Com S // Cpr E // Math 5250

Numerical Analysis of High-Performance Computing

Instructor: Songting Luo

Lecture 1: Overview and Introduction

Outline

1. Overview
2. Why is 5250 a math class?
3. What is high-performance computing?
4. Objectives of this class
5. An example to get the semester started

Overview

Overview

- Today's lecture
  - Get the class information: Discussion of syllabus via Canvas
  - Understand whether this is a class worth taking

Instructor: Songting Luo

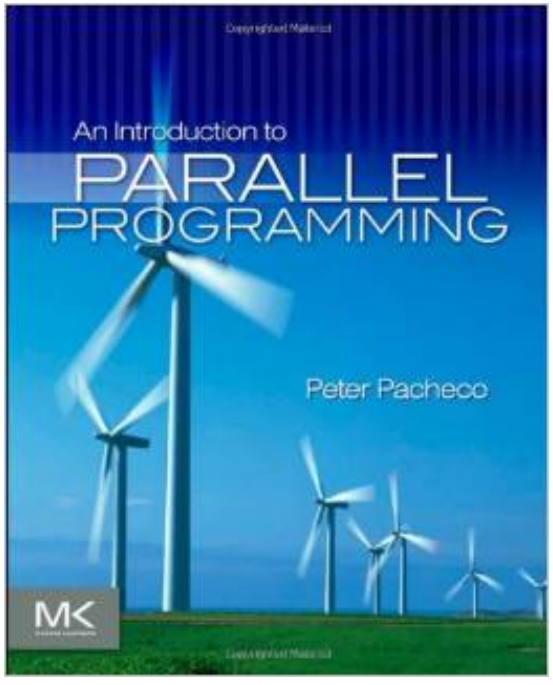
- Contact Information
  - Office: 430 Carver Hall
  - Phone: 515-294-8140
  - Email: luos@iastate.edu
- Research interests:
  - Broadly speaking: computational and applied mathematics, partial differential equations, mathematical modeling, numerical analysis, scientific computing, machine learning
  - More specifically: problems and applications related to waves, optics and quantum mechanics
  - Application areas: geophysics, nano optics, material sciences, medical imaging
- Research webpage:

`https://faculty.sites.iastate.edu/luos`

Some useful information

- **Class Time:** 11:00am – 01:00pm Tuesdays & Thursdays
- **Location:** 449 Carver Hall (Math Computer Lab)
- **Course website:** via Canvas
- **Grades reported:** via Canvas
- **Discussion forum:** via Canvas
- **Office Hours:** 10:00am – 11:00am on Tuesdays & Thursdays

Textbook/Reference



- **Book title:** An Introduction to Parallel Programming
- **Book author:** Peter Pacheco, University of San Francisco
- The book is for reference. Lecture notes are almost self-contained.
- We first consider programming on a single processor
- Later do some OpenMP and MPI

Lectures and Assignments

- Each class is 2 hours long!!!
- Lecture portion will only use up a portion of this < 1 hour
- For the remainder of time, students will have Lab Assignments
- Each Lab Assignment will be saved/uploaded to Canvas for grading by me
- Each Lab Assignment will have a strict due date/time and specific formatting requirements to make grading easier for me
- At the end of semester, you will work on, present, and write-up a Final Project
- Each student must produce a Final Project
- Each Final Project will include a code written by you (hopefully related to your research interests) that makes use of HPC tools discussed in class
- Last 2 weeks of semester will be devoted to the Final Project (Week 15: students work on projects during class time), (Week 16: student presentations)

Grading

- **Lab Assignments:** 50%
- **Final Project:** 50%

- **Total:** 100%

- **Final grades:** (no curve, score rounded to nearest integer)

Score	Letter Grade
≥ 89	at least A-
≥ 78	at least B-
≥ 67	at least C-
≥ 56	at least D-
< 56	F

Rules on Assignments

- You are encouraged to discuss with other students:
  - Read posts and contribute posts to Discussion on Canvas
  - But . . . do not give detailed answers to assignments
- Only submit code that you wrote and debugged yourself
- Copy/paste of non-trivial code is not acceptable
  - non-trivial = more than a line or so
  - includes reading someone else's code and then writing it verbatim
- Use of third party libraries that directly implement the solution of a Lab Assignment or Final Project is strictly prohibited unless specifically asked to do so
- If in doubt, talk to me first

