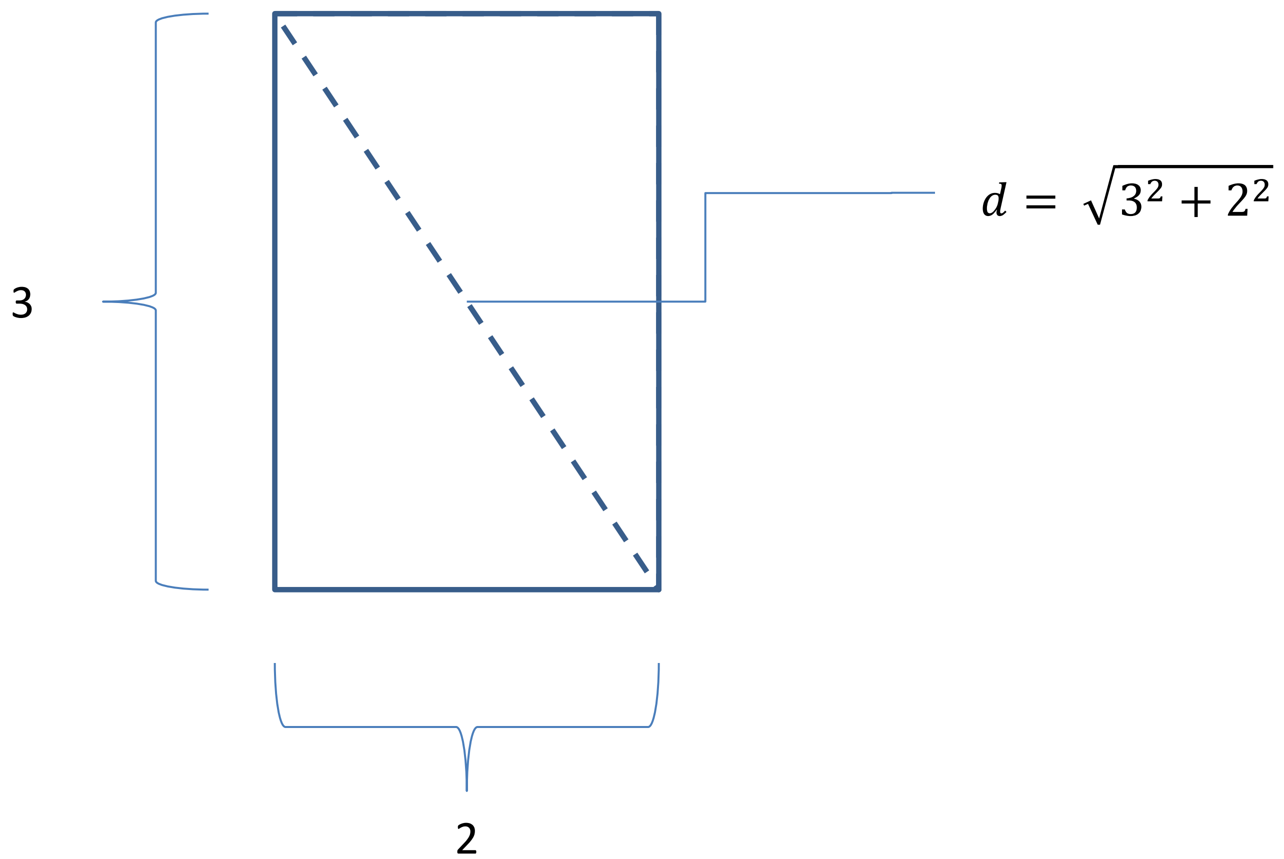
Analytic	**Numerical**
1/2	.5
1/3	.333…
π	3.14159…

Approximation

Computers can not represent numerical values to infinite precision. Therefore some approximation must be made which introduces error.

