Simulation: How-to

Course web page:

http://goo.gl/EB3aA

Outline

- Spring systems
- Ordinary differential equations & difference equations
- HW #2

Particle Update

- Given particle state at time t consisting of position, velocity, etc., how do we compute new values at time $\mathbf{x}(t + \Delta t)$?
- Typically, we don't have an explicit parametric function $\mathbf{x}(t)$ that we can just evaluate for any t
  - E.g., a spline curve
- Rather, we have a **set of forces** and an initial value for the particle state
- We have to simulate the action of the forces on the particle to "see what happens"!

Forces

- **Unary:** "Global" forces applied to particles independently
  - Gravity: Can regard as constant acceleration in downward direction:

$$\mathbf{f}_{gravity}(t) = m\,\mathbf{g}$$

  - Drag: Resistance to motion through medium proportional to speed:

$$\mathbf{f}_{drag}(t) = -k_d\,\mathbf{v}(t)$$

  - Just need to sum component forces acting on particle at time t to get net force **f**(t). For example, for a "cannonball" shot through the air:

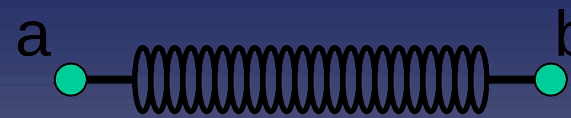
$$\mathbf{f}(t) = \mathbf{f}_{gravity}(t) + \mathbf{f}_{drag}(t) + \ldots$$

- **n-ary:** Interaction forces between particles
  - Gravitational attraction
  - Electrical charge
  - Springs

Gravitational attraction, Electrical charge } Based on proximity

Springs } Specific to connected particles

n-ary Forces: Springs



- 2 connected particles *a* and *b* exert force on one another proportional to displacement from **resting length** r of spring
- Assuming time t, let $\Delta\mathbf{x} = \mathbf{x}_a - \mathbf{x}_b$, $\mathbf{d} = \Delta\mathbf{x}/|\Delta\mathbf{x}|$, and $\Delta\mathbf{v} = \mathbf{v}_a - \mathbf{v}_b$. Then the force on *a* is (where $\mathbf{f}_b = -\mathbf{f}_a$):

$$\mathbf{f}_a = -\left[k_s(||\Delta\mathbf{x}|| - r) + k_d \Delta\mathbf{v} \cdot \mathbf{d}\right] \mathbf{d}$$

spring constant ("stiffness") — damping constant (like "spring drag")

See molecule examples at http://www.myphysicslab.com

Particle Update: Passive Dynamics

- Only exterior forces
  - No motors/muscles
- At each moment in time t, a given particle has:
  - Position **x**(t)
  - Velocity **v**(t)
  - Mass m
  - Net force **f**(t) acting on it according to **Newton's 2nd law of motion** F = m a. Rewrite acceleration as:

