INTRO TO MODERN COMPUTING TECHNIQUES
FOR LU FACULTY & STUDENTS
Organized by LU-CCRE

1. An Introduction to Linux

This tutorial is for the beginning Linux user. This is meant to get the user acquainted with some of the basic principles of the Linux operating system. Linux is a very powerful free operating system (OS) which includes a multitude of tools for programmers and system administrators. It is heavily used in the academic and scientific communities because it is so versatile, and has over 40 years of development of scientific tools. It is a robust, stable, and flexible system that allows programmers to tailor the OS to run on a variety of hardware from phones (Android) to supercomputers. This tutorial will demonstrate the download and installation of Linux on desktop and laptop machines and the introduction to the shell commands.

2. An Introduction to Fortran Programming

Fortran is one of the premier languages for numerical analysis and high performance computing. On average about half the applications on supercomputers are written in Fortran. Fortran has a long history. It was developed in 1954 as the first high-level programming language. Legacy code conforming to the former FORTRAN 77 (and earlier) standards is now called "old" Fortran. About 25 years ago, a major effort was made to modernize the language. Since then, updates have been made to the standard: Fortran 95, Fortran 2003, and Fortran 2008. Fortran continues to evolve steadily to fit the needs of the (super)computing community.

3. An Introduction to C Programming

The C programming language is heavily used in the scientific and high performance computing community, and also happens to be the same language that many operating systems are written in. Thus it is important for scientists and engineers to have a good understanding of the language to assist them in their computing needs. This document is provided for the beginning programmer who has an interest in learning the effective use of the C language.

4. An Introduction to Python

Python is a programming language designed with ease of programming and readable code as its foremost goals. Python has risen to prominence in scientific computing as an excellent tool for doing data conversions, scripting parameter studies, and in general providing the “glue” to hook together many pieces of the scientific workflow. A quick overview of the language is presented, along with a few tricks to maximize the utility of Python for engineering and science modeling. This is not intended as a comprehensive syntax review for Python; rather, it’s enough to get started for beginners (with pointers to more detailed resources), plus a few hints for improvement once you make it out of the beginner phase. Although we will be learning Python on UNIX/Linux, nearly all of the Python language operates the same on Windows; this platform-independence is another contributor to Python's wider appeal.

5. Introduction to R for Statistics and Data Analysis

R is an integrated environment for doing statistical and data analysis. It was created in the 1990s and is now maintained by the R Development Core Team. It has excellent data handling capabilities and can construct production quality graphics. The R language is an object-oriented, interpreted language that allows the user to employ loops, branching and functional definitions so there is a considerable level of program control. R is an open source version of S, a widely respected statistical package, written in a mixture of Fortran, C and R. It will run on Linux, MAC and Windows operating systems. The base R distribution contains most common statistical procedures, including regression and ANOVA, linear and nonlinear modeling, classical parametric and nonparametric tests, cluster analysis, and time series analysis.

6. An Introduction to the Message Passing Interface (MPI)

This is a short introduction to the Message Passing Interface (MPI) designed to convey the fundamental operation and use of the interface. This introduction is designed for readers with some background programming Fortran, and should deliver enough information to allow readers to write and run their own (very simple) parallel Fortran programs using MPI. MPI is a library of routines that can be used to create parallel programs in Fortran77 or C. Standard Fortran and C include no constructs supporting parallelism so vendors have developed a variety of extensions to allow users of those languages to build parallel applications. It is a library that runs with standard Fortran or C programs, using commonly-available operating system services to create parallel processes and exchange information among these processes.

7. Octave/MATLAB Programming

MATLAB was originally created in the 1970s. Although MATLAB itself provides a great deal of basic matrix manipulation, plotting, and general purpose scientific programming capability, a large part of its functionality is actually available through specialized "toolboxes" such as the Optimization toolbox, the Statistics toolbox, the Signal Processing toolbox, the Bioinformatics Toolbox, the Image Processing Toolbox, etc. GNU Octave is a free open source software package which is "mostly compatible" with MATLAB. It provides a very similar command line interpreter interface, and implements most of the functionality as the commercial MATLAB product. Octave has been under development since 1992.

8. Introduction to GPGPU and CUDA Programming

Compute Unified Device Architecture (CUDA) is a programming API created by NVIDIA for general-purpose usage of its GPUs (GPGPU). CUDA supports a number of high level programming languages such as C and Fortran, and it features a large performance library, easing the task of GPU programming. It provides a means of utilizing a heterogeneous computing environment where users can perform serial or coarse-grain parallel computing using CPUs, while offloading massively parallel tasks to the GPUs. As the hardware design for GPUs is optimized for highly parallel applications, the programming model is very different from the traditional serial programming model using CPUs. The goal of this module is to expose you to the basic CUDA programming concepts and techniques.

Upon completion of four (4) of these modules attendees will be awarded a **certification of accomplishment** for **Intro to Modern Computing Techniques** from **CCRE.**