Computational Methods
CMSC/AMSC/MAPL 460

Ordinary differential equations

Ramani Duraiswami,
Dept. of Computer Science

Several slides adapted from Profs. Dianne O'Leary and Eric Sandt, TAMU

Ordinary differential equations

- Mathematical modeling involves posing models and then solving these models numerically or analytically
- ODEs represent a powerful method of modeling
  - Especially if things depend on rates of change
- Rate of change of distance is velocity

$$v = dx/dt \qquad x(t) = \int_0^t \frac{dx}{d\tau} d\tau$$

- Knowing the velocity as a function of τ we can integrate using numerical quadrature
- Rate of change of velocity is acceleration

$$a = dv/dt = d^2x/dt^2$$

- Given initial conditions ($v(0)=0$, $x(0)=0$) find the location $x(t)$ at time t given that the object falls with a constant acceleration of 10 m/s^2

Solution by simple integration

$$\int dv = \int 10 dt$$

$$v = 10\,t + c_1$$

$$\int dx/dt\, dt = \int 10t\, dt$$

$$x = 10t^2/2 + c_1 t + c_2$$

- Use initial conditions
- $v(0)=0$ so c_1 is zero
- $x(0)=0$ so c_2 is zero
- Final solution $x=5t^2$
- Could handle more complex functions of t under the integral $\frac{dx}{dt} = f(t)$ $\qquad x(t) = \int_0^t f(\tau) d\tau$

What if simple integration would not work?

- Example: Let the velocity be a function of x and t
- $dx/dt=f(x,t)$ $$x(t) = \int_0^t f(x(\tau), \tau) d\tau$$
- Cannot be simply integrated
- This is the typical type of problem we need to solve in ODEs
- This is nonlinear because solution x depends on itself
- The linear case could be solved using numerical quadrature

Writing a 2nd order system in standard form

- ODEs in standard form: way they are input to software
- Written as a set of 1st order equations with initial conditions

$$u'' = g(t, u, u')$$

$$u(0) = u_0$$

$$u'(0) = v_0$$

- where u_0 and v_0 are given.
- Let $y_1 = u$ and $y_2 = u'$. Then, in standard form:

$$y_2' = g(t, y_1, y_2)$$

$$y_1' = y_2$$

$$y_1(0) = u_0 \quad y_2(0) = v_0$$

Standard form

- We'll work with problems in **standard form**,

$$y' = f(t;\, y)$$

$$y(0) = y_0$$

- where the function y has m components,
- y' means the derivative with respect to t, and
- y_0 is a given vector of initial conditions (numbers).
- Writing this component-by-component yields

$$y'_{(1)} = f_1(t,\, y_{(1)} \cdots\, y_{(m)})$$

$$\cdots$$

$$y'_{(m)} = f_m(t,\, y_{(1)} \cdots\, y_{(m)})$$

with $y_{(1)}(t_0)$, …, $y_{(m)}(t_0)$ given initial conditions

A modeling exercise: predator prey problems

- Eco-system (island) that contains rabbits and foxes
- Island has plenty of food for rabbits
- Rabbits reproduce like crazy and would fill-up the island
- Foxes eat rabbits
- Let $r(t)$ represent the number of rabbits and $f(t)$ the number of foxes.
- Model the number of rabbits and foxes on the island and decide if it will reach an equilibrium

Rabbit and fox population

- Rabbit population will grow at a certain rate
  - a is the natural growth rate of rabbits in the absence of predation,
- Rabbits will die as they are eaten by foxes. Let the rabbit die if it encounters a fox.
  - b is the death rate per encounter of rabbits due to predation,
- Fox population dies off if they cannot eat rabbits
  - c is the natural death rate of foxes in the absence of food (rabbits),
- Foxes reproduce if they have food
  - e is the efficiency of turning predated rabbits into foxes.
- Initial conditions $R(0)=r_0$ *and* $F(0)=f_0$
- Volterra equations

