High-Performance Scientific Computing

Applied Mathematics 483/583, Spring 2013

University of Washington, Seattle

Instructor: Randy LeVeque

http://faculty.washington.edu/rjl/classes/am583s2013/

https://canvas.uw.edu/courses/812916

More seats available today in room 206!

Course logistics

Sections for UW students:

	in-class	virtual	online Masters
AMath 483	A	B	
AMath 583	A	D	B

Course logistics

Sections for UW students:

	in-class	virtual	online Masters
AMath 483	A	B	
AMath 583	A	D	B

Also offered on Coursera this year, starting in May.

Assignments and grading

See course webpage / class notes for schedule and assignments.

6 homework assignments, due Wednesday night.

Final project, due Finals week.

These will be turned in by pushing to a bitbucket repository.

Assignments and grading

See course webpage / class notes for schedule and assignments.

6 homework assignments, due Wednesday night.

Final project, due Finals week.

These will be turned in by pushing to a bitbucket repository.

Short quiz associated with each lecture.

Must be completed before the next lecture takes place!

Will require seeing lecture and perhaps additional reading.

Discussion board and other help

Please use the Piazza discussion board.

Getting help from fellow students is a great way to learn.

Discussion board and other help

Please use the Piazza discussion board.

Getting help from fellow students is a great way to learn.

But... please don't give detailed answers to assignments.

You only should submit code that you have written and debugged yourself.

Read and respect the honor code.

TAs and office hours

Two TAs are available to all UW students:

Scott Moe and Susie Sargsyan (AMath PhD students)

Office hours: posted on Canvas course web page

`https://canvas.uw.edu/courses/812916`

Tentative:

M 1:30-2:30 in Lewis 208,

T 10:30-11:30 in Lewis 212 (*)

W 1:30-2:30 in Lewis 208 (*)

F 12:00-1:00 in Lewis 212

(*) GoToMeeting also available for 583B students

My office hours: M & W in CSE Atrium, 9:30 – 10:45.

Applied Math is now in Lewis Hall!

[map]

Outline of this lecture

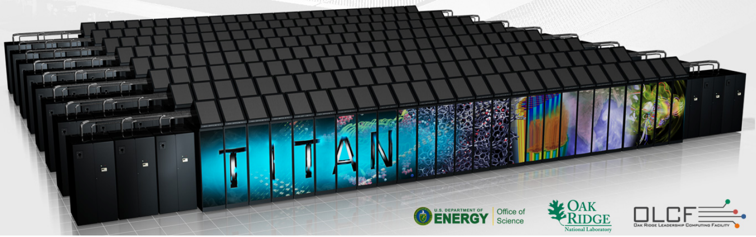
- Goals of this course, strategy for getting there
- Computer/software requirements
- Brief overview of material
- Demo and discussion on heat conduction problem

Overview

High Performance Computing (HPC) often means heavy-duty computing on clusters or supercomputers with 100s of thousands of cores.

"World's fastest computer"

#1. Titan (Oak Ridge National Lab):
560,640 cores, ≈ 20 Petaflops



See `http://top500.org` for current list.

Increasing speed

Moore's Law: Processor speed doubles every 18 months.
$\implies$ factor of 1024 in 15 years.

Going forward: Number of cores doubles every 18 months.

Top: Total computing power of top 500 computers

Middle: #1 computer

Bottom: #500 computer

`http://www.top500.org`

Overview

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 on scientific computing as opposed to other computationally demanding domains, for which somewhat different tools might be best.

Focus and Topics

Efficiently using single processor and multi-core computers

- Basic computer architecture, e.g. floating point arithmetic, cache hierarchies, pipelining
- Using Unix (or Linux, Mac OS X)
- Language issues, e.g. compiled vs. interpreted, object oriented, etc.
- Specific languages: Python, Fortran 90/95
- Parallel computing with OpenMP, MPI, IPython

Focus and Topics

Efficiently using single processor and multi-core computers

- Basic computer architecture, e.g. floating point arithmetic, cache hierarchies, pipelining
- Using Unix (or Linux, Mac OS X)
- Language issues, e.g. compiled vs. interpreted, object oriented, etc.
- Specific languages: Python, Fortran 90/95
- Parallel computing with OpenMP, MPI, IPython

Efficient programming and good software practices

- Version control: Git, bitbucket
- Makefiles, Python scripting
- Debuggers, code development and testing
- Reproducibility

Strategy

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.
  Browse the bibliography in the notes.

Strategy

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.
  Browse the bibliography in the notes.

Lecture notes

- html and pdf versions from class webpage
  (green = link in pdf file)
- Written using Sphinx: Python-based system for writing documentation.
- Learn by example!! Source for each file can be seen by clicking on “Show Source” on right-hand menu.
- Source files are in class bitbucket repository. You can clone the repository and run Sphinx yourself to make a local version.

```
$ git clone https://rjleveque@bitbucket.org/rjleveque/uwhpsc.git

$ cd uwhpsc/notes

$ make html

$ firefox _build/html/index.html
```

Lecture slides

Slides from lectures will be linked from the class webpage

In single-slide form and 3-up form that has room for taking notes.

