Ordinary Differential Equation Algorithms

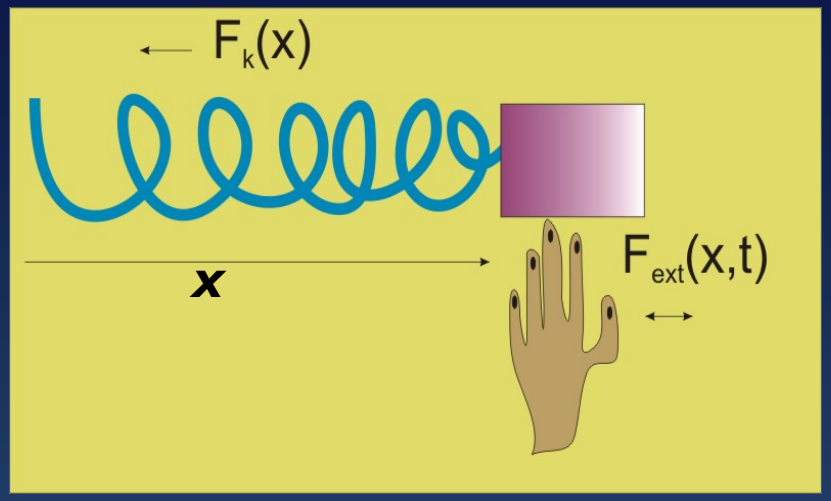
- *Benefit from standard form*
- *Same algorithm, any ODE*
- *Similar to differentiation*

Rubin H Landau
with
Sally Haerer

Computational Physics for Undergraduates
BS Degree Program: Oregon State University

Support by NSF & OSU

Review: Forced Oscillator



$$\frac{d^2x}{dt^2} = \frac{1}{m}F\left(t, \frac{dx}{dt}, x\right) \qquad (1$$

$\Rightarrow$ **Standard form**

$$y^{(0)}(t) \stackrel{\text{def}}{=} x(t) \qquad (3$$

2) $$\frac{d\vec{y}(t)}{dt} = \vec{f}(t, \vec{y})$$

$$\mathbf{y}^{(1)}(t) \stackrel{\text{def}}{=} \frac{dx}{dt} \equiv \frac{dy^{(0)}}{dt} \qquad (4$$

---

5) $$\frac{dy^{(0)}(t)}{dt} = y^{(1)}(t)$$

$$f^{(0)} = y^{(1)}(t) \qquad (7$$

6) $$\frac{dy^{(1)}(t)}{dt} = \frac{1}{m}F(t, y^{(0)}, y^{(1)})$$

$$f^{(1)} = F(t, y^{(0)}, y^{(1)}) \qquad (8$$

Differential Equation Algorithms

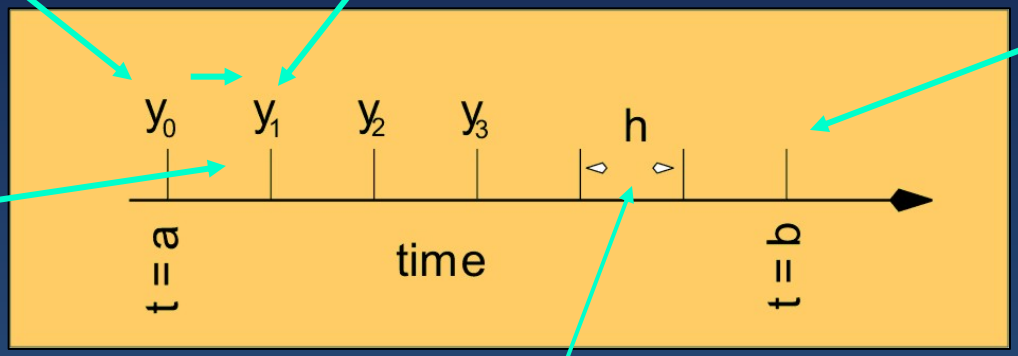
"time" stepping = (integration)