$$\mathbf{a}(t) = \dot{\mathbf{v}}(t) = \ddot{\mathbf{x}}(t) = \mathbf{f}(t)/m$$

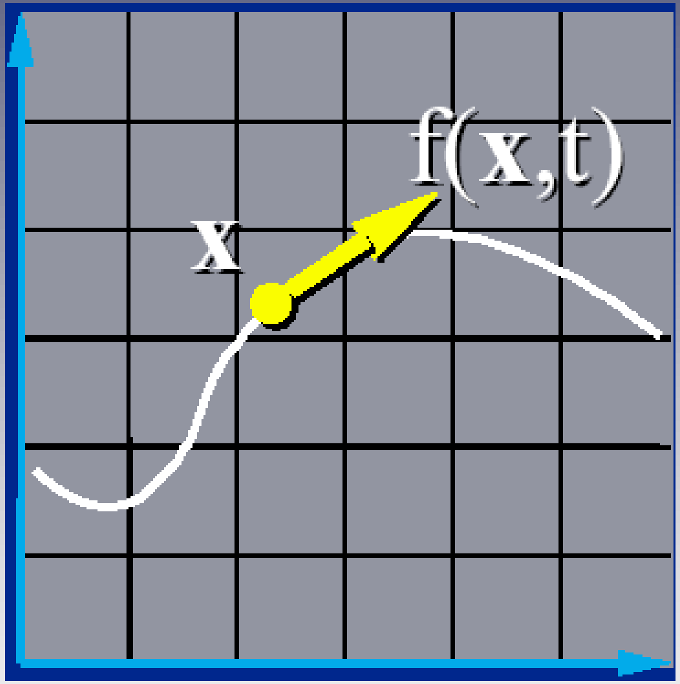
University of Delaware

Passive Update: Steps

- Particle is initialized when created with $\mathbf{x}(t) = \mathbf{x}_0$ and $\mathbf{v}(t) = \mathbf{v}_0$
- So assume we have $\mathbf{x}(t)$ and $\mathbf{v}(t)$
- Now we want:
  1. $\mathbf{v}(t + \Delta t)$: Integrate acceleration $\mathbf{a}(t)$
  2. $\mathbf{x}(t + \Delta t)$: Integrate velocity $\mathbf{v}(t)$

Ordinary Differential Equations

- Consider points on **unknown** parametric curve **x**(t) with **known** derivatives (i.e., tangents) f(**x**, t)



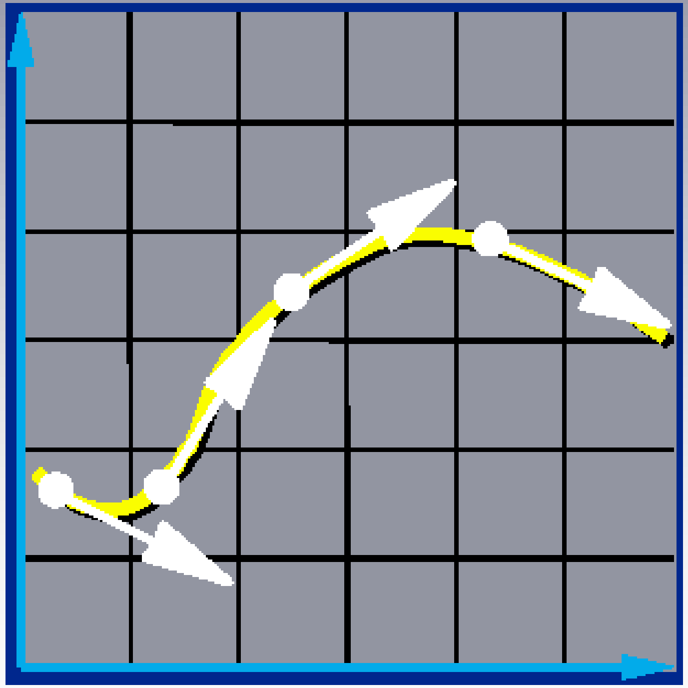
from A. Witkin's SIGGRAPH course notes

$$\frac{\partial \mathbf{x}(t)}{\partial t} = \dot{\mathbf{x}}(t) = f(\mathbf{x}, t)$$

different ways of writing derivative

ODE: Initial Value Problems

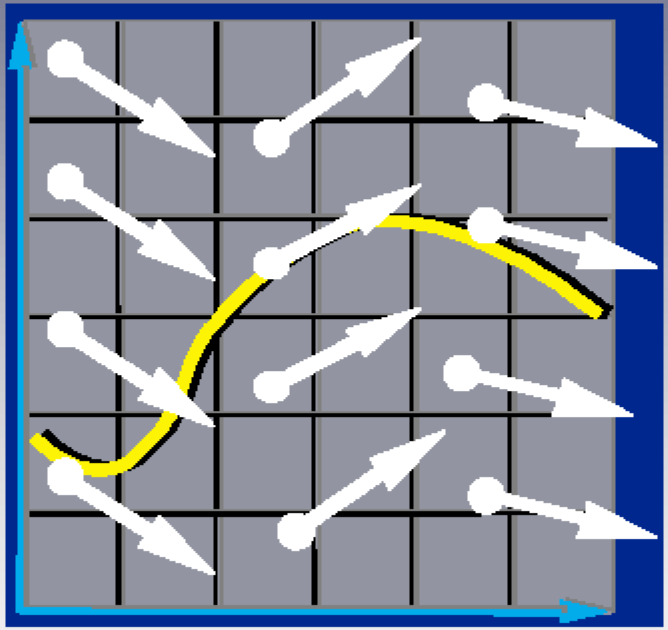
- Suppose we do know the function value at some point
  $\mathbf{x}(t_0) = \mathbf{x}_0$
- How do we compute other values $\mathbf{x}(t)$ where $t \neq t_0$?



