

Welcome to Computational Science I Scientific Computing II

informal introductions

Rubin H Landau

**Head, Computational Physics for Undergraduate
BS Degree Program at Oregon State University**



with

Sally Haerer

Director/Producer/Editor

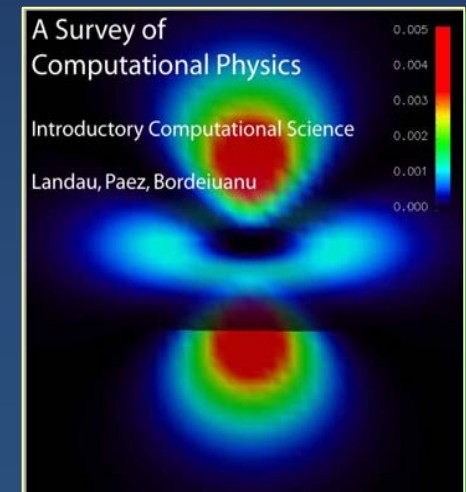
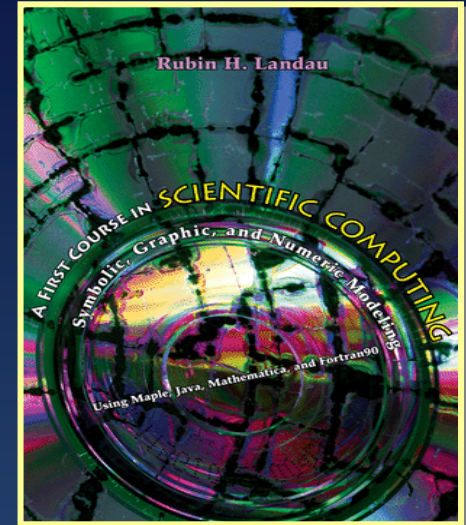


Computational Science I

Scientific Computing II

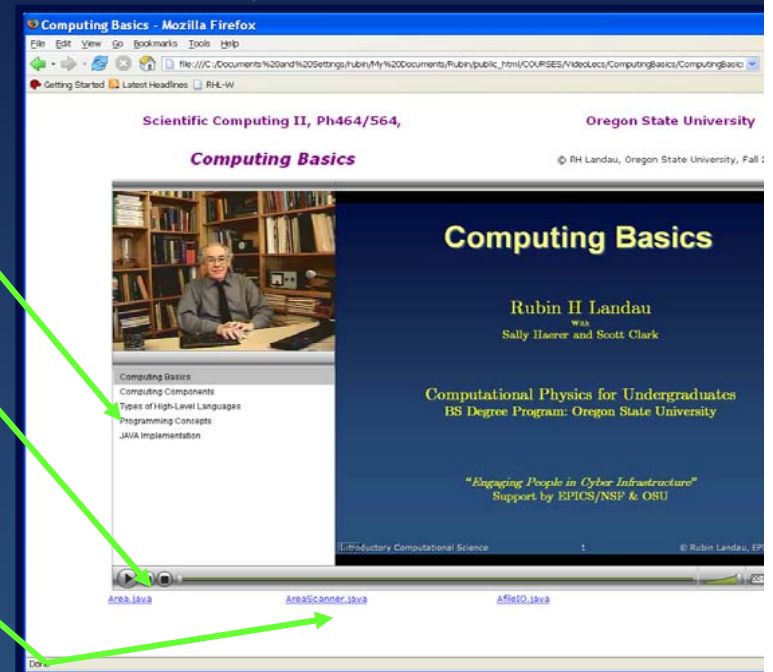
(Dept S C)

- How use computers to do science, engineering
- II: Assume know way around keyboard
- II: Assume compiled language (Java)
- I: *A First Course in Scientific Computing*
Princeton University Press, 2006
- I \Rightarrow this course \Rightarrow Computational X (\uparrow X)
- Lectures \leftarrow text: Landau, Paez, Bordeianu
A Survey of Computational Physics
Princeton University Press, 2008