= leapfrog =

initial solution

$y_0 \equiv y(t=0)$

$y_1 \equiv y(t=1\ h)$



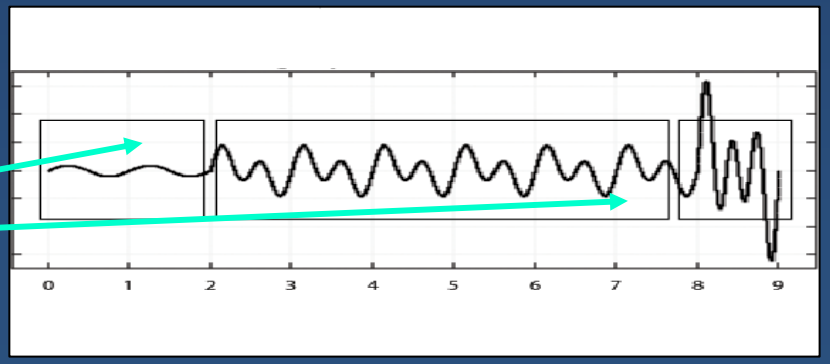
Quit

derivative
$f(t=0, y_0)$

$[\ y_1 \neq y^{(1)}\ ]$

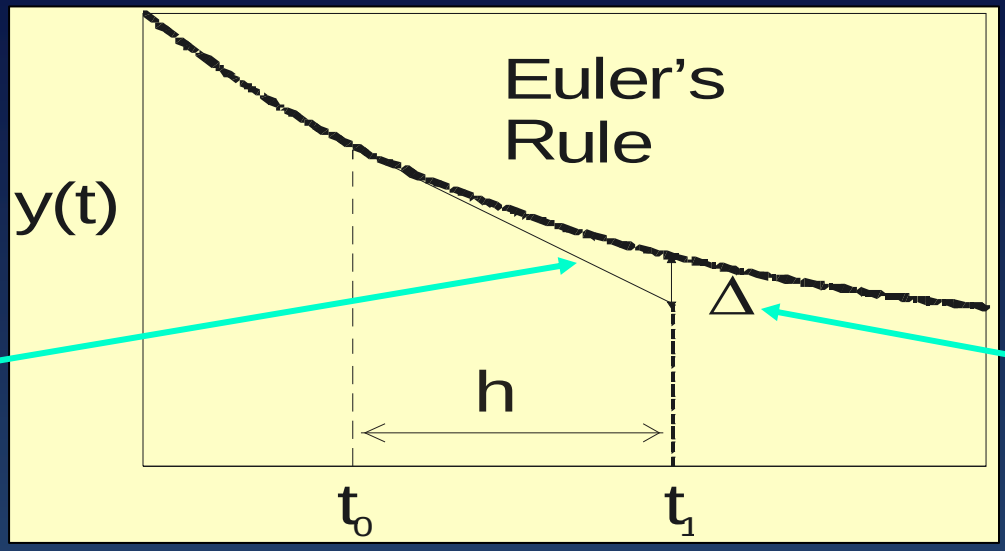
Accuracy $\Rightarrow$ small h $\Rightarrow$ round-off error

Industrial Stength
Adaptive step
Size



h = ?

Too Simple Algorithm: Euler's



Linear tangent line (extrapolate Ø)

Error

1) $$\frac{d\mathbf{y}(t)}{dt} = \mathbf{f}(t, \mathbf{y}),$$

2) $$\frac{\mathbf{y}(t_{n+1}) - \mathbf{y}(t_n)}{h} \simeq \mathbf{f}(t_n, \mathbf{y})$$

3) $$\Rightarrow \quad \mathbf{y}_{n+1} \simeq \mathbf{y}_n + h\mathbf{f}(t_n, \mathbf{y}_n)$$

Order 1: Error ε = O(h^2)

Not as good as

1st Year Physics (t=h)

$$x = x_0 + v_0 h + \tfrac{1}{2} a h^2 \quad (4$$

$$v = v_0 + ah \quad (5$$

Yet is self-starting!