from A. Witkin's SIGGRAPH course notes

ODE: Vector field

- Derivative f(**x**, t) is defined for all **x**, so it defines a **vector field**
- Think of this vector field as "pushing" point along—we can choose where to drop the point, but the vector field carries it from there
- Represents **velocities** of point

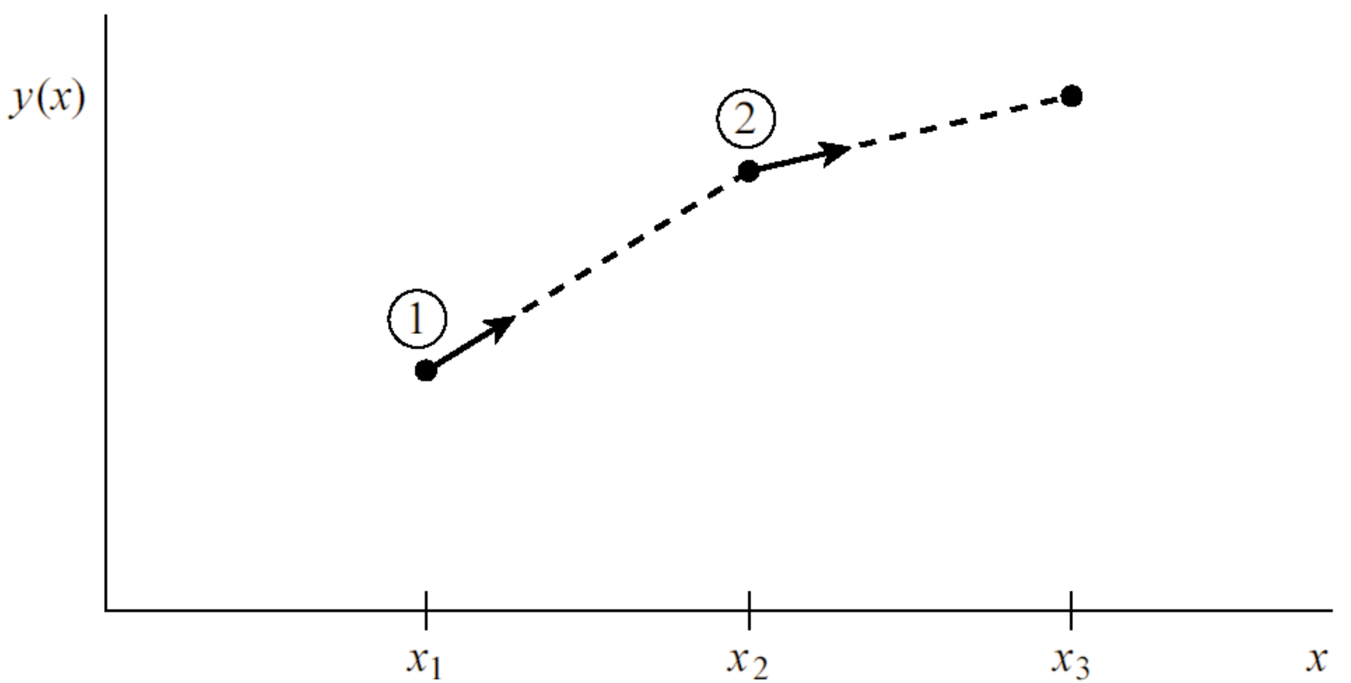


from A. Witkin's SIGGRAPH course notes

Euler Integration

- First order (linear) approximation using a **step size** of Δt :

