P3800 Computational Physics

Nearly all problems in science today require

- approximations
- computers, often big ones, for

1 - Calculus: integration, differentiation, solving ODE's and PDE's

2 - Matrices (Linear algebra): solving systems of eq$^{\underline{n}}$s, eigenvalues

3 - Simulations: many degrees of freedom yield emergent behaviour, e.g. particles crystallization

Three main tools to investigate physical phenomena

1 - Experiment

2 - Mathematical Physics - paper and pencil theory

3 - Computation

All "easy" problems were solved long ago (by def$^{\underline{n}}$?) and involved simplifying assumptions

e.g. - projectile motion $\rightarrow$ neglect air resistance

- spring force $F = -kx$ valid for small oscillations

- (QM) H-atom can be solved exactly, but He and H^- require heavy computations

For system with many particles / degrees of freedom

e.g
- molecules in a liquid
- star in a galaxy
- electrons in a metal
- climate model

computational methods are often the only way to get useful results.

When number of particles is too large to handle directly, their effects are modelled in a statistical way, e.g. Brownian motion

Computational Physics – two main branches (although separation is not always clear)

- solving equations e.g. $F = ma$

  Theory yields an eq^{n} that describes the system (e.g. projectile). Solving that eq^{n} yields a theoretical realization of the system (projectile's trajectory).

- simulations : No analytic theory for describing the phenomenon of interest. Instead, use theory of underlying components of a system and see how phenomenon arises.

e.g. Pressure in a container full of a non-ideal gas.

In all cases computations involve discrete manipulation of stored information

calculus $\Rightarrow$ discretized representation of continuous variables

$$\text{"}x\text{"} \Rightarrow x_1, x_2, x_3 \ldots$$

and equations (ODE's , PDE's)

Infinitesimal differences not possible $\frac{dy}{dx} \rightarrow \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$

$\rightarrow$ Taylor series / expansion is a big tool.

simulations $\Rightarrow$ e.g. particles, discrete by nature.

This course is a first step towards high performance computing (HPC) to solve physical problems

Main languages of HPC today are C, C++, Fortran (see Computer Basics slides)

16-bit floating point example

$\left\{ \begin{array}{l} 1 \text{ sign bit} \\ 5 \text{ bits for exponent} \\ 10 \text{ bits for mantissa} \end{array} \right.$

$11/2 = 5.5$

$(5)_{10} = (101)_2 \quad (1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0)$

$(0.5)_{10} = (0.1)_2 \quad (1 \times 2^{-1} = 0.5)$

$\therefore \quad (5.5)_{10} = (101.1)_2$

$= (1.011)_2 \times (2)_{10}^{2}$ ← exponent

leading 1 is the hidden bit

10-bit mantissa m

0110000000

exponent = a − bias

$2 = a - 15$

$a = (17)_{10} = (10001)_2$

16-bit bitstring for $(5.5)_{10}$

$$\underset{\text{sign bit (+ve is 0)}}{0} \; \underbrace{10001}_{a} \; \underbrace{0110000000}_{m}$$

Convert 19.13 to binary

$19 \div 2 = 9$ remainder 1 ← this is the least significant bit

$9 \div 2 = 4$ r 1

$4 \div 2 = 2$ r 0

$2 \div 2 = 1$ r 0

$1 \div 2 = 0$ r 1 ← most sig. bit

$(19)_{10} = 10011$

$0.13 \times 2 = \underline{0}.26$ carry 0 ← most sig. bit

$0.26 \times 2 = \underline{0}.52$ carry 0

$0.52 \times 2 = \underline{1}.04$ carry 1

$0.04 \times 2 = 0.08$ carry 0

$0.08 \times 2 = 0.16$ carry 0

$0.16 \times 2 = 0.32$ c 0

$0.32 \times 2 = 0.64$ c 0

$0.64 \times 2 = \underline{1}.28$ c 1

$0.28 \times 2 = 0.56$ c 0 ...

$(19.13)_2 = 10011.0010000010\ldots$