$$dR/dt = aR - bRF$$
$$dF/dt = ebRF - cF$$

Standard form

- Volterra's model

$$dR/dt = 2R - \alpha RF$$
$$dF/dt = \alpha RF - F$$

- Another example from the book

$$\ddot{u}(t) = -u(t)/r(t)^3$$
$$\ddot{v}(t) = -v(t)/r(t)^3$$

where

$$r(t) = \sqrt{u(t)^2 + v(t)^2}$$

The vector $y(t)$ has four components,

$$y(t) = \begin{bmatrix} u(t) \\ v(t) \\ \dot{u}(t) \\ \dot{v}(t) \end{bmatrix}$$

The differential equation is

$$\dot{y}(t) = \begin{bmatrix} \dot{u}(t) \\ \dot{v}(t) \\ -u(t)/r(t)^3 \\ -v(t)/r(t)^3 \end{bmatrix}$$

```
function ydot = twobody(t,y)
r = sqrt(y(1)^2 + y(2)^2);
ydot = [y(3); y(4); -y(1)/r^3; -y(2)/r^3];
```

Solving differential equations: Euler's method

- As in quadrature: use Taylor series
- Euler's method

$$y(t+h) = y(t) + hy'(t) + h^2/2 \qquad y''(\xi)$$

for some point ξ *in* $[t, \leq t+h]$

- Note that

$$y'(t) = f(t, y(t)).$$

- *March forward*

$$y(h) = y_1 = y(0) + hy'(0)$$

$$= y_0 + hf(t_0, y_0)$$

$$y_{n+1} = y_n + hf_n$$

Example

Consider

$$\frac{dy}{dx} = x + y$$

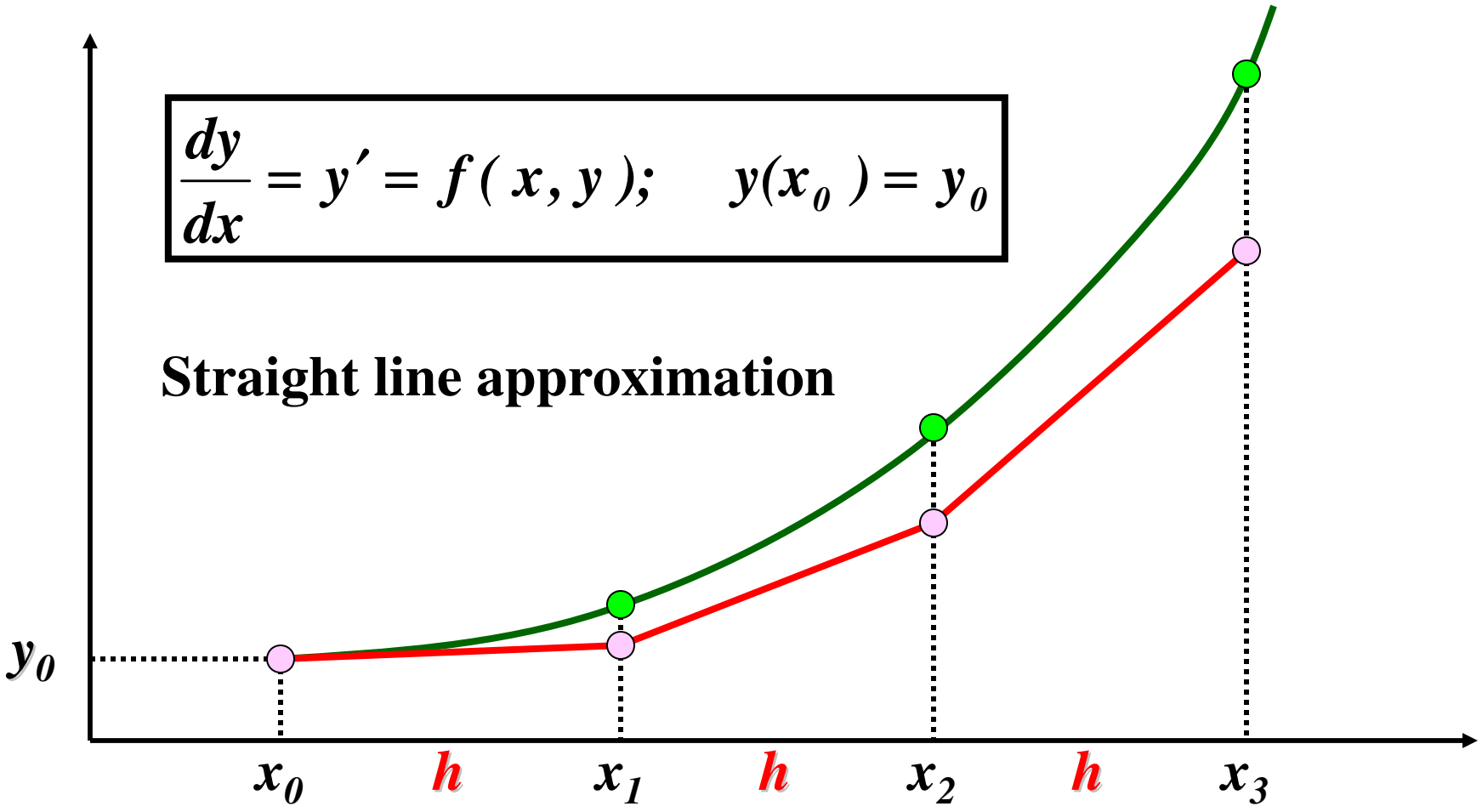
The initial conditions is :