Analytic	Numerical	Approximate
1/2	.5	.5
1/3	.333…	.333
π	3.14159…	3.14159

Example 1 (significant digits)



Example 1

- $d = \sqrt{3^2 + 2^2} = \sqrt{13} = 3.605551275$

Analytical result.

Numerical approximation. *

*** Significant Digits** – digits beginning with the leftmost *nonzero* digit and ending with the rightmost *correct* digit, including final zeros that are exact. (10 in this example)

Example 1

- What if the sides of the rectangle were accurate to only ± .001 ?
- $\sqrt{2.001^2 + 3.001^2} \geq d \geq \sqrt{1.999^2 + 2.999^2}$
- $3.6069 \ \geq d \ \geq 3.6042$
- These results are rounded to 5 significant digits.

Error

- Absolute error

$$|exact\ value - approximate\ value|$$

- Relative error

$$\frac{|exact\ value - approximate\ value|}{|exact\ value|}$$

Error Example

Exact	Rounded	Significant Digits	Absolute Error	Relative Error
0.00347	0.0035	2	0.3×10^{-4}	0.865×10^{-2}
30.158	30.16	4	0.2×10^{-2}	0.66×10^{-4}

Accuracy

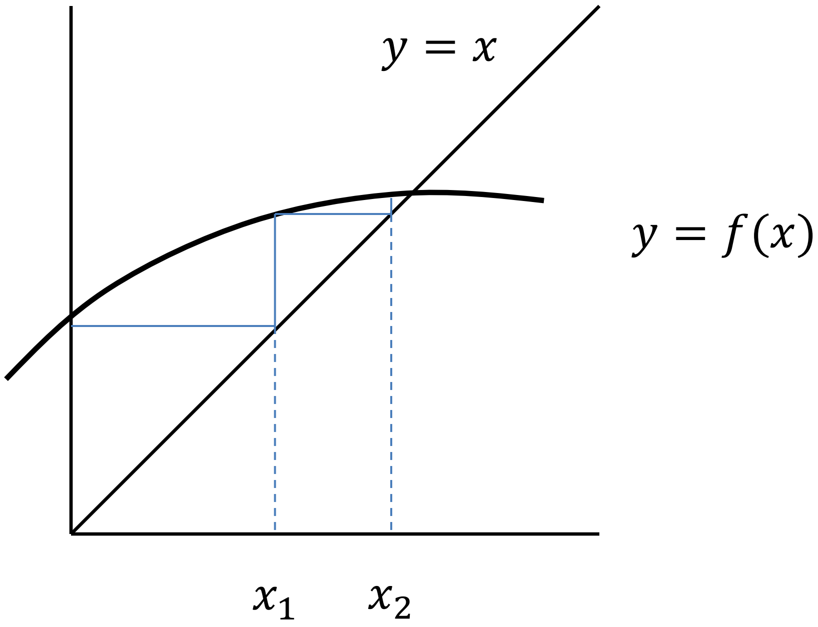
Accurate to *n* decimal places means you can trust *n* digits to the right of the decimal place.

Accurate to *n* significant digits means you can trust a total of *n* digits as being meaningful beginning with the leftmost nonzero digit.

Iteration or Successive Approximation

- Solve an equation of the form
$$x = f(x)$$
- Compute the sequence
$$x_1 = f(x_0), x_2 = f(x_1), \ldots$$
- If $\{x_n\}$ converges to α we have
$\lim_{n} f(x_n) = f(\alpha)$ so $x = \alpha$ satisfies the equation $x = f(x)$.

Geometric Interpretation



Successive Approximation

- Find the positive root of the quadratic $f(x) = 5x^2 + 4x - 1$.
- $\alpha_1 = -1, \alpha_2 = \frac{1}{5}$
- Equivalent to finding positive root of $x = g(x) = \frac{1-5x^2}{4}$
- Construct the sequence:

$$x_1 = g(x_0), x_2 = g(x_1), \ldots$$

Successive Approximation

k	x_k
0	.000
1	.250
2	.172
3	.213
4	.193
5	.203
6	.198

Successive Approximation

The sequence $\{x_k\}$ converges to the positive root $\alpha_2 = 1/5$.

$$\lim_{k\to\infty} g(x_k) = g(\alpha_2)$$

Why did it work?

