Computational Physics (6810): Session 9

Dick Furnstahl
Nuclear Theory Group
OSU Physics Department
March 8, 2017

Session 8 Stuff

Session 9 Overview
Damped, driven harmonic oscillator

\[ \ddot{x} + \gamma \dot{x} + \omega_0^2 x = f(t) \quad \dot{x} \equiv dx/dt, \quad \ddot{x} \equiv d^2x/dt^2 \]

- *linear* equation \(\implies\) only single powers of \(\ddot{x}, \dot{x}, x\)
- \(\implies\) important for superposition:

\[ x_{\text{total}}(t) = x_{\text{homogeneous}}(t) + x_{\text{particular}}(t) \]

- \(x_{\text{homogeneous}}(t)\) will always be damped \(\implies\) transient!
- particular solution will have \(x \propto e^{i\omega_{\text{ext}} t}\)

\[ \implies (-\omega_{\text{ext}}^2 + i\omega_{\text{ext}} \gamma + \omega_0^2) e^{-i\omega_{\text{ext}}t} = A e^{-i\omega_{\text{ext}}t} \]

- so the driving frequency \(\omega_{\text{ext}}\) *is* the frequency in the linear domain \(\implies\) green dots are on top of each other
- What if nonlinear? More interesting possibilities!
Chaos

- Characteristics of chaos (see Session 8 notes)
  - past behavior not repeated (not periodic)
  - deterministic but not predictable, because uncertainty (or imprecision) in initial conditions grows exponentially in time
  - system has distributed power spectrum (see Mathematica notebooks)

- Necessary conditions for chaos
  - $\geq 3$ independent variables and the equations have nonlinear terms coupling
  - Three equations for the pendulum (with $\phi = \omega_{ext} t$)

$$
\frac{d\theta}{dt} = \omega \\
\frac{d\omega}{dt} = -\alpha \omega - \omega_0^2 \sin \theta - f_{ext} \cos \phi \\
\frac{d\phi}{dt} = \omega_{ext}
$$

- Session 10: Mathematica notebook pendulum.nb
  - gives results for Session 8 “Looking for Chaos”
  - power spectrum: what frequencies are in the $\theta(t)$ plot?
Using the GDB debugger

- GDB is broken on Macs, incomplete on Cygwin → Linux
  - You can run from a Mac or Windows machine by using ssh to the machine called fox (as described)
- Introduction to concept of debuggers; others will be available
- Contrived example to expose you to segmentation faults
- Use the debugger to track down where seg faults occur

Segmentation faults

- From accessing memory that doesn’t belong to your program
  ```c
  double x[10];
  x[10] = 5; // why does this seg fault?
  ```
- Other segfaults: stack overflow; dereferencing unassigned pointers:
  ```c
  int *my_ptr=0;
  *my_ptr = 3;
  ```
Session 9 Stuff (cont.)

- **Profiling:** finding out where your program spends its time
  - Code first for correctness and clarity; then worry about speed
  - 90/10 rule: 90% of the time is spent in 10% of the code, and only 10% of the time in the remaining 90% of the code
  - So find out where the program spends its time
  - gprof counts *statistically*; subject to fluctuations but usually you only care about the coarse distribution of time

- **Optimization by the compiler**
  - Knuth: “We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil”
  - See Wikipedia “Compiler Optimization” for good background discussion
  - It matters *a lot*
  - Use `-O0` (no optimization) when debugging
  - Use (at least) `-O2`; be careful with `-O3` and higher
  - Default optimization is compiler/installation dependent
    - specify in makefile
Session 9 Stuff (cont.)

- Example optimizations done by you or behind the scenes
  - `-fgcse` “global/local common subexpression elimination”

\[
\begin{align*}
a &= b \times c + g; \\
\text{tmp} &= b \times c \\
\implies a &= \text{tmp} + g; \\
d &= b \times c \times d; \\
d &= \text{tmp} \times d
\end{align*}
\]

- switch loops to improve “locality of reference”
  - e.g., use fast memory sequentially
  - branching (e.g., if statements) can prevent prefetching of instructions \(\implies\) cut down to enhance predictability

- Revisiting `area.cpp` with a C++ class
  - Where are objects created and destroyed?
  - Extending the class
  - private vs. public variables

- `rsync`: backup/mirror files *but only the differences!*