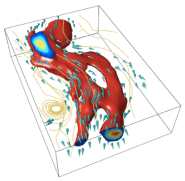
CS 404: Survey and Use of Software Libraries for Scientific Computing



Andrew Pershing
3134 Snee Hall
ajp9@cornell.edu
255-5552

Outline

- Course Description
- Details
- Policies
- Intro to CIS Tools Curriculum
- Basic Concepts
- Intro to vectors and matrices

Course Goals

- This course will:
  - Survey available software libraries for scientific computing
  - Discuss several library formats and how to use them
  - Consider the legal/ethical issues associated with using someone else's code

Syllabus

1. Intro, Philosophy
2. Types of libraries & where to find them
3. Using libraries I: compiling and linking
4. Survey of numerical methods and available libraries
5. Using libraries II: inter-language operability
6. Calling MATLAB
7. Getting what you pay for--legal and moral issues
8. Java packages or DLL's
9. Graphics & GUIs
10. Intro to MPI and parallelism
11. Cornell Theory Center and Velocity
12. MPI Lab

Course Business:

- http://www.cs.cornell.edu/Courses/cs404/2002sp
  - Contains syllabus, lecture notes, examples, homework
- Location
  - Mondays--211 Upson
  - Wednesdays and Fridays--ACCEL Green Room
- Office Hours
  - Wednesday & Thursday, 12-2 in 3134 Snee (or by appointment)
- Registration:
  - get my signature or CS Undergrad office (303 Upson)
  - S/U only, 1 credit
  - Last day to add/drop: Monday, Apr. 1!

Requirements

- No official text
- Need to find a computer where you can
  - 1. edit text and do e-mail
  - 2. compile code (mostly C)
  - 3. Check out ACCEL Facility in Carpenter Library, departmental labs

Course Policies

- 3 assignments: due Friday, 5PM by e-mail
- If you complete each assignment on time and demonstrate a basic command of the material, you will pass!
- Also, attendance in Wed. & Fri. labs is REQUIRED
  - can miss one lab, but you are responsible for material

The Contract

- This course operates as a contract between you and me
- I agree to:
  - Begin and end lecture on time
  - Put lecture notes on website before lecture
  - Be available during office hours
  - Make the assignments of reasonable length (~2 hours) focusing on material from lectures

The Contract

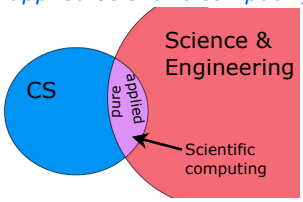
- By registering for the course, you agree to:
  - Arrive on time
  - Participate in the course by asking questions and coming to office hours
  - Turn in your assignments on time
    - Late work will not be accepted and will jeopardize you chance of passing!
    - The only exceptions are for documented, university-sanctioned reasons such as severe illness or by prior arrangement made w/ me 3 days before (includes religious holidays, sports, etc.)

CIS and FCI

- *Cornell University has recognized that computing and information science has emerged as a key enabling discipline vital to nearly all of its scholarly and scientific pursuits.*
- *The Faculty of Computing and Information is founded on the recognition that the ideas and technology of computing and information science are relevant to every academic discipline.*
- *We are united in the need to bring together a core of faculty in this field from across the traditional colleges.*

CIS Tools Curriculum

- CS 404 (should be CIS 404) is the fourth in a series of courses designed to teach *applied scientific computing*



CIS Tools Curriculum

- "Pure" Scientific Computing
  - Focus is on algorithms for general problems such as optimization, linear systems, differential equations
  - Concerned with accuracy, stability, and efficiency of these algorithms
- "Applied" Scientific Computing
  - How to apply general algorithms to solve scientific problems
  - Algorithms are "black boxes" that we string together to get our work done

CIS Tools Curriculum

- Fall: MATLAB
  - 401: the basics
  - 402: visualization
- Spring: General tools
  - 403: Developing scientific computer programs (compilers, debuggers, managing large projects)
  - 404: Numerical libraries

Why a course on libraries?

- A large part of the power and popularity of computers stems from their ability to make copies
  - MP3 files
  - Software
  - Code
- Software libraries are a way of distributing subroutines to solve related problems
- There are many reasons to use libraries, but it is not always easy
- This course will try to make the range of software available to you

What can libraries do?