How to Use these Lectures

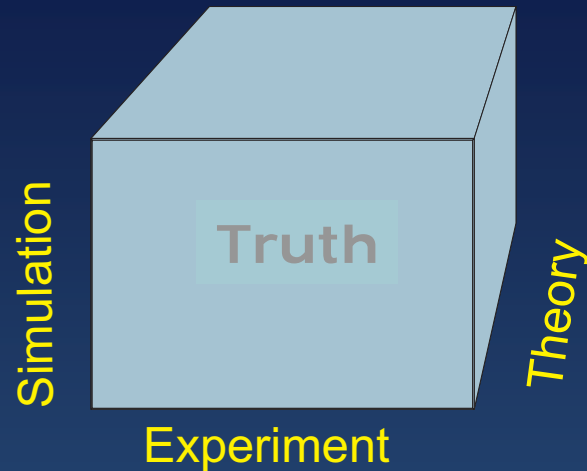
- As always, before/after reading text (> lectures)
- Web based, universal, flash-enabled browser
- View, listen, take anywhere; podcast
- Table Contents below
- Stop, replay parts; jump around
- Tired of my voice: headphones
- Adjust volume: computer controls
- Links: codes, animations, color viz,...
- Real practitioner \neq news reader (\pm)
- Carefully planned, *not* scripted \leq boring
- Accessible slide (LaTeX) rubin@science.oregonstate.edu
- Distance Ed technology \Rightarrow closer to profs, increase access



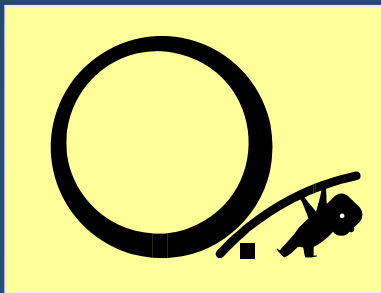
Philosophy, Local Setup

- Attitude: Learn computation by/while computing
- + Motivate: science, math, creativity: do practical applications
- \neq *Theory* of scientific computation
- \neq UG black box; look in box, get hands dirty 
- Compiled language: closer to algorithms, math
- Sample codes \Rightarrow t extend, explore (realistic)
- Loaded computer (lab) and projects \Rightarrow material = yours
 - Java Developer's Kit + shell, others OK (free)
 - Java: excellent for UG; Research: C(++), Fortran95 (codes)
 - Gnuplot, AceGr, PtPlot, OpenDX*, JAMA (matrix libe)
 - jEdit, shell, Emacs, (studio)
- When, where, meet, which computers?
- How, when assignments, exams?
- e.g.: link to my class pages

Nature of Computational Science

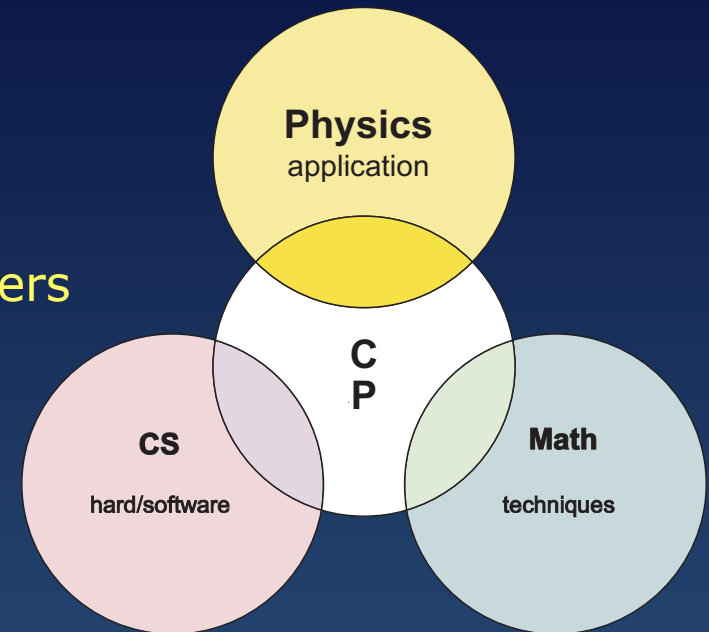


- *Paradigm shift* doing science
- *Teach*: Explore models, solve problems
- Understand depths > otherwise possible
- Difficult, complex problems
 - \Rightarrow beyond analytic solution
 - \Rightarrow human endurance
- Computer = Σ
 - super-calculating machine
 - lab: numerical simulation world
 - lever for our intellectual abilities