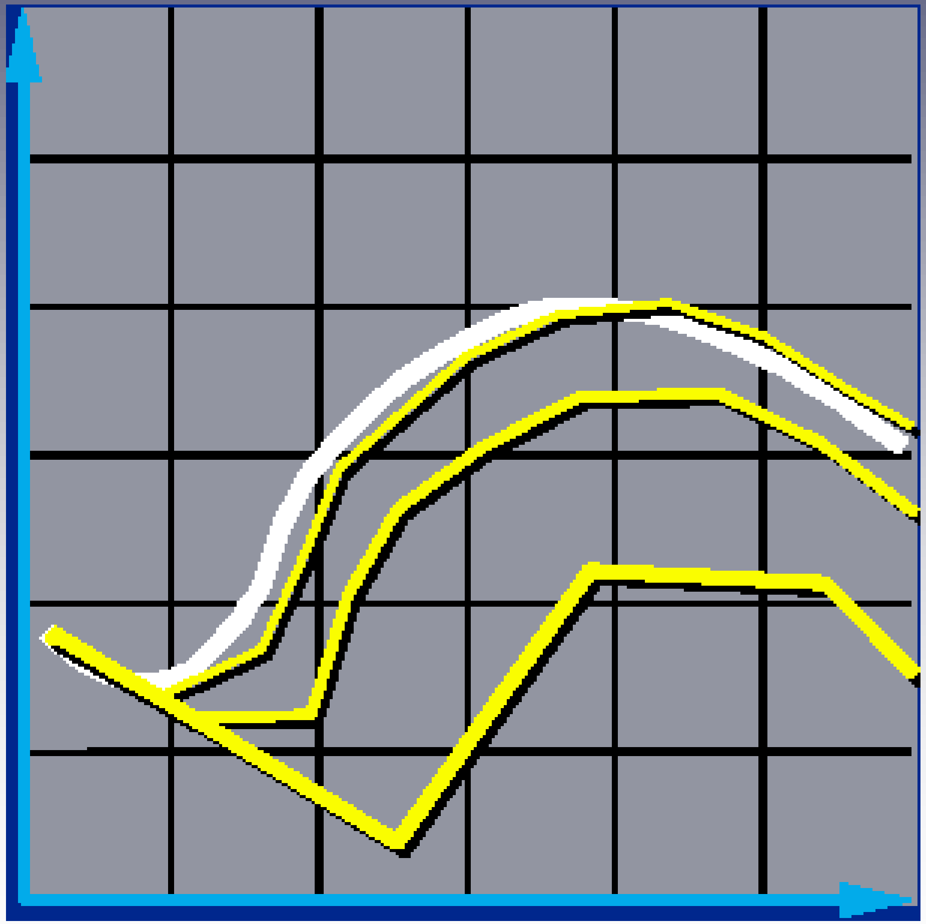
$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t f(\mathbf{x}, t)$$