$$y(0) = 1$$

The analytical solution $y(x) = 2e^{x} - x - 1$

Euler's Method: First-order Taylor Method

$$\frac{dy}{dx} = y' = f(x, y); \quad y(x_0) = y_0$$



Euler's Method Example

Consider

$$\frac{dy}{dx} = x + y$$

The initial condition is: $y(0) = 1$

The step size is: $\Delta h = 0.02$

The analytical solution is:

$$y(x) = 2e^{x} - x - 1$$

Euler's Method Example

The algorithm has a loop using the initial conditions and definition of the derivative

The derivative is calculated as:

$$y_i' = x_i + y_i$$

The next y value is calculated:

$$y_{i+1} = y_i + \Delta h \; y_i'$$

Take the next step:

$$x_{i+1} = x_i + \Delta h$$

Euler's Method Example

The results

x_n	y_n	y'_n	hy'_n	Exact Solution	Error
0	1.00000	1.00000	0.02000	1.00000	0.00000
0.02	1.02000	1.04000	0.02080	1.02040	-0.00040
0.04	1.04080	1.08080	0.02162	1.04162	-0.00082
0.06	1.06242	1.12242	0.02245	1.06367	-0.00126
0.08	1.08486	1.16486	0.02330	1.08657	-0.00171
0.1	1.10816	1.20816	0.02416	1.11034	-0.00218
0.12	1.13232	1.25232	0.02505	1.13499	-0.00267
0.14	1.15737	1.29737	0.02595	1.16055	-0.00318
0.16	1.18332	1.34332	0.02687	1.18702	-0.00370
0.18	1.21019	1.39019	0.02780	1.21443	-0.00425
0.2	1.23799	1.43799	0.02876	1.24281	-0.00482

Euler's Method

The trouble with this method is

- Lack of accuracy
- Small step size needed for accuracy

Backward Euler

- We approximated the derivative at the initial point.
- In backward let us approximate it at the final point
- Find y_{n+1} so that

$$y_{n+1} = y_n + h\, f(t_{n+1}; y_{n+1}) :$$

- **Taylor series derivation**

$$y(t) = y(t+h) - h y'(t+h) + \tfrac{1}{2} h^2 y''(\xi)$$

$$y_{n+1} = y_n + h f_{n+1}$$

- How can we use it? Must solve a non-linear equation
- Generally not used in this way, but as a "correction step" in a "predictor-corrector" scheme.

