Introduction to Scientific Computing

Pablo Brubeck

Department of Physics
Tecnologico de Monterrey

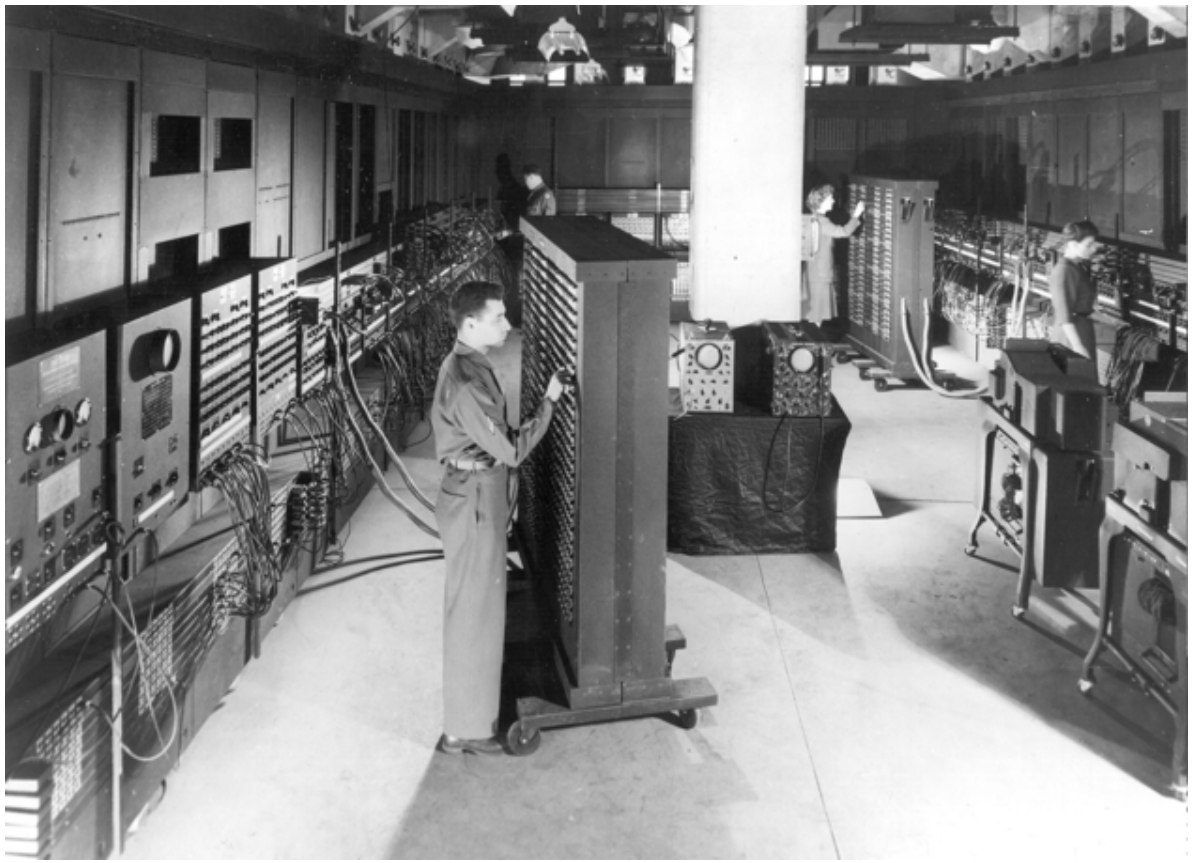
October 7, 2016

Outline

Outline

Ada Lovelace wrote the first computer program in 1842 to calculate the Bernoulli Numbers.

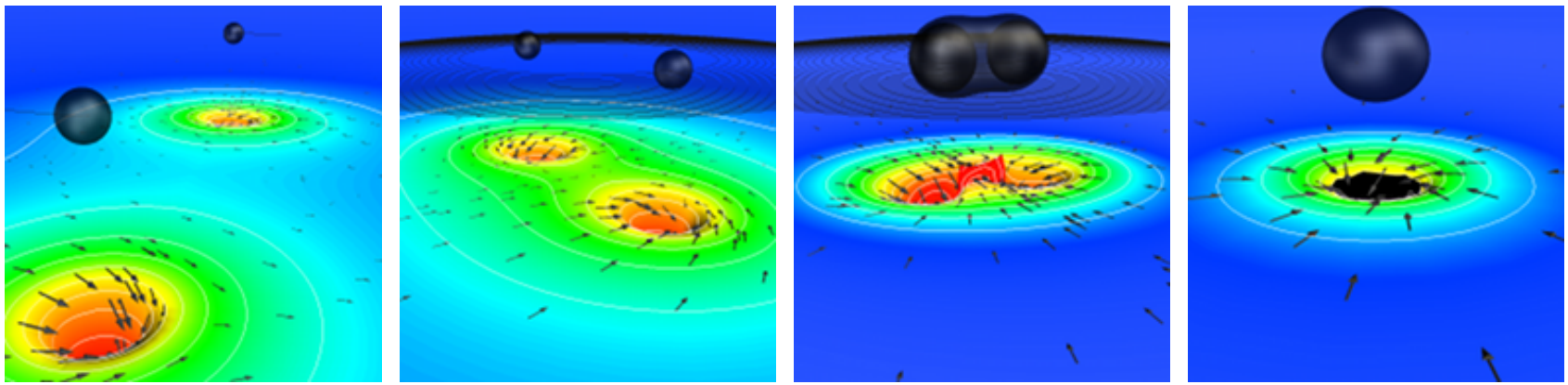
The first programmable computer was the ENIAC, used for scientific and military applications.