Note: Slides will contain things not in the notes, lectures will also include hands-on demos not on the slides.

Prerequisites

Some programming experience in some language,
e.g., Matlab, C, Java.

You should be comfortable:

- editing a file containing a program and executing it,
- using basic structures like loops, if-then-else, input-output,
- writing subroutines or functions in some language

You are not expected to know Python or Fortran.

Some basic knowledge of linear algebra, e.g.:

- what vectors and matrices are and how to multiply them
- How to go about solving a linear system of equations

Some comfort level for learning new sofware and willingness to dive in to lots of new things.

Computer/Software requirements

You will need access to a computer with a number of things on it, see the section of the notes on Downloading and Installing Software.

Note: Unix is often required for scientific computing.

Windows: Many tools we'll use can be used with Windows, but learning Unix is part of this class.

Computer/Software requirements

You will need access to a computer with a number of things on it, see the section of the notes on Downloading and Installing Software.

Note: Unix is often required for scientific computing.

Windows: Many tools we'll use can be used with Windows, but learning Unix is part of this class.

Options:

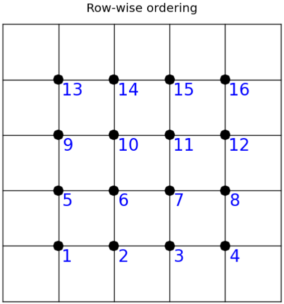
- Install everything you'll need on your own computer, or find a computer with what you need.
- Install VirtualBox and use the Virtual Machine (VM) created for this class.
- Use Amazon Web Services with the Amazon Machine Image (AMI) created for this class.

Demo...

```
$ git clone \
  https://bitbucket.org/rjleveque/uwhpsc.git

$ cd uwhpsc/lectures/lecture1

$ make plots

$ firefox *.png
```

Steady state heat conduction

Discretize on an $N \times N$ grid with N^2 unknowns:



Assume temperature is fixed (and known) at each point on boundary.

At interior points, the steady state value is (approximately) the average of the 4 neighboring values.

Steady state heat conduction

$$u_{i,j} = \frac{1}{4}(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1})$$

Holds for $i, j = 1,\ 2,\ \ldots,\ N$ with $u_{0,j}$ known on boundary.

Gives a linear system $Au = b$, with N^2 equations N^2 unknowns.

Matrix A is $N^2 \times N^2$, for $N = 120$, $N^2 = 14400$.

Steady state heat conduction

$$u_{i,j} = \frac{1}{4}(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1})$$

Holds for $i, j = 1,\ 2,\ \ldots,\ N$ with $u_{0,j}$ known on boundary.

Gives a linear system $Au = b$, with N^2 equations N^2 unknowns.

Matrix A is $N^2 \times N^2$, for $N = 120$, $N^2 = 14400$.

Very sparse: each row of matrix A has at most 5 nonzeros.

Gaussian elimination is not the best approach.

Steady state heat conduction

$$u_{i,j} = \frac{1}{4}(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1})$$

Holds for $i, j = 1,\ 2,\ \ldots,\ N$ with $u_{0,j}$ known on boundary.

Gives a linear system $Au = b$, with N^2 equations N^2 unknowns.

Matrix A is $N^2 \times N^2$, for $N = 120$, $N^2 = 14400$.

Very sparse: each row of matrix A has at most 5 nonzeros.

Gaussian elimination is not the best approach.

Jacobi iteration: (Also a poor method!)

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

Speedup for problems like steady state heat equation

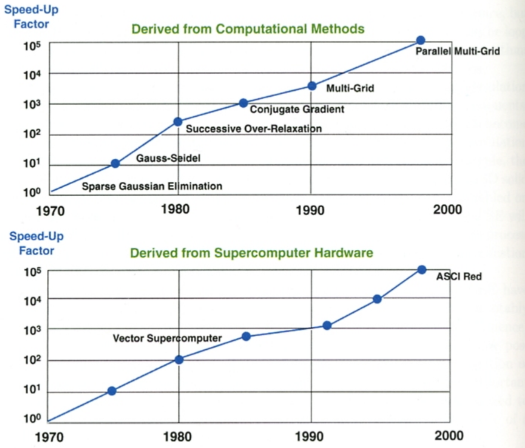


Fig. 2 *Comparison of the contributions of mathematical algorithms and computer hardware.*

Source: SIAM Review 43(2001), p. 168.

Quiz on Lecture 1

Available later today on Canvas webpage:

https://canvas.uw.edu/courses/812916

Class Virtual Machine

Available online from class webpage

This file is large! About 765 MB compressed.

After unzipping, about 2.2 GB.

Also available from TAs on thumb drive during office hours,

Or during class on Wednesday or Friday.