Modified Euler Method

The Modified Euler method uses the slope at both old and the new location and is a predictor-corrector technique.

$$y_{n+1} = y_n + \Delta h\left(\frac{y'_n + y'_{n+1}}{2}\right) + O\left(\Delta h^2\right)$$

The method uses the average slope between the two locations.

Modified Euler Method

The algorithm will be:

$$y'_{n} = f\left(x_{n}, y_{n}\right)$$

$$y^{*}_{n+1} = y_{n} + \Delta h \; y'_{n}$$

Initial guess of the value

$$y'^{*}_{n+1} = f\left(x_{n+1}, y^{*}_{n+1}\right)$$

Updated value

$$y_{n+1} = y_{n} + \Delta h \left(\frac{y'_{n} + y'^{*}_{n+1}}{2} \right)$$

Modified Euler's Method Example

Consider

$$\frac{dy}{dx} = x + y$$

The initial condition is: $y(0) = 1$

The step size is: $\Delta h = 0.02$

The analytical solution is:

$$y(x) = 2e^{x} - x - 1$$

Modified Euler's Method Example

The results are:

				Estimated	Solution	Average	Exact	Error
x_n	y_n	y'_n	hy'_n	y_{n+1}	y'_{n+1}	$h(y'_n+y'_{n+1})/2$	Solution	
0	1.00000	1.00000	0.02000	1.02000	1.04000	0.02040	1.00000	0.00000
0.02	1.02040	1.04040	0.02081	1.04121	1.08121	0.02122	1.02040	0.00000
0.04	1.04162	1.08162	0.02163	1.06325	1.12325	0.02205	1.04162	-0.00001
0.06	1.06366	1.12366	0.02247	1.08614	1.16614	0.02290	1.06367	-0.00001
0.08	1.08656	1.16656	0.02333	1.10989	1.20989	0.02376	1.08657	-0.00001
0.1	1.11033	1.21033	0.02421	1.13453	1.25453	0.02465	1.11034	-0.00001
0.12	1.13498	1.25498	0.02510	1.16008	1.30008	0.02555	1.13499	-0.00002
0.14	1.16053	1.30053	0.02601	1.18654	1.34654	0.02647	1.16055	-0.00002
0.16	1.18700	1.34700	0.02694	1.21394	1.39394	0.02741	1.18702	-0.00002
0.18	1.21441	1.39441	0.02789	1.24229	1.44229	0.02837	1.21443	-0.00003
0.2	1.24277	1.44277	0.02886	1.27163	1.49163	0.02934	1.24281	-0.00003

MEM: Improves order of the method

If we were to look at the Taylor series expansion

$$y'_{n+1} = y_n + h\, y'_n + \frac{1}{2} h^2 y''_n + O\left(h^3\right)$$

Use a forward difference to represent the 2nd derivative

$$y'_{n+1} = y_n + h\, y'_n + \frac{1}{2} h^2 \left(\frac{y'_{n+1} - y'_n}{h} + O(h) \right) + O\left(h^3\right)$$

$$= y_n + h\, y'_n + \frac{1}{2} h y'_{n+1} - \frac{1}{2} h y'_n + O\left(h^3\right)$$

$$= y_n + h \left(\frac{y'_{n+1} + y'_n}{2} \right) + O\left(h^3\right)$$

Predictor Corrector methods

- **P** (predict): Guess y_{n+1} (e.g., using Euler's method).
- **E** (evaluate): Evaluate $f_{n+1} = f(t_{n+1}; y_{n+1})$.
- **C** (correct): Plug the current guess in, to get a new guess:

$$y_{n+1} = y_n + h_n f_{n+1} :$$

- **E**: Evaluate

$$f_{n+1} = f(t_{n+1}; y_{n+1}).$$

- Repeat the CE steps if necessary.
- We call this a PECE (or PE(CE)^k) scheme.