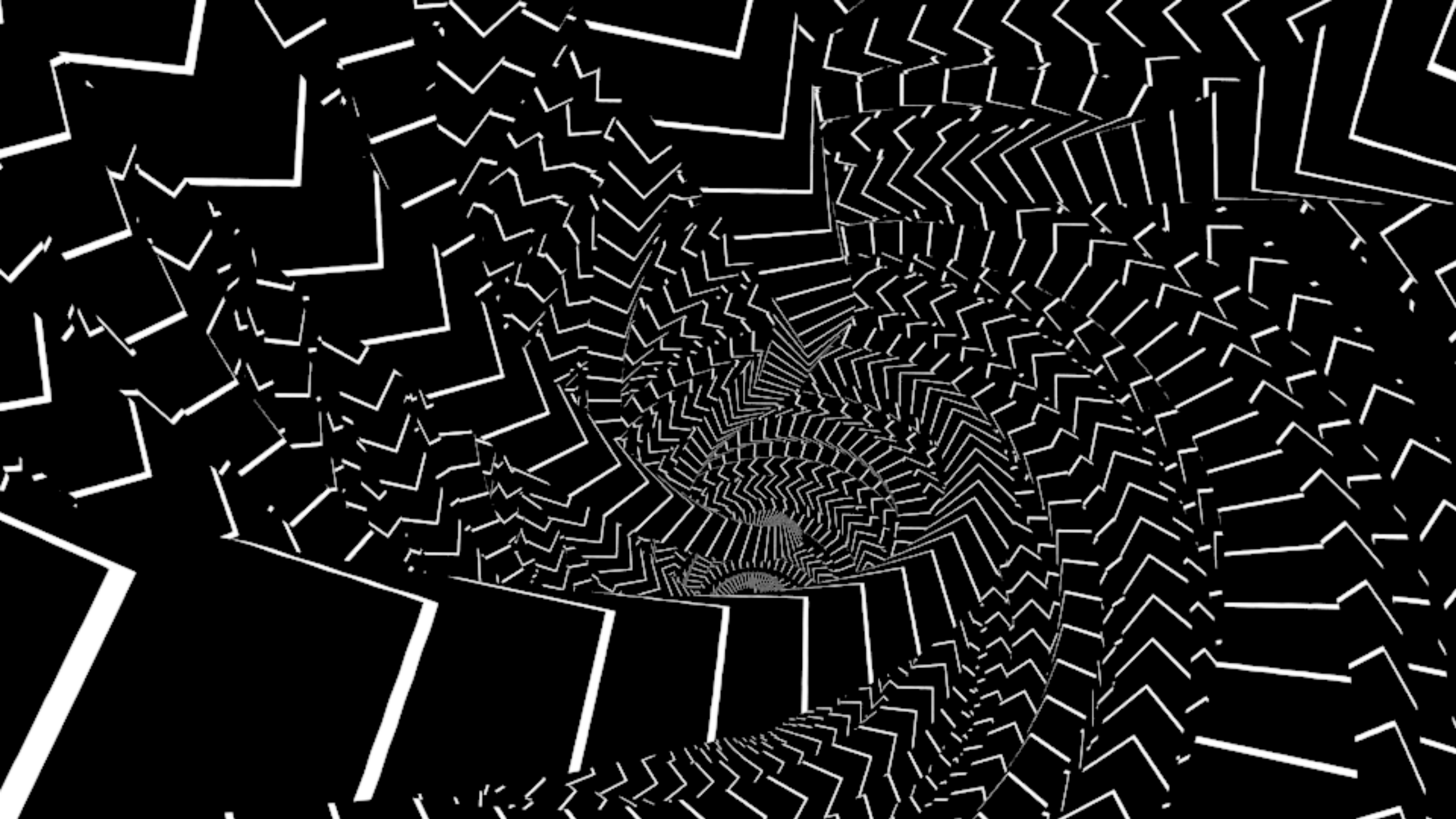


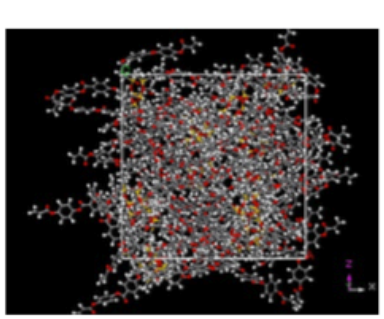
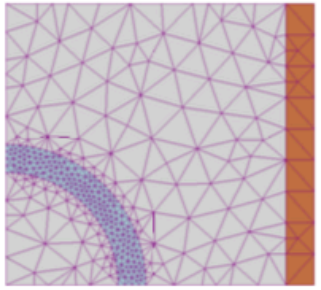
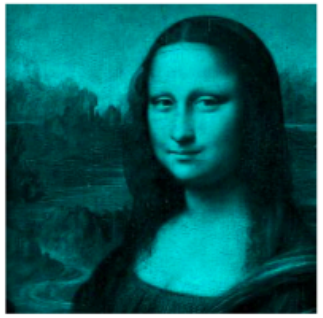
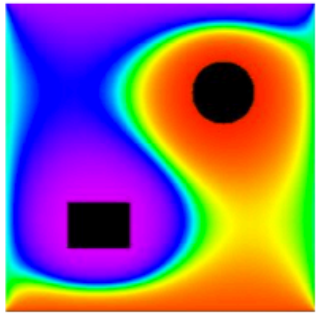
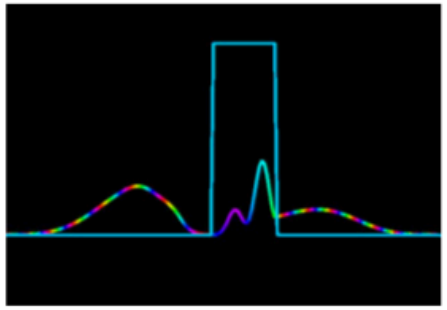
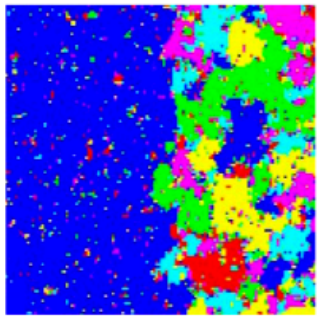
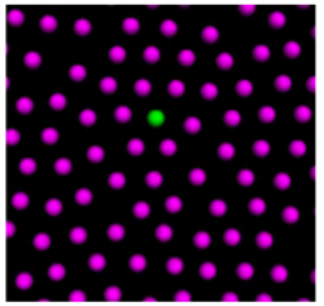
EC526 Parallel Algorithms for High Performance Computing

Spring 2024: Tues, Thurs 1:30pm-3:15pm CDS 163

Graduate Hands on Special Topics Course

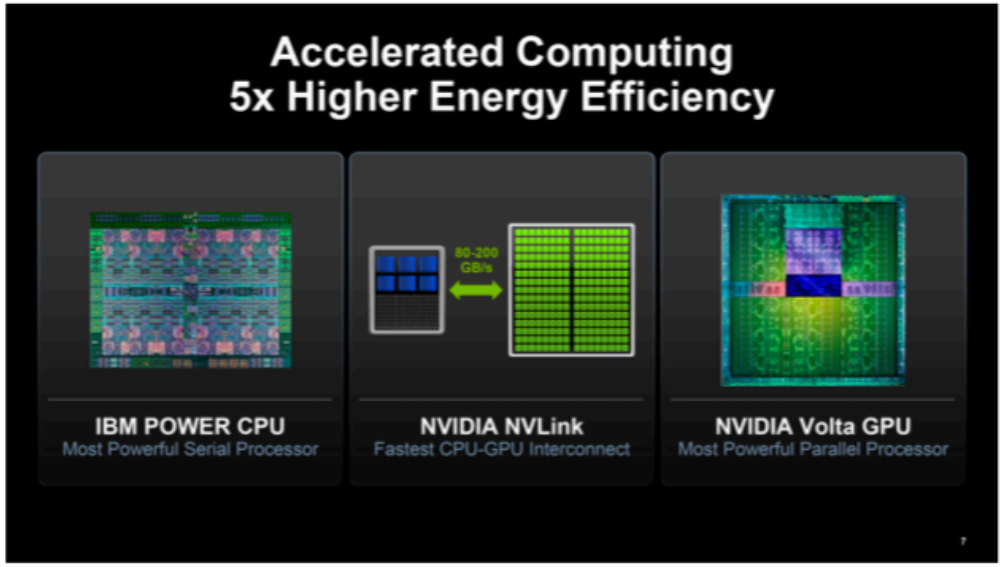
Course Description: The explosive advances in High Performance Computing, Big Data and Machine Learning require software design and parallel algorithms for distributed computing and data sets. Examples will be drawn from FFTs, Dense and Sparse Linear Algebra applied to simple physical systems using Multigrid Solvers, Monte Carlo Sampling and Finite Elements with a final team project to explore one example in more detail. Coding exercises will be in C++ in the UNIX environment with parallelization using MPI message passing, OpenMP threads and OpenACC threds on GPUs. Rapid prototypes and graphics methods will be provided with Mathematica.(**Instructor: Prof. Richard Brower**)