from Numerical Recipes

Euler Integration: Step Sizes

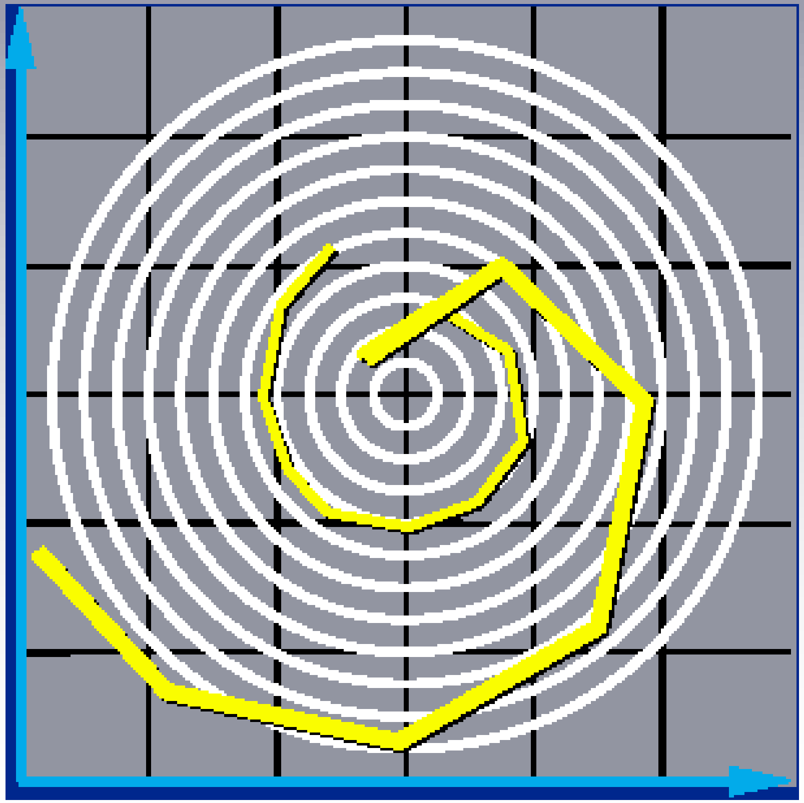
- Piecewise linear approximation of curve
- Shorter step sizes better, but more evaluations :(



from A. Witkin's SIGGRAPH course notes

Euler integration: Issues

- Small accuracy errors at each step can accumulate
- Alternative methods (e.g., Runge-Kutta) can mitigate this



from A. Witkin's SIGGRAPH course notes

Higher-order solvers

- More accurate alternatives to Euler method
- Look at Taylor series expansion:

$$
\begin{aligned}
\mathbf{x}(t_0 + \Delta t) = \mathbf{x}(t_0) + \Delta t \frac{\partial}{\partial t}\mathbf{x}(t_0) \\
+ \frac{\Delta t^2}{2!}\frac{\partial^2}{\partial t^2}\mathbf{x}(t_0) + \ldots + \frac{\Delta t^n}{n!}\frac{\partial^n}{\partial t^n}\mathbf{x}(t_0)
\end{aligned}
$$

- Euler method uses just first two terms—the rest is error

University of Delaware

Midpoint method (aka "2nd order Runge-Kutta", aka RK2)

- Include third term in Taylor series for more accuracy $\frac{\Delta t^2}{2}\frac{\partial^2}{\partial t^2}\mathbf{x}(t_0)$
- Letting f(**x**, t)=f(**x**(t)) for simplicity, compute second derivative with its own Taylor series approximation and substitute to get:

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \Delta t f\left(\mathbf{x}(t) + \frac{\Delta t}{2} f(\mathbf{x}(t))\right)$$

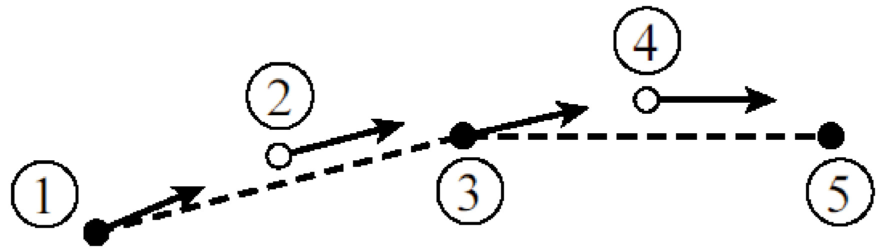
Midpoint/RK2 method: Steps

1. Compute Euler step $\Delta\mathbf{x} = \Delta t\ f(\mathbf{x}, t)$
2. Evaluate first derivative at midpoint (half step)

$$f_{\text{mid}} = f(\mathrm{x}(t) + \frac{\Delta \mathrm{x}}{2}, t + \frac{\Delta t}{2})$$