Margaret Hamilton led the development of the code for Apollo 11.

This code is now available on GitHub.

Today we are using supercomputers to simmulate black holes



Spectral Einstein Code (SpEC) simulation of inspiral and merger of two black holes.

Scientific Computing problems may be classified into:

- **Root finding** Solve a system of equations

Scientific Computing problems may be classified into:

- **Root finding** Solve a system of equations
- **Optimization** Maximize or minimize a cost function

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

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

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** By hand, Computer Algebra Systems.

Approaches of Scientific Computing

- **Analytical** By hand, Computer Algebra Systems.
- **Numerical** Finite Differences, Spectral Methods.

Approaches of Scientific Computing

- **Analytical** By hand, Computer Algebra Systems.
- **Numerical** Finite Differences, Spectral Methods.
- **Deterministic** Graphs, Dynamic Programming.

Approaches of Scientific Computing

- **Analytical** By hand, Computer Algebra Systems.
- **Numerical** Finite Differences, Spectral Methods.
- **Deterministic** Graphs, Dynamic Programming.
- **Heuristic** Montecarlo, Genetic Algorithms.

Approaches of Scientific Computing

- **Analytical** By hand, Computer Algebra Systems.
- **Numerical** Finite Differences, Spectral Methods.
- **Deterministic** Graphs, Dynamic Programming.
- **Heuristic** Montecarlo, Genetic Algorithms.
- **Machine Learning** Aritficial Neural Networks.

Outline

Solve the problem first, then write code.

Coding will always take an additional effort.

Solve the problem first, then write code.

Coding will always take an additional effort.

You may want to apply the **Feynman Algorithm** before writing the code:

Solve the problem first, then write code.

Coding will always take an additional effort.

You may want to apply the **Feynman Algorithm** before writing the code:

1. Write down the problem.

Solve the problem first, then write code.

Coding will always take an additional effort.

You may want to apply the **Feynman Algorithm** before writing the code:

1. Write down the problem.
2. Think real hard.

Solve the problem first, then write code.

Coding will always take an additional effort.

You may want to apply the **Feynman Algorithm** before writing the code:

1. Write down the problem.
2. Think real hard.
3. Write down the solution.

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)
- COMSOL (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 (High perfomance computing, old school)
- C++ (Low-level memory management)
- Java (Multi-platform applications)

Explicit data types and memory management.

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

Reference books

- **Problem solving and programming Java/C++** Savitch
- **Numerical Analysis** Burden and Faires
- **Numerical Recipes** Press, Teukolsky et al
- **Spectral Methods in MATLAB** Trefethen
- **Computational Science and Engineering** Strang
- **Game Physics** Eberly

Popular Scientific Computing libraries

User firendly

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

Popular Scientific Computing libraries

User firendly

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

God tier

- **GSL** GNU Scientific Library for C/C++
- **Intel MKL** Math Kernel Library
- **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** Numerical computing with functions
- **DistMesh** Simple mesh generator in MATLAB
- **NFFT** Non-equispaced Fast Fourier Transform
- **ARPACK** Large scale eigenvalue solver
- **SCPACK** Schwartz-Christoffel conformal mapping
- **MultiParEig** Multiparameter eigenvalue problem

Outline

A Boundary Value Problem (BVP) is a differential equation

Find $u(x)$ that satifies a differential equation and boundary conditions. Example:

$$\frac{d^2u}{dx^2} = 1000cos(5\pi x)e^{-x^2}$$

With boundary conditions $u(-1) = 2$ and $u(1) = -1$.

Many natural-occuring BVPs are second order and linear

Linear BVP (and some non-linear) may be discretized into matrix equations.

$$Ax = b$$

Numerical linear algebra has provided many succesfull methods for scientific computing problems.