Getting Help

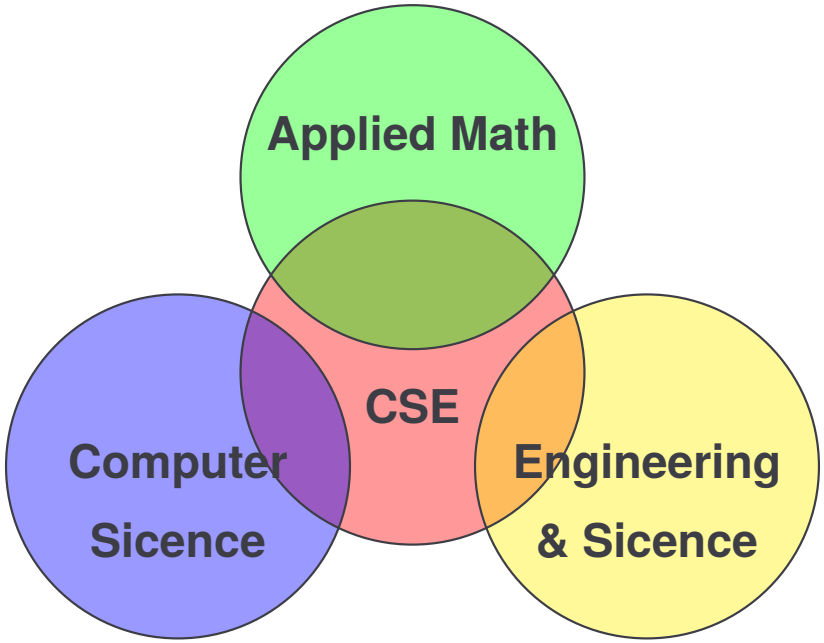
- Please do not email regarding specific questions about Lab Assignments
  - Email me: to schedule a one-on-one meeting outside of office hours
  - Don't email me: to ask clarifications on Problem 3 of the current Lab Assignment (that's what the forum is for)
  - Don't email me: that you can't compile your code (after trying by yourself, this should also go on the forum)
- Any assignment-related question should be posted on the forum
  - I continuously monitor the forum
  - If you can answer a Forum/Discussion post, please do so
  - Keeps all of us on the same page
- The Forum/Discussion can be **VERY** useful (as long as you follow the rules)
- **Office Hours.**

A note on prerequisites

- Prerequisites
  - Undergraduate linear algebra (Math 2070 or Math 3170 at ISU)
  - Vector-vector & matrix-vector multiplications, Gaussian elimination
  - Basic numerical analysis (Math 4810 at ISU)
  - We will not be proving theorems on abstract vector spaces, ...
- As for computer programming ...
  - Most of you have some experience in programming (Java, MATLAB, Python, ...)
  - I will cover material assuming no prior knowledge (as much as possible)
- A comment on numerical methods ...
  - Math 3730/4810/5610/5620/5170/5650/5660/6660/6670 are concerned with numerical methods (i.e., development and theoretical analysis)
  - In Math 5250 we will use some methods as examples, but won't bother with numerical method development or numerical analysis
  - ∴ 3730/4810/5610/5620/5650/5660/5170/6660/6670 are not prerequisites

Why is 5250 a math class?

CSE Defined **[SIAM Review, 43, 163–177, 2001]**



Computational Science & Engineering (CSE)

CSE is a broad multidisciplinary area that encompasses application, applied math, and computer science/engineering.
Computer simulations have become an important part of the research repertoire, supplementing (and in some cases replacing) experimentation.
Going from application area to computational results requires domain expertise, mathematical modeling, numerical analysis, algorithm development, software implementation, program execution, analysis, validation, and visualization of results. CSE involves all of this.

CSE in the ISU Math Department

- We already teach many numerical methods classes:
  - **Math 3730:** intro to numerical analysis
  - **Math 4810:** numerical methods for ODEs
  - **Math 5610/5620:** intro to numerical analysis at the graduate level
  - **Math 5170:** finite difference methods
  - **Math 5650:** continuous optimization
  - **Math 5660:** discrete optimization
  - **Math 6660:** finite element methods
  - **Math 6670:** finite volume methods
- Math department is a long-time contributor to HPC efforts on campus

What is high-performance computing?

HPC Defined

High-Performance Computing

http://insidehpc.com/hpc-basic-training/what-is-hpc/

High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.

World's Fastest Supercomputers (http://top500.org)

The need for speed?

Computations can supplement or replace physical experiments that are:

- too difficult (e.g., building large wind tunnels)
- too expensive (e.g., building a “throw-way” passenger jet)
- too slow (e.g., waiting for climate or galactic evolution)
- too dangerous (e.g., nuclear weapons)
- too controversial (e.g., stem cell research)

HPC can benefit researchers who wish to carry out huge amounts of repetitive calculations on large amounts of data and wish to obtain valid results in a reasonable amount of time.

Application areas include:

- quantum chemistry
- cosmology and astrophysics
- climate dynamics, weather prediction, volcano/earthquake modeling
- crash test simulations, aircraft/spacecraft design
- nuclear reactors, fusion reactors

Phillip Colella's "Seven Dwarfs"

High-end simulation in the physical sciences = 7 numerical methods

- structured grids (including locally structured grids, e.g. AMR)
- unstructured grids
- fast Fourier transform (FFT)
- dense linear algebra
- sparse linear algebra
- particles
- Monte Carlo

Source: Slide from *Software Requirements for Scientific Computing*, Phillip Colella, 2013.

