Physics -

Idea - intuitions for simulation
many people need for projects
preview of coming weeks

Topics - Newton's Laws
ODE's ← STIFFNESS
Beyond Particles
Penalty Methods
Beyond Penalty Methods

Why Physics?
Why not Physics?

Realism
Things simple enough to be characterized
CONTROL
STABILITY
PREDICTABILITY
Physics II

Newton's Laws (see 10-2)

I - particle at rest remains at rest or continues to move in a straight line.

II - accel of a particle is proportional to, and in the same direction as, the net force acting on the particle.


\[ \frac{\partial p}{\partial t} = \sum f \quad \text{or} \quad f = m \dot{x} \]

The world is an ODE!

PARTICLES

Aristotelian Physics

\[ F = MV \]  ← why

Pushing a particle around
GENERALIZATION
- Any Forces magic, hack, ....
- "generalized particles"

1. a particle
2. n particles \( x, f \) are \( 3 \times n \) vectors
3. n particles, different masses \( M \) is a diagonal matrix

Change of co-ordinates
- Change units \( \text{(meters to feet)} \)
  \[ Sf = m Sa \quad \leftarrow \text{put converters on both sides} \]
- move to polar co-ordinates
Physics Lectures

Day 1 -
Basics of Simulation

Why

Newton's Laws - the generalizations we'll need

Solving ODEs

Basic Idea

Forward vs. Backwards Euler

Time step vs. multiple evaluation

RK2 / RK4

Adaptive step size

What forces?
Solving an ODE

\[ f = \text{derivative of function} \]

\[ \dot{x} = f(x, t) \quad \leftarrow \text{what is } x(t)? \]

Initial value problem gives \( x(t_0), f \), find \( x(t+h) \)

**Step size**

\[ \text{example: spring} \]

\[ x = [\dot{x}], \dot{x} = f(x) = \begin{pmatrix} -1x \end{pmatrix} \]

How?
1. Assume \( f \) is constant over step
   \[ x_{t+h} = f(x(t), t) \]
   \[ x(t+h) = x(t) + h \dot{x}_t \]

Problem: what if \( f \) is not constant (it never is)
- Make \( h \) smaller \( \rightarrow \) better approximation
- Take lots of small steps (if you want a big step)
- Have "more like constant" functions

Explicit (evaluate \( f \)) Euler (piecewise constant approx)

**Numerical ODE solving**

All you know about \( f \) is how to evaluate it

**Time / Stiffness trade off**

Equivalent to take smaller step or softer spring

Problem:
- Want stiff things \( (f \propto \text{non-linear } f) \)
- Want big timestep
- Want robustness \( (\text{don't know if step size is too big}) \)
What if you allow two evals?

given $t, x(t)$ find $x(t+h) \Rightarrow x_0 \Rightarrow x_1$

1. Two euler steps:

   - $x_0 = f(x_0)$
   - $x_{\frac{h}{2}} = x_0 + \frac{h}{2} \cdot \hat{x}_0$
   - $\hat{x}_{\frac{h}{2}} = f(x_{\frac{h}{2}})$
   - $x_1 = x_{\frac{h}{2}} + \frac{h}{2} \cdot \hat{x}_{\frac{h}{2}}$

2. Use both estimates of $f$ to get a better model

   fit a line to the points

   [Diagram of line through points A and B]

   note piece $A \approx$ piece $B$ so \[ \text{line} = \text{tabulation} \]

   - $x_0 = f(x_0, t)$
   - $x_{\frac{h}{2}} = x_0 + \frac{h}{2} \cdot \hat{x}_0$
   - $\hat{x}_{\frac{h}{2}} = f(x_{\frac{h}{2}}, t + \frac{h}{2})$
   - $x_1 = x_0 + h \cdot \hat{x}_{\frac{h}{2}}$

Is this better than #1?

depends - is a line better than piecewise constant?

usually higher order (line) beats piecewise constant

Trick: Adapt step size.

if two approaches give different answers, then too non-linear

if RK2 \approx RK1 \text{, then RK1 is good enough}
What if 3 evals?

1. Could try to "fit" a quadratic (rather than a line)
   evaluate \( \dot{x} \) at 2 more places
   (quadratic through \( x(0), \dot{x}(0), x(t) \))

2. Could try to get a better estimate of \( \dot{x} \)
   (since it was only using \( \dot{x}_0 \))

If you have \( N \) evals, what is the "best" thing to do

Runge-Kutta Methods fit \( N-1 \) degree polynomials
scheme for any \( N \)

Most popular is RK4

\[
\begin{align*}
\dot{x}_0 &= f(x_0) \\
k_1 &= h f(x_0, t) \\
k_2 &= f(x_0 + \frac{k_1}{2}, t + \frac{h}{2}) \\
k_3 &= f(x_0 + \frac{k_2}{2}, t + \frac{h}{2}) \\
k_4 &= f(x_0 + k_3, t + h) \\
x_1 &= x_0 + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6}
\end{align*}
\]

4 evals / step

Each step is independent
Adaptive Step Size
important — figure out if too much error (which builds up, leads to explosion)
need for speed and robustness
Can’t measure error (usually)
Can see if methods are different
try method 1
try better method (2)
if answers are different, method 1 is clearly no good
reduce step size! (no guarantee 2 is good enough)

RK45 — do RK4 step (not Radau IA, 45-th order!)
do RK5 step
not twice as much would since evals are similar

Question Why RK4 not 4 Euler?
evaluations are usually the expense
for most functions, RK4 does better < is a more realistic model

Why 2x RK4 not RK8?
much higher than RK4, equations get messy
An Alternative 5 Predictor Corrector Methods

0 look at previous times to predict future
advance time to future - see what really happened
2 correct based on real sample

simple pc's suppose we have

\[ x(t-1), x(t) \] to predict \[ x(t+1) \]

use prediction to compute \( x(t+1) \)
use \( x(t+1) \) to compute \( \dot{x}(t+1) \)
use \( \dot{x}(t+1) \) to better compute \( \dot{x}(t+1) \)

Generally no better than RK (effectively equivalent)

Better methods:

0 Richardson extrapolation

as \( h \to 0 \), things get better

think of solution as limit

\( g(n) = \text{solution we get with steps} \)

try \( n = 1, 2, 3, 4, \ldots \)

extrapolate to \( n = \infty \)

idea: step to \( H \) with lots (hundreds) of evals

use modified midpoint \( \to \) extrapolate to infinite steps

try 2, 4, 6, 8, 10 \ldots steps

see how our estimate of the limit improves

stop when it isn't improving

Bulirsch-Stoer Method