Better Algorithm: Runge-Kutta

- **Still one step, higher order, *rk2*: simple, use *rk4***
- **Industrial standard: *rk4*, *rk45*; ≈ *interpolate***

1) $\frac{dy}{dt} = f(t,y)$

2) $y(t) = \int f(t,y)dt$

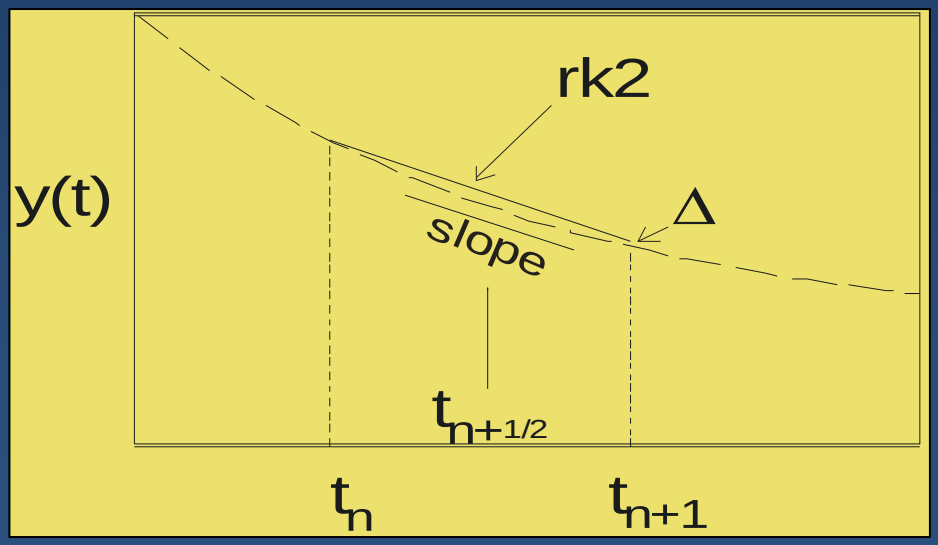
3) $y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f(t,y)dt$

4) $n+ \overset{\text{def}}{=} n+1/2$

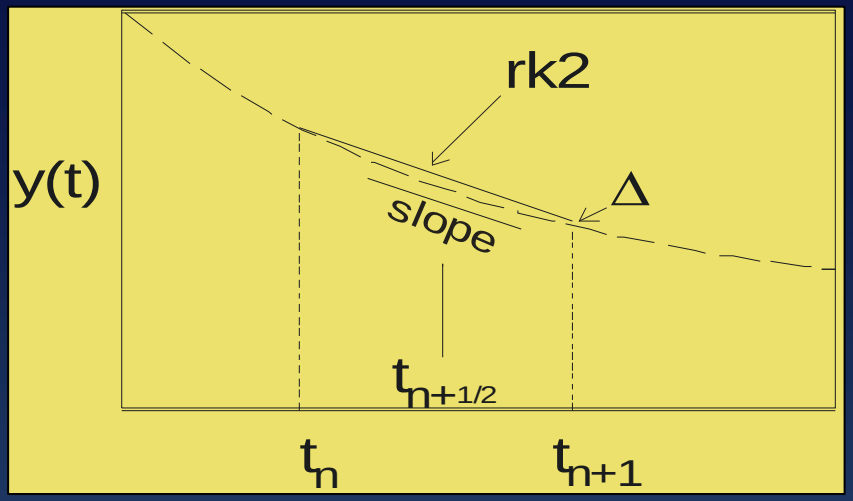
5) $f(t,y) \simeq f(t_{n+}, y_{n+}) + (t - t_{n+})\frac{df}{dt}(t_{n+}) + O(h^2)$

$$\int_{t_n}^{t_{n+1}} f(t,y)dt \simeq f(t_{n+}\, y_{n+})h, \qquad (7$$

$$\Rightarrow \boxed{y_{n+1} \simeq y_n + hf(t_{n+},\, y_{n+})} + \mathcal{O}(h^3) \qquad (8$$



E.G.: Apply rk2



$$y_{n+1} \simeq y_n + h f(t_{n+1/2},\, y_{n+1/2}) \qquad (1$$

- **Higher order, 1 method call**
- **Intermediate $y_{n+½}$ =?**
- **Use Euler's algorithm**