What and Why this Course

Seven Dwarf: The 7 HPC dwarfs are illustrative of applications and algorithmic motifs for exploration in term project.

Exercises will be presented in zoom lecture and run at scale on class accounts on the MGHPCC Shared Computer Cluster. Prerequisite is programming experience in C at the level of EC327 or EC602 or consent of the instructor.



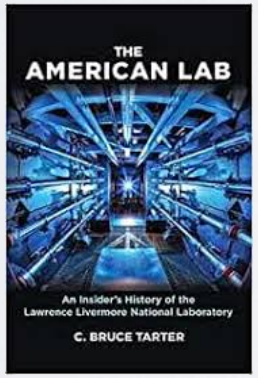
EC526
FORMER BU POSTDOCS

AND EC526 ASSISTANCE

LEFT TO RIGHT:

EVAN, SAUL, KATE
(NVIDIA)

(LLL CALIF)



Grades — Do the work get good grades.

Software Exercises: 40 % of grade: Every week to two weeks there will be a written Homework and a Laboratory software exercise. The grades will be based on the written answers and output of these programs in terms of performance (speed and accuracy) results presented in tables and graphs. Some exercises will include "pencil and paper" work on the algorithmic concepts. Code must be not be "copy and pasted"' from on line examples and no sharing of code between students.

Mid Term Exam: 20 % of grade: There will be no final exam.

Team Project: 30% of grade: After the midterm term we will start the projects but hopefully teams and topics will be formed earlier. The presentation of the project at the end of the semester is in leu of a final exam.

In class Participation: 10% of grade: The goal of in class work is to help with developing good coding methods in term of style organization, clarity and comments. Sharing ideas (not code) is encourage in class and on Slack. Readable code is useful especially in team software projects who of course do share code.

* There may be occasional random in class puzzles to keep everyone entertained and thinking .

Syllabus — Selection in response to Student interest/projects

1. First Part Weeks 1-4 : **Algebra**

    - Intro to HPC and Engineering impact
    - Language type: Symbolic, Interpreted and Compiled
    - The ubiquitous Linear Algebra: HPC/ML/BigD/QC
    - Numerical algebra for calculus and Floating Point Errors.
    - Differentiation & Integration (Gauss and Monte Carlo)
    - Newtons method for root finding & Non-linear optimization
    - Data Analysis, Curve fitting &Error analysis
    - FFT and related examples of recursion for divide and conquer
    - Simple ODEs (oscillation vs relaxation)

2. Second Part: Weeks 5-9: **Parallelization Tools**

- PDE in Electrostatics and Image Processing,
- Select Projects & Parallelization Method
- OpenMP and OpenACC (for CUDA) threads.
- Message passing with MPI and Network Performance
- Scripts for submitting to HPC systems.
- Use of Numerical Libraries

3. Third Part Weeks 10-14 (5 weeks): **Special Topics/Projects**

- Dynamical PDEs for Heat Flow vs Wave propagation.
- Conjugate Gradient and iterative solvers
- Multi-gird Linear Solvers
- MD short range (neighborhood tables) vs long range Coulomb
- MonteCarlo for Magenets: Cluster and Graphs
- Finite Element Method in 2D
- Digital Quantum Computing Programs