Successive Approximation

Mean value theorem:

$$g'(\varepsilon_k) = \frac{g(x_k) - g(x_{k-1})}{x_k - x_{k-1}}$$

$$= \frac{x_{k+1} - x_k}{x_k - x_{k-1}}$$

$$x_{k-1} < \varepsilon_k < x_k$$

Successive Approximation

If:

$$|g'(\varepsilon_k)| < 1$$

Then:

$$|x_{k+1} - x_k| < |x_k - x_{k-1}|$$

Convergence is assured provided:

$|g'(\varepsilon_k)| < 1$ for all x in the interval containing the initial iterate $x_0$and the root $\alpha_{2.}$

Taylor Series

- All the computer does is add and multiply
- How do we evaluate $e^x, \cos x, \sqrt{x}$?
- Use the Taylor series:

$$f(x) = \sum_{k=0}^{\infty} \frac{(x-c)^k}{k!} f^k(c)$$

Taylor Series

- Example e^x

  expand about c $= 0$ $(e^0 = 1)$

$$f(x) = e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$

- Can’t evaluate infinite series so what do we do?

Taylor Series

- Truncate
- Taylor series polynomial of degree n

$$p_n(x) = \sum_{k=0}^{n} \frac{(x-c)^k}{k!} f^{(k)}(c)$$

$$e^x \approx p_n(x) = 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^n}{n!}$$

Taylor Series

- Example

```
from math import factorial;
from math import exp;
x=2.0;
pn = 0.0;
for i in range(10):
    pn = pn + x**i / factorial(i);
    err = exp(x) - pn;
    relerror = err/exp(x);
    print i, err, relerror;
```

```
0 6.38905609893 0.864664716763
1 4.38905609893 0.59399415029
2 2.38905609893 0.323323583817
3 1.0557227656 0.142876539501
4 0.389056098931 0.0526530173437
5 0.122389432264 0.0165636084806
6 0.0335005433751 0.00453380552625
7 0.00810371797827 0.00109671896786
8 0.00175451162906 0.000237447328261
9 0.000343576884796 4.64980750174e-05
```

Taylor Series

- Taylor's Theorem for $f(x)$

If the function f possesses continuous derivatives of orders 0 through $n+1$in $[a,b]$, then for any c and x in $[a,b]$,

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(c)}{k!}(x-c)^k + E_{n+1}$$

$$E_{n+1} = \frac{f^{(n+1)}(\varepsilon)}{(n+1)!}(x-c)^{n+1}$$

Truncation Error

- Can't evaluate infinite series so we truncate after term n.
- The *truncation error* is then:

$$E_{n+1} = \frac{f^{(n+1)}(\varepsilon)}{(n+1)!}(x-c)^{n+1}$$

Taylor Series in terms of h

- Taylor's Theorem for $f(x+h)$

If the function f possesses continuous derivatives of orders 0 through $n+1$ in $[a, b]$, then for any x in $[a, b]$,

$$f(x+h) = \sum_{k=0}^{n} \frac{f^{(k)}(x)}{k!} h^k + E_{n+1}$$

$$E_{n+1} = \frac{f^{(n+1)}(\varepsilon)}{(n+1)!} h^{n+1}$$

Big O

- The error term depends on h explicitly and because ε depends on h.
- $E_{n+1} \rightarrow 0$ as $h^{n+1} \rightarrow 0$.
- For large n this convergence is rapid.

$$E_{n+1} = \vartheta(h^{n+1})$$

Next time

- Section 1.1 and 1.2 in the book correspond to this weeks lectures.
- More on Taylor series, errors, and floating point.
- Some programming examples. (Python)