2) $$y_{n+1/2} \simeq y_n + \frac{dy}{dt}\frac{h}{2}$$

3) $$= y_n + \tfrac{1}{2} h f(t_n,\, y_n)$$

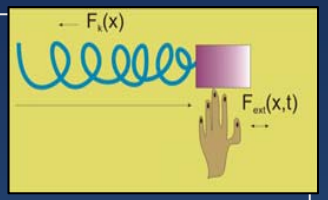
** Putting it all together **

4) $$\vec{y}_{n+1} \simeq \vec{y}_n + \vec{k}_2$$

5) $$\vec{k}_2 = h\vec{f}(t_n + \frac{h}{2},\, \vec{y}_n + \frac{\vec{k}_1}{2}),$$

6) $$\vec{k}_1 = h\,\vec{f}(t_n,\, \vec{y}_n)$$

e.g. spring



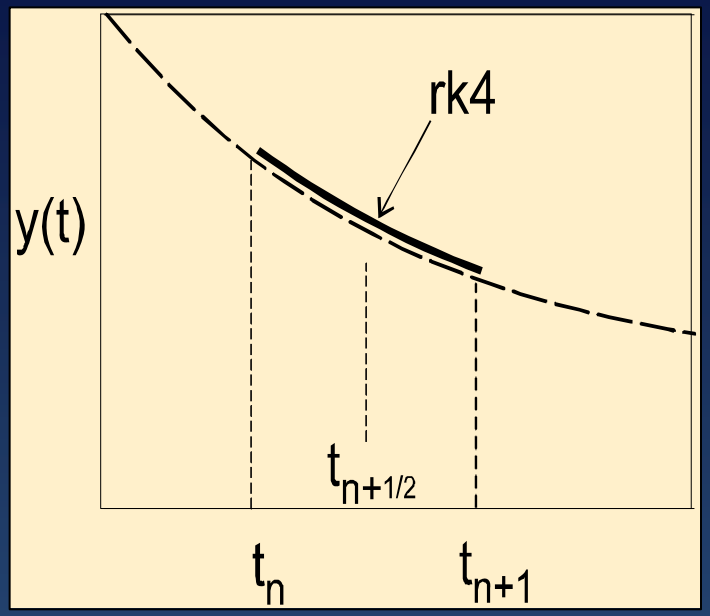
$$y_1^{(0)} = y_0^{(0)} + h f^{(0)}(\frac{h}{2},\, y_0^{(0)} + k_1)$$

$$\simeq x_0 + h[v_0 + \frac{h}{2} F_k(0)] \qquad (7$$

$$y_1^{(1)} = y_0^{(1)} + h f^{(1)}[\frac{h}{2}\, y_0 + \frac{h}{2} f(0,\, y_0)]$$

$$\simeq v_0 + h\frac{1}{m} F_k(y^{(1)}(0) + \frac{k_1}{2}) \qquad (8$$

Rk4: Fourth-Order Runge-Kutta



- Power, precision, simple
- Expand **f (t, y)** ***O*(h^3)** about midpoint
- Better ***f*** **($t_{n+½}$, $y_{n+½}$)**
  - 4 method calls
- rk2: *O*(h^3) error; rk4: *O*(h^5) error

$$\vec{y}_{n+1} = \vec{y}_n + \tfrac{1}{6}(\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4) \qquad (1$$

$$\vec{k}_1 = h\vec{f}(t_n, \vec{y}_n), \qquad \vec{k}_2 = h\vec{f}(t_n + \frac{h}{2}, \vec{y}_n + \frac{\vec{k}_1}{2}) \qquad (2$$

$$\vec{k}_3 = h\vec{f}(t_n + \frac{h}{2}\,\vec{y}_n + \frac{\vec{k}_2}{2}), \qquad \vec{k}_4 = h\vec{f}(t_n + h, \vec{y}_n + \vec{k}_3) \qquad (3$$

O(h) ***O(h^3)***

You deserve a break now!