Tools & Links for Computing

GitHub: https://github.com/brower/EC526_2024

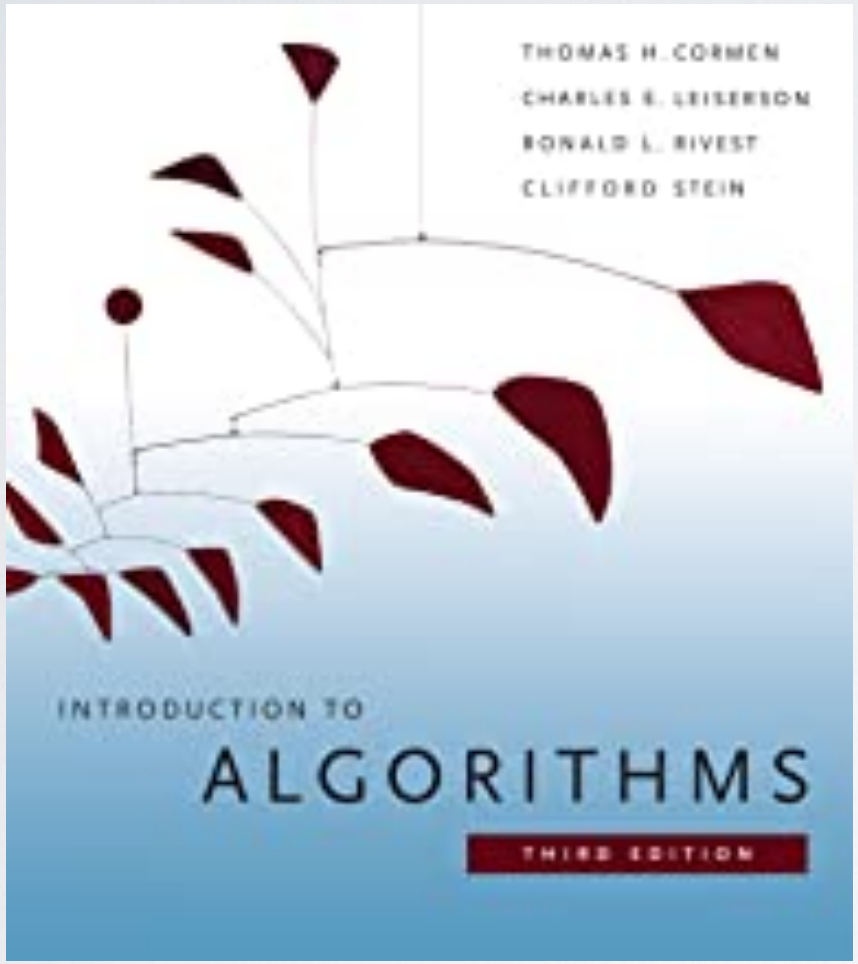
MGHPCC: https://www.mghpcc.org

SCC-ondemand : https://scc-ondemand2.bu.edu/pun/sys/dashboard

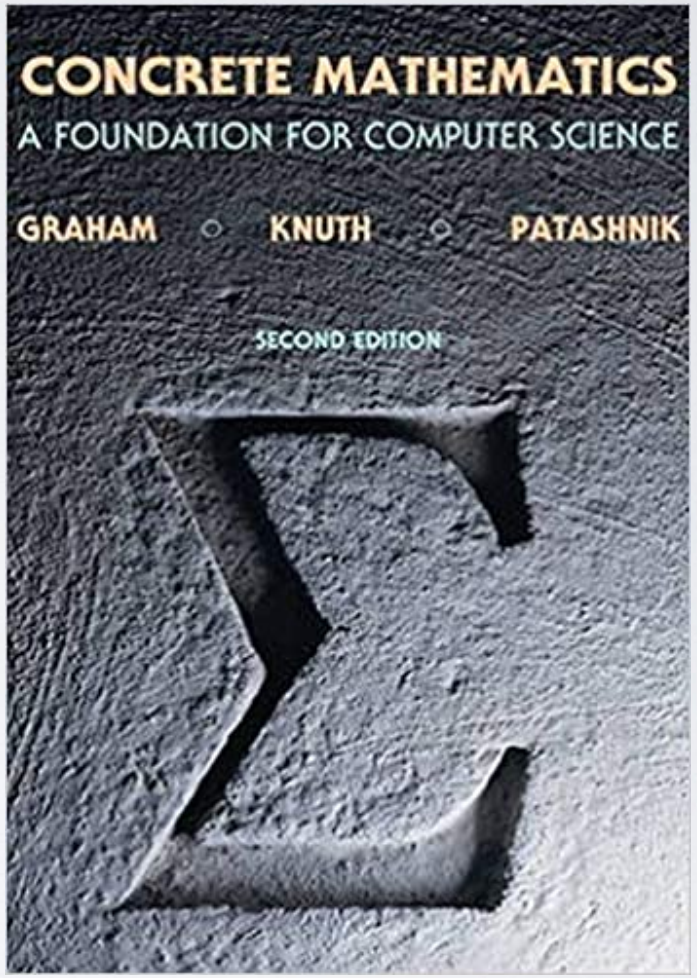
Mathematica: http://www.bu.edu/tech/services/cccs/desktop/distribution/mathsci/mathematica/student/

