Introduction to Scientific Computing

Pablo Brubeck[1]

[1]Department of Physics
Tecnologico de Monterrey

September 17, 2016

Outline

Outline

The 4 steps of Scientific Computing

1. Problem
2. Equation
3. Algorithm
4. Visualization

Scientific Computing problems may be classified into:

- Root finding (Solve a system of equations)
- Optimization (Maximize or minimize a cost function)
- Modeling of data (Curve fitting, interpolation, extrapolation)
- Numerical evaluation (Functions, series, derivatives, integrals)
- Differential equations (Boundary Value Problems, Initial Value Problems)

Approaches of Scientific Computing

- Analytical
- Numerical
- Algorithmic
- Heuristic

Outline

Solve the problem first,
then write code.

High-level programming provides the most straight-foward solutions

A high-level approach relies on abstraction and a programming language rich in functions.

- Python (Fuctional)
- MATLAB (Numerical)
- COMsolve (Finite Element Method)
- Mathematica (Functional)

Low-level programming provides the most efficient solutions

A low-level approach ussually requires more lines of code, but it is the way to achieve the most efficient solutions.

- Fortran (Old school, the latin of programming languages)
- C++ (Universal)
- Java (Multi-platform applications)

Explicit data types and memory management.

Meaningful function and variable names may save your life.

Naming conventions Consistency

Tips and tricks.

- You can indent several lines of code at once.
- Decrease indent shortcut (Shift-Tab)

Commonly, your problem has already been solved or at least partially solved

The internet was invented for scientific collaboration. Google is your friend!

Here are some usefull sites

- Stack Overflow
- Stack Exchage (Super User, Mathematics, Physics)
- Physics Forums
- Online documentation

Popular Scientific Computing libraries

User firendly

- NumPy, SciPy, matplotlib (MATLAB inside Python)
- Armadillo (MATLAB inside C++)
- Plotly (Online data visualization)

State of the art (God Tier)

- Intel MKL (Math Kernel Library)
- GSL (GNU Scientific Library for C/C++)
- BLAS and LAPACK (Linear Algebra)
- FFTW (Fastest Fourier Transform in the West)
- CUDA, OpenCL (GPU programming)
- OpenGL, Vulkan, Direct3D (graphics)

Specialized Scientific Computing libraries

- Chebfun (Spectral Methods)
- NFFT (Non-equispaced Fast Fourier Transform)
- ARPACK (Sparse Eigensolver)
- SCPACK (Schwartz-Christoffel Conformal mapping)
- DistMesh (Simple mesh generator in MATLAB)

Outline

A Boundary Value Problem (BVP) is a differential equation

Find $y(x)$ that satifies a differential equation and boundary conditions.

Many natural-occuring BVPs are second order and linear

Linear problems