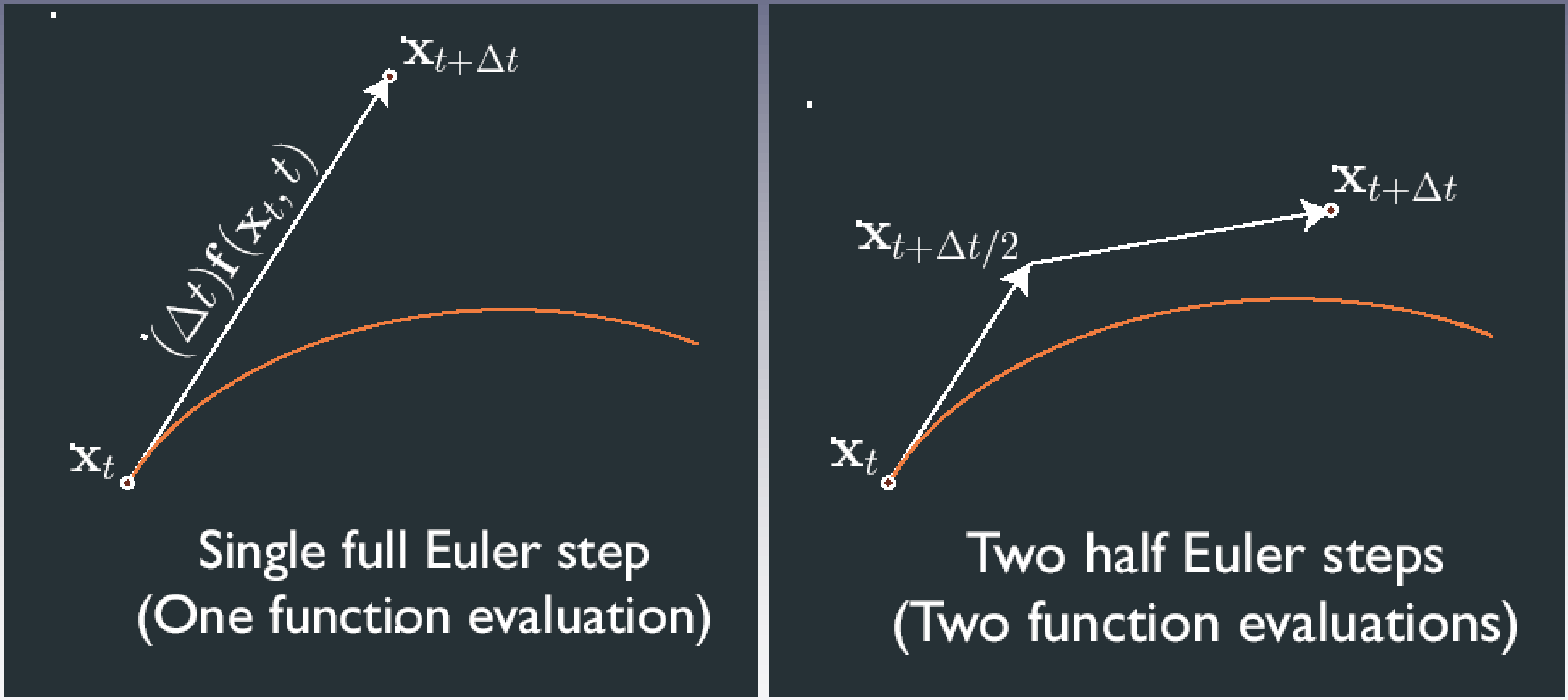
3. Take full step using midpoint derivative