Comment: Our “Text” is Course Lectures Notes

Classic CS Algorithm Ignore Real Computation but idea are often the same



No FP—No Convergence Tests — No Error Analysis



Nice Algebra of Discetization
No actual applications


FRONTIER
OAK RIDGE National Laboratory
U.S. DEPARTMENT OF ENERGY
Hewlett Packard Enterprise
AMD
FRONTIER
OAK RIDGE National Laboratory
U.S. DEPARTMENT OF ENERGY
Hewlett Packard Enterprise
AMD
FRONTIER

DoE HPC Roadmap: Exascale computing project (2021-2025)

Frontier AMD CPU, AMD GPU; HIP

ORNL

Perlmutter AMD CPU, Nvidia GPU; CUDA

NERSC

Aurora Intel CPU, Intel GPU; SYCL

Argonne

HPC computing in the US *will* be accelerated

“Native” programming models are all distinct

- **Possible strategies**
  1. **Abstract the differences (write an interface that can be implemented on them all)**
  2. **Use someone else’s abstraction (e.g. Kokkos)**
  3. **Rely on a standard like OpenMP 5.0 *target offload***

Warning:

Need lots of care with Numeral errors:

Approximation to Calculus and Floating Point round-off

We will look at these one by one. (Really fun too)

(This is real computing generally neglected by CS texts and most applied math folks. Crucial to all Engineering and Science) https://en.wikipedia.org/wiki/Floating-point_arithmetic

Pi = 11.00100100001111110110101010001000100001011010001100001000110100 11

In binary single-precision floating-point, this is represented as $s = 1.10010010000111111011011$ with $e = 1$.

This has a decimal value of **3.1415927**410125732421875

Mathematica PI = 3.14159265358979323846264338327950288 4

Topic #1 Derivative & Integrals

Comment: Almost ALL Computing is Linear Algebra Why?

