Why computational physics?

- Why/when computers instead of analytical (exact) approach?

Answer: When it is impractical or impossible to find an analytical solution! (Not when it's just hard to do!)

- When an precise theory exists (e.g. Newton's laws, Schrodinger Eq., statistical mechanics, electrodynamics), typically the "exactly solvable" applications are very few.

- To find solutions that can be tested against experiments, numerical methods become an important (or even essential) approach.

Important problems/applications

- Large systems of coupled differential equations

  Examples: Schrodinger's equation, trajectories of many interacting particles/masses

- Large systems of linear equations (linear algebra/matrices)

  Examples: Normal modes, solutions to differential equations in a basis, eigenvalues/eigenvectors

- Nonlinear differential equations

  Examples: Fluid dynamics, plasma physics, chaos

- Analysis of large amounts of data

  Examples: Spectral methods, fitting data to functions

What tools are commonly used/available?

- Ordinary desktop/laptop

  Convenient, cheap, limited in power, serial

- Supercomputer facility

  Extremely powerful, usually parallel machines, run by experts, funded by govt. agencies, competitive grants. National Center for Supercomputer App. (NCSA), NERSC, etc.

- Beowulf computer

  Large parallel machine built from desktops,“do-it-yourself”, relatively cheap, widely used

What other tools might one use?

- Software tools/numerical libraries

  Tools for common numerical tasks, e.g. manipulation of large matrices in linear algebra

- Software for specific applications

  Fluid dynamics, electronic structure of molecules and crystals, modeling thermodynamics of alloys, etc.

- As computational methods continue to become more widespread, strong chance of “reinventing the wheel”.

Where do we start?

- Extremely broad topic (all areas of physics!)… let's start with ordinary differential equations

- Review of solutions of ordinary linear diff. eq.

Pseudocode

- Independent of any computer language
- General “outline” of code
- Might not contain actual details/algorithm

- Declare variables and arrays
- Initialize variables
- Do the calclation (implement algorithm)
- Store (output) the results

Fortran 77 (and programming) basics

```
PROGRAM    program-name

   IMPLICIT   NONE
   [specification part]

   [execution part]

STOP
END

   [subprogram part]
```

Define variables, constants, arrays

The program code!

End the main program

Subroutines
(e.g. matrix diagonalization)

Why computational physics?

Why computational physics?

Why computational physics?