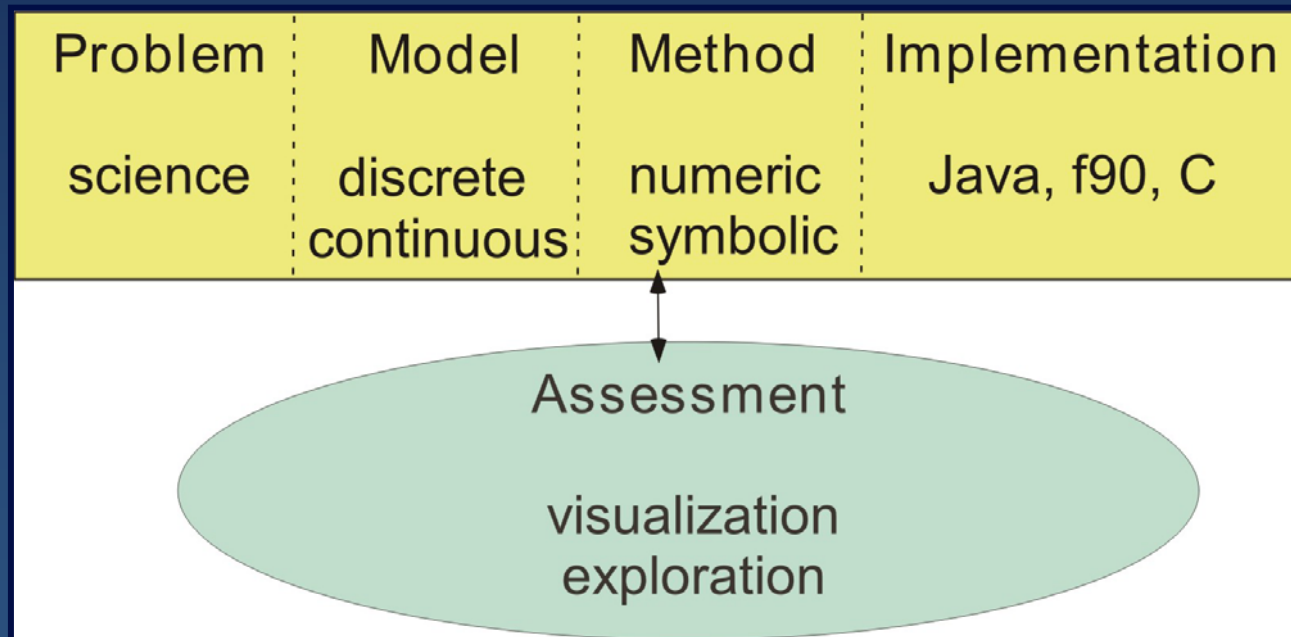
What is Computational Science?

- CX (CSE): multi-discipline, bridge
 - CSE focus: science problems + computers
 - common techniques, philosophy
 - vs subspecialization
- Computer Science (CS) focus:
 - computing for its own intrinsic interest
 - develop hardware and software tools
- Not just semantics:
- CSE: "tomorrow's problems yesterday's computers"
- CS: ... other way around"



Problem-Solving Paradigm

(Viewpoint: Course, Text)



Course Topics (Tools, 1/3 Text)

- 1. Intro & Computational Basics*
- 2. Errors & Uncertainties in Computations*
- 3. Trial & Error Searching*
- 4. Visualization, Object-Oriented Programming*
- 5. Monte Carlo Simulations*
- 6. Matrix Computing, Scientific Libraries*
- 7. Computer Hardware & Tuning**
- 8. Integration & Differentiation*
- 9. Data Fitting*
- 10. Differential Equations*

Computational Physics Education by R Landau



Good Bye for Now

Time to move on and get to work!