Calculus has continuous "operators" —> Linear Algebra on Computers

In Computer Science : Geometry (points/distances) —> Lists/Graphs

Calculus	—	Computing

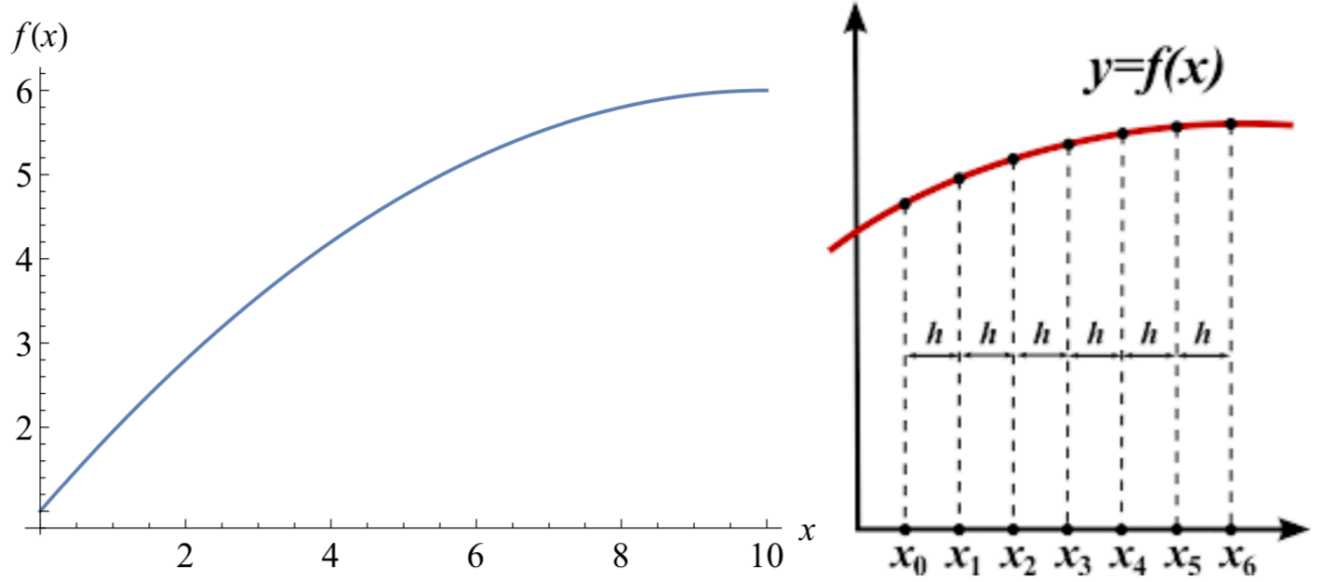
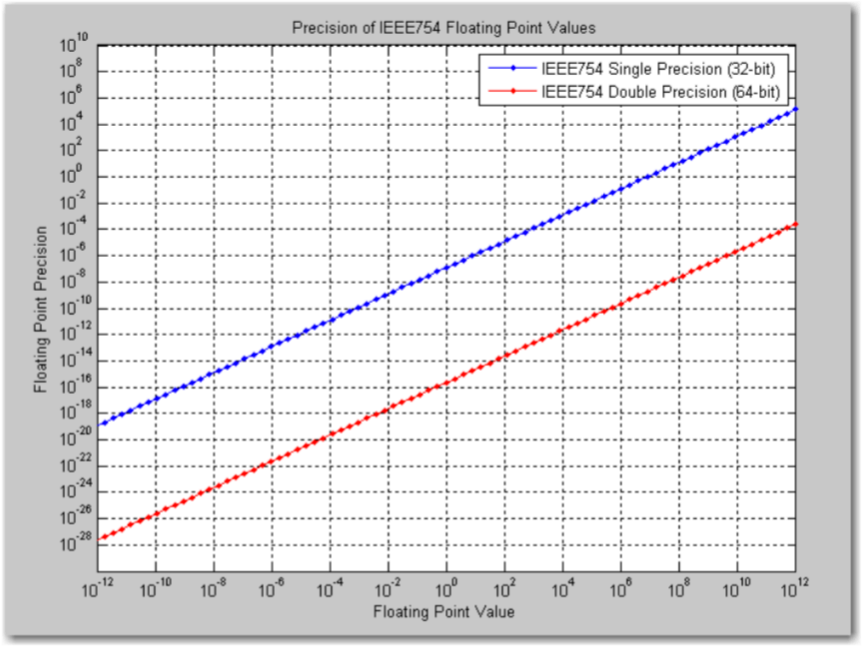


Figure 1: On left a function of x. On right the discrete points evaluated on a grid

Everything in Computer has FINITE number of bits!

