Differential Equation Basics

Andrew Witkin

PIXAR Animation Studios
A Canonical Differential Equation

\[ \dot{x} = f(x, t) \]

- \( x(t) \): a moving point.
- \( f(x, t) \): \( x \)'s velocity.
The differential equation

\[ \dot{x} = f(x, t) \]

defines a vector field over \( x \).
Integral Curves

Start Here

Pick any starting point, and follow the vectors.
Given the starting point, follow the integral curve.

\[
\begin{align*}
\dot{x}(t) &= f(x, t) \\
\mathbf{x}(t_0) &= \mathbf{x}_0
\end{align*}
\Rightarrow \mathbf{x}(t), t \geq t_0
Closed Form Solutions

Some simpler IVPs have closed form solutions

\[
\begin{align*}
\dot{x}(t) &= -kx(t) \\
x(t_0) &= x_0
\end{align*}
\]

\[
x(t) = x_0 e^{-k(t-t_0)}, \quad t \geq t_0
\]
Numerical Solutions
Numerical Solutions
\[ \Delta x = \Delta t f(x_0, t_0) \]
Numerical Solutions

\( x_0, t_0 \)

\( x_1, t_1 \)

\( f(x_1, t_1) \)
Numerical Solutions

\[ \triangle t \mathbf{f}(\mathbf{x}, t_1) \]
Numerical Solutions
Numerical Solutions
Numerical Solutions
Numerical Solutions
Numerical Solutions
Euler’s Method

- Simplest numerical solution method
- Discrete time steps
- Bigger steps, bigger errors.

\[ x(t + \Delta t) = x(t) + \Delta t f(x, t) \]
Efficiency

cost = \frac{\text{cost}}{\text{step}} \times \# \text{ steps}
Problem I: Inaccuracy

Error turns $x(t)$ from a circle into the spiral of your choice.
The Midpoint Method

a. Compute an Euler step
\[ \Delta x = \Delta t f(x, t) \]

b. Evaluate \( f \) at the midpoint
\[ f_{\text{mid}} = f \left( x + \frac{\Delta x}{2}, t + \frac{\Delta t}{2} \right) \]

c. Take a step using the midpoint value
\[ x(t + \Delta t) = x(t) + \Delta t f_{\text{mid}} \]
Adaptive Time Stepping

smaller step  larger step
More methods…

• Euler’s method is 1st Order.

• The midpoint method is 2nd Order.

• Just the tip of the iceberg. See *Numerical Recipes* for more.

• Helpful hints:
  – *Don’t* use Euler’s method (you will anyway.)
  – *Do* use adaptive step size.
Problem II: Instability
to Neptune!
As unresolved surface features accumulate, they can cause instability.
Convergence

consistency + stability

see also: Dahlquist equivalence theorem, and Lax equivalence theorem
Modular Implementation

- **Generic operations:**
  - Get dim(x)
  - Get/set x and t
  - Deriv Eval at current (x,t)

- **Write solvers in terms of these.**
  - Re-usable solver code.
  - Simplifies model implementation.
Solver Interface

System

Solver

Dim(state)

Get/Set State

Deriv Eval
void eulerStep(Sys sys, float dt)
{
    float t;
    vector<float> x, dx;

    t = getTime(sys);
    x = getState(sys);
    dx = dt * derivEval(sys, x, t);
    setState(sys, x + dx, t + dt);
}