`https://www.krellinst.org/doecsgf/conf/2013/pres/pcolella.pdf`

Objectives of this class

Focus and Objectives of Math 5250

Our focus is more modest but we will cover material that is:

- Essential to know if you eventually want to work on supercomputers
- Extremely useful for any scientific computing project, even on a laptop

Focus and Objectives of Math 5250

Our focus is more modest but we will cover material that is:

- Essential to know if you eventually want to work on supercomputers
- Extremely useful for any scientific computing project, even on a laptop

Efficiently use single processor and multi-core computers:

- Basic computer architecture (e.g., floating point arithmetic, cache hierarchies, pipelining)
- Using Unix (specifically on Mac OS X, Linux)
- Language issues (compiled vs. interpreted, object oriented)
- Specific languages: Python, C
- Parallel computing via OpenMP, MPI

Focus and Objectives of Math 5250

Our focus is more modest but we will cover material that is:

- Essential to know if you eventually want to work on supercomputers
- Extremely useful for any scientific computing project, even on a laptop

Efficiently use single processor and multi-core computers:

- Basic computer architecture (e.g., floating point arithmetic, cache hierarchies, pipelining)
- Using Unix (specifically on Mac OS X, Linux)
- Language issues (compiled vs. interpreted, object oriented)
- Specific languages: Python, C
- Parallel computing via OpenMP, MPI

Efficient programming and good software practices:

- Makefiles, Python scripting
- Debuggers, code development and testing
- Reproducibility

Focus and Objectives of Math 5250

So much material, so little time...

- Concentrate on basics, simple motivating examples.
- Get enough hands-on experience to be comfortable experimenting further and learning much more on your own.
- Learn what's out there to help select what's best for your needs.
- Teach many things "by example" as we go along.
- You'll be expected to read notes and suggested readings.

Note: lectures will also include hands-on demos.

Note: slides and various demo files will be made available through Canvas

HPC-Class: Education Cluster

IOWA STATE UNIVERSITY
OF SCIENCE AND TECHNOLOGY

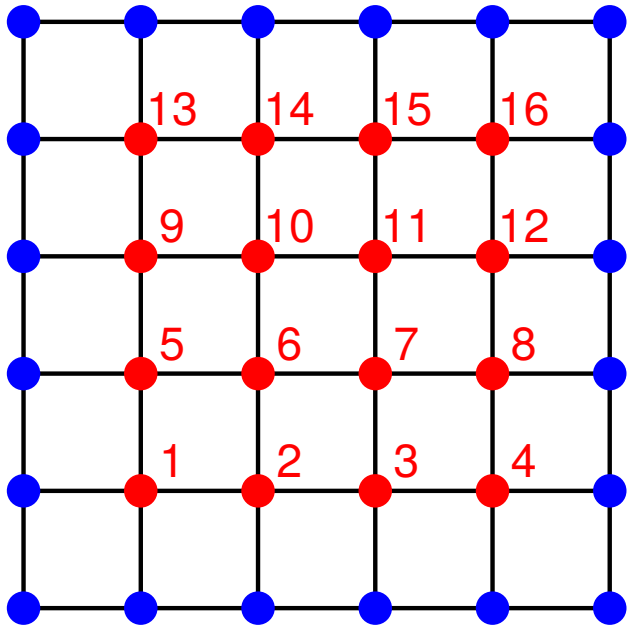
HPC-Class (http://www.hpc.iastate.edu)

System & Equipment

An example to get the semester started

Steady-state heat conduction

- Discretize an $N \times N$ grid with N^2 unknowns



- Assume temperature is fixed (and known) at each boundary point
- neighboring value = node that shares an edge with current node
- Every interior point has **exactly** 4 neighbors
- For each interior point, the steady state value is approximately the average of the 4 neighboring values

Steady-state heat conduction

- Let U_{ij} be the approximate temperature at lattice point (i, j):

$$U_{ij} = U_{i+(j-1)N} \qquad \text{where} \quad i = 1, 2, \ldots, N \quad \text{and} \quad j = 1, 2, \ldots, N$$

- Averaging rule:

$$U_{ij} = \frac{1}{4}\left(U_{i-1\,j} + U_{i+1\,j} + U_{i\,j-1} + U_{i\,j+1}\right)$$

- **(Skipping some details for now)** This averaging rule can be written as a linear system of the form:

$$A\vec{U} = \vec{b}, \quad \text{where} \quad A \in \mathbb{R}^{N^2 \times N^2}, \quad \vec{U} \in \mathbb{R}^{N^2}, \quad \vec{b} \in \mathbb{R}^{N^2}$$

- If $N = 120$, $N^2 = 14400$, but A is **very sparse** (each row of A has only 5 non-zeros), Gaussian elimination is not the best approach
- **Jacobi iteration** (also a poor method, but super easy to implement):

$$U_{ij}^{[k+1]} = \frac{1}{4}\left(U_{i-1\,j}^{[k]} + U_{i+1\,j}^{[k]} + U_{i\,j-1}^{[k]} + U_{i\,j+1}^{[k]}\right)$$