https://en.wikipedia.org/wiki/Single-precision_floating-point_format

https://en.wikipedia.org/wiki/Double-precision_floating-point_format



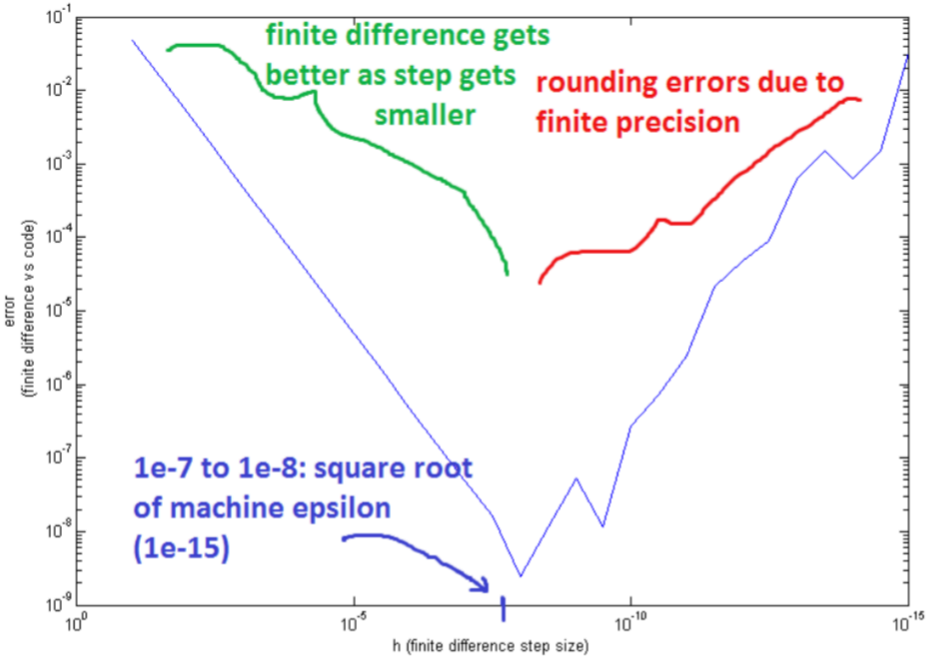


Figure 3: On the Right the error in the finite difference approximation to a derivative (y-axis) relative to size set size h (x-axis).

Mathematicians Derivatives

Math: $\frac{d}{dx} f(x) = g(x)$ or $D[f(x)] \rightarrow g(x)$

Property: "Linear Algebra"

Give me an other example?

$$D[c_1 f_1(x) + c_2 f_2(x)] \equiv c_1 D[f_1(x)] + c_2 D[f_2(x)]$$

$$D[x^k] = k x^{k-1} \quad \text{for} \quad k = 0, 1, \cdots n$$

Given: $\frac{d}{dx} x^n = n x^{n-1}$

and $f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \cdots$

we know $g(x) = D[f(x)] = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + \cdots$

$$b_0 = a_1 \ , \ b_1 = 2a_2 \ , \ b_2 = 3a_3 \ , \ \cdots \ , \ b_k = (k+1)a_{k+1} \ , \ \cdots$$

MATRIX FORM DERIVATIVES OF POWER

$$g(x) = D[f(x) \implies b_i = \sum_{j=0}^{n} D_{ij} a_j$$

$$\begin{bmatrix} b_{-1} \\ b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix}$$

Note Det = 0 NO INVERSE WHY

Infinite matrix to get all function with power taylor series.

Slick Trick
Eigenfunctions of logarithmic derivative

$x \frac{d}{dx} x^k = xD[x^k] = kx^k$

Mathematicians/Computer Limit Derivatives

$$\frac{d}{dx}f(x) \equiv \lim_{h\to 0}\frac{f(x+h)-f(x)}{h}$$

Computer use: $\frac{d}{dx}f(x) \simeq \Delta_h[f(x)] \equiv \frac{f(x+h)-f(x)}{h}$ for $h = 10^{-6}$ Right difference*

Note Still "Linear Algebra" – Actually Matrix * vector (CALL MAT-VEC)

$$\Delta_h[c_1 f_1(x) + c_2 f_2(x)] = c_1\Delta_h[f_1(x)] + c_2\Delta_h[f_2(x)]$$

PROBLEM: How small should "h" be? Limit is 0/0 Computer can do that!

---

* Why not use? $\frac{d}{dx}f(x) \simeq \widetilde{\Delta}_h f(x) = \frac{f(x)-f(x-h)}{h} = -\Delta_{-h}$ Left difference?

What about Special Calculus D tricks ?

Product Rule: $(fg)' = f'g + fg'$, $D[f(x)g(x)] = D[f(x)]g(x) + D[f(x)]g(x)$

Ratio Rule: $(f/g)' = f'/g - fg'/g^2$, $D[f(x)g(x)] = D[f(x)]g(x) - f(x)D[g(x)]/g^2(x)$

Chain Rule: $\frac{d}{dx} f(g(x)) = f'(g)g'(x)$, $D[f(g(x)), x] = D[f(y), y = g(x)]D[g(x), x]$

Second Derivativez: $\frac{d^2}{dx^2} f(x)$, $D^2[f(x)] = D[D[f(x)]]$

Do they hold when substitute derivatives by difference? $D[f] \to \Delta_h f(x)$

What is the inverse?

$D[f(x)] = g(x) \implies g(x) = D^{-1}[f(x)]$? Is there an Anti-derivative?

Will discuss one by one very carefully.