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t\ f_{\text{mid}}$$



from Numerical Recipes

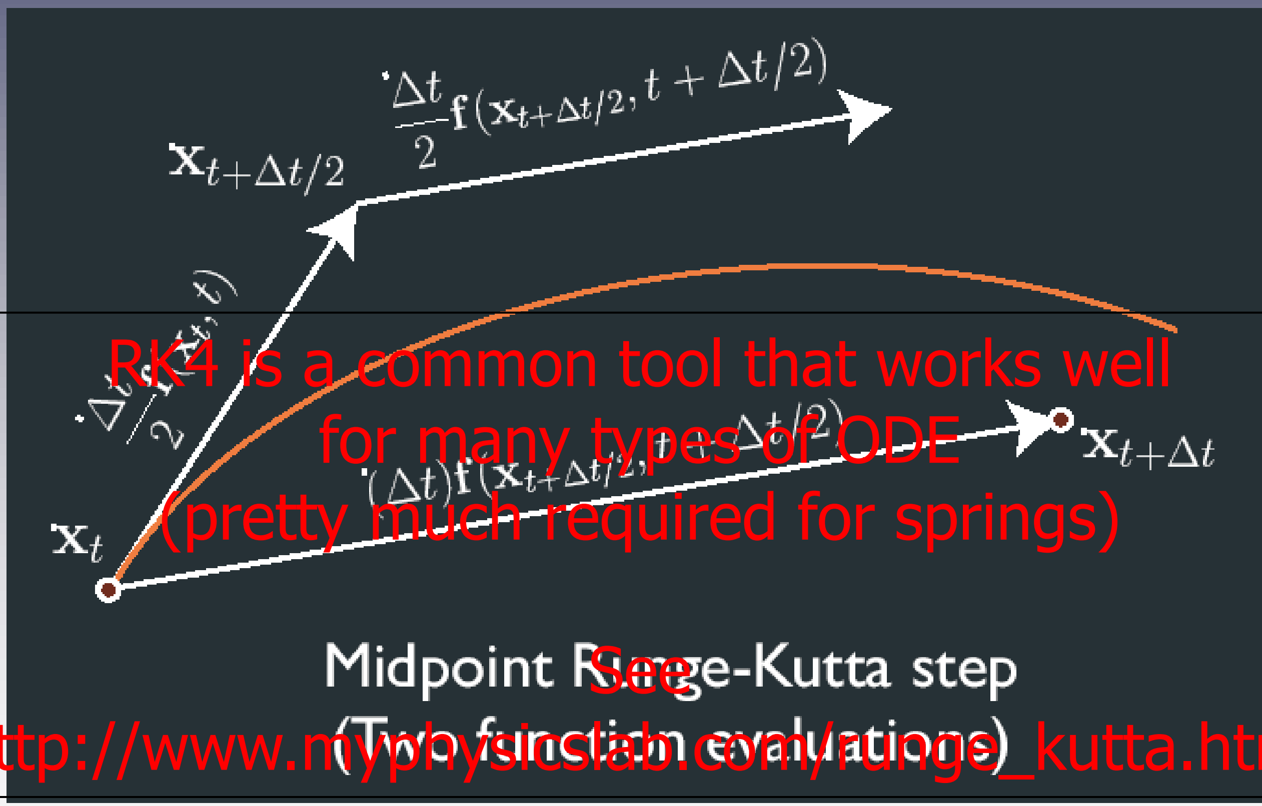
Midpoint/RK2 ODE solver is not the same as two half Euler steps!



courtesy of F. Pfenning

University of Delaware

Midpoint/RK2 ODE solver is not the same as two half Euler steps!



RK4 is a common tool that works well for many types of ODE (pretty much required for springs)

See http://www.myphysicslab.com/runge_kutta.html

HW #2!

- Big idea: 2-D particle system with physics simulation
  - Simulate a waterfall, fireworks, snowstorm, smoke, tornado, flocking, or a combination of these (like “Particle Dreams”)
  - Implement a mass-spring system to simulate collisions of deformable bodies
  - Simulate some variant of pachinko, pinball, Labyrinth, or billiards where a ball or balls collide or roll around. Besides just walls, there might be bumpers, flippers, elevators, conveyor belts, trampolines, cannons, rocket thrusters, parachutes, wind-making fans, etc.
- Provided: Bounce!
- Required elements: see assignment page