- Libraries have been created for most simple and many complex tasks
  - Reading/writing data, especially standard formats
  - Standard CS problems like searching and sorting
  - Mathematical functions, random numbers
  - Linear algebra: matrix & vector manipulation, matrix analysis, linear systems
  - Ordinary differential equations
  - Tools for PDEs, especially meshing/gridding
  - graphics

Why use a library?

- Reduce development time
  - By using a library, you save yourself the time of writing and debugging the code
- Standardize your software
  - Using the same libraries as others in your field makes it easier to compare results and describe techniques
- Improve performance
  - Libraries, especially for low-level functions, are often heavily optimized and tuned to specific systems

Applied Scientific Computing

- Emphasis is less on developing new algorithms, rather, it is on obtaining new scientific results.
  - We are either running a simulation, or analyzing data (perhaps from a simulation).
  - We need to be able to develop new code or modify existing code to fit our needs
  - We should make this process easier for ourselves or colleagues the next time.
  - We need to get the code to run on our system.
  - We will need to debug the code and verify that it is solving the correct problem.

Library issues

- Getting a library to work can be tough
  - especially calling one language from another
- If you use a library, you are using someone else's code
  - Do you need to pay?
  - Can you pass this on to a colleague?
  - How should you acknowledge the libraries' authors?

Intro to Vectors and Matrices

- Numerical solutions to many mathematical problems involve operations with vectors and matrices
- The simplest and most common numerical libraries are for these problems
  - BLAS--Basic Linear Algebra Subroutines
  - LAPACK--Linear Algebra PACKage

Vectors

- A vector is a collection of numbers that together have some meaning
  - Example: position of a particle in 3D

$$x = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

  - Key property of a vector is its length (dimension)

Vector Operations

- scalar multiplication
  - $a*x=[a*x_1, a*x_2, \ldots, a*x_N]^T$
- vector addition
  - $x+y=[x_1 + y_1, x_1 + y_2, \ldots, x_N + y_N]^T$
- AXPY
  - Combination of scalar mult & vector add
  - ax+y
  - Most processors have multiple adders and multipliers, so AXPY's can be done quickly
- dot product
  - $x \cdot y = x_1*y_1 + x_1*y_2 + \ldots + x_N*y_N$

Matrices

- A matrix is a collection of vectors

$$A = \begin{bmatrix} a_1 & a_2 & \dots & a_N \end{bmatrix}$$

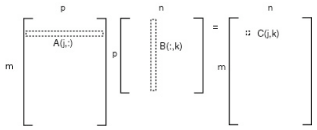
$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & & \vdots \\ a_{M1} & a_{M2} & \dots & a_{MN} \end{bmatrix}$$

Matrix Operations

- scalar multiplication
  - $c*A=[c*a_1, c*a_2, \dots, c*a_N]$
- matrix addition
  - $A+B=[a_1 + b_1, a_1 + b_2, \dots, a_N + b_N]$
  - Only works if A and B are the same size
- matrix multiplication
  - A*B=C
  - A is m-by-n, B is n-by-p, then C is m-by-p

Matrix Multiplication

- A is m-by-p and B is p-by-n then C is m-by-n:
  - C(i,j)= a(i,1)*b(1,j)+a(i,2)*b(2,j)+ … + a(i,p)*b(p,j)



- Another view:
  - C(i,j)=a(i,:)*b(:,j);
    - 1-by-p p-by-1 answer is 1-by-1

Linear Systems

- We can represent a system of linear equations as a matrix-vector product
  - $a_1*x + b_1*y = w_1$
  - $a_2*x + b_2*y = w_2$

$$\begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

  - Ax=w

BLAS

- BLAS contains routines for elementary vector and matrix problems
- BLAS are often heavily optimized for a particular OS/processor/compiler combination
  - can improve performance
  - check compiler documentation

BLAS

- BLAS are grouped into 3 levels
  - Level 1--vector operations
    - AXPY
    - Dot product
  - Level 2--matrix-vector operations
    - Matrix vector product
  - Level 3--matrix-matrix operations
    - Matrix-matrix products

LAPACK

- LAPACK
  - provides routines for linear algebra based on BLAS primitives
    - Solution of linear systems
    - Matrix factorizations
    - Eigenvalues
- For more info on BLAS or LAPACK
  - www.netlib.org
  - Come to ACCEL on Wednesday