

Computational Methods
CMSC/AMSC/MAPL 460

Ordinary differential equations

Ramani Duraiswami,
Dept. of Computer Science

Several slides adapted from Prof. ERIC SANDT, TAMU

ODE: Previous class

- Applications and examples
- Standard form
- Examples of converting equations to standard form
 - Volterra equation
- Euler Method (an explicit method)
- Backward Euler Method (an implicit/nonlinear method)
- Predictor corrector methods

Today

- Runge Kutta methods
- Matlab function RK45
- Solve Volterra equation
- Multistep methods: Adams Bashforth
- Implicit methods: Adams Moulton

Runge-Kutta Methods

- General class of methods that use evaluations at intermediate points to achieve high order
- Derivation of the 2nd order RK method
- Look for a formula of the type

$$y_{n+1} = y_n + ak_1 + bk_2$$

$$k_1 = hf(x_n, y_n)$$

$$k_2 = hf(x_n + \alpha\Delta h, y_n + \beta k_1)$$

- Specification of α , β , a , and b provides the formula

Runge-Kutta Methods

The initial conditions are:

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

To derive method we use the Taylor series expansion including 2nd order terms

$$y(x_{n+1}) = y(x_n) + h \frac{dy(x_n, y_n)}{dx} + \frac{h^2}{2!} \frac{d^2 y(x_n, y_n)}{dx^2}$$

Runge-Kutta Methods

Expand the derivatives:

$$\frac{d^2 y}{dx^2} = \frac{d}{dx} [f(x, y)] = f_x + f_y \frac{dy}{dx} = f_x + f_y f$$

The Taylor series expansion becomes

$$y_{n+1} = y_n + hf + h^2 \left[\frac{1}{2} (f_x + f_y f) \right]$$

Have expressed second derivative in terms of 1st derivatives of f

Runge-Kutta Methods

Compare with formula we want for Runge-Kutta

$$y_{n+1} = y_n + ahf + bhf(x_n + \alpha h, y_n + \beta hf)$$

The definition of the function

$$f(x_n + \alpha h, y_n + \beta hf) = f + \alpha hf_x + \beta hf_y$$

Expand the next step

$$\begin{aligned} y_{n+1} &= y_n + ahf + bh(f + \alpha hf_x + \beta hf_y) \\ &= y_n + [a + b]hf + b\alpha h^2 f_x + b\beta h^2 f f_y \end{aligned}$$

Runge-Kutta Methods

Compare with the Taylor series

$$y_{n+1} = y_n + [a + b]hf + b\alpha h^2 f + b\beta h^2 f f_y$$

$$[a + b] = 1$$

$$\alpha b = \frac{1}{2}$$

$$\beta b = \frac{1}{2}$$

4 Unknowns

Runge-Kutta Methods

The Taylor series coefficients (3 equations/4 unknowns)

$$[a + b] = 1, \quad \alpha b = \frac{1}{2}, \quad \beta b = \frac{1}{2}$$

If you select “a” as

$$a = \frac{2}{3}, \quad b = \frac{1}{3}, \quad \alpha = \frac{3}{2}, \quad \beta = \frac{3}{2}$$

If you select “a” as

$$a = \frac{1}{2}, \quad b = \frac{1}{2}, \quad \alpha = \beta = 1$$

Note: These coefficient would result in a modified Euler or Midpoint Method

Runge-Kutta Method (2nd Order)

Example

Consider

$$\frac{dy}{dx} = -y^2$$

Exact Solution

$$y = \frac{1}{1+x}$$

The initial condition is:

$$y(0) = 1$$

The step size is:

$$\Delta h = 0.1$$

Use the coefficients

$$a = \frac{1}{2} \quad b = \frac{1}{2}, \quad \alpha = \beta = 1$$

Runge-Kutta Method (2nd Order)

Example

The values are

$$k_1 = hf(x_i, y_i)$$

$$k_2 = hf(x_i + h, y_i + k_1)$$

$$y_{i+1} = y_i + \frac{1}{2}[k_1 + k_2]$$

Runge-Kutta Method (2nd Order) Example

- The values are similar to that of the Modified Euler
 - also a second order method

x_n	y_n	y'_n	k_1	Estimate	Solution	k_2	Exact	Error
			hy'_n	y^*_{n+1}	y^*_{n+1}	$h(y^*_{n+1})$		
0	1.00000	-1.00000	-0.10000	0.90000	-0.81000	-0.08100	1.000000	0.000000
0.1	0.90950	-0.82719	-0.08272	0.82678	-0.68357	-0.06836	0.909091	-0.000409
0.2	0.83396	-0.69549	-0.06955	0.76441	-0.58433	-0.05843	0.833333	-0.000629
0.3	0.76997	-0.59286	-0.05929	0.71069	-0.50507	-0.05051	0.769231	-0.000740
0.4	0.71507	-0.51133	-0.05113	0.66394	-0.44082	-0.04408	0.714286	-0.000789
0.5	0.66747	-0.44551	-0.04455	0.62292	-0.38802	-0.03880	0.666667	-0.000801
0.6	0.62579	-0.39161	-0.03916	0.58663	-0.34413	-0.03441	0.625000	-0.000790
0.7	0.58900	-0.34692	-0.03469	0.55431	-0.30726	-0.03073	0.588235	-0.000768
0.8	0.55629	-0.30946	-0.03095	0.52535	-0.27599	-0.02760	0.555556	-0.000738
0.9	0.52702	-0.27775	-0.02778	0.49925	-0.24925	-0.02492	0.526316	-0.000705
1	0.50067	-0.25067	-0.02507	0.47560	-0.22620	-0.02262	0.500000	-0.000671

Runge-Kutta Method (2nd Order) Example [b]

The values are $a = \frac{2}{3}$, $b = \frac{1}{3}$, $\alpha = \frac{3}{2}$, $\beta = \frac{3}{2}$

$$k_1 = hf(x_i, y_i)$$

$$k_2 = hf(x_i + \alpha h, y_i + \beta k_1)$$

$$y_{i+1} = y_i + ak_1 + bk_2$$

Runge-Kutta Method (2nd Order) Example [b]

The values are

			k_1	Estimate	Solution	k_2	Exact	Error
x_n	y_n	y'_n	hy'_n	y^*_{n+1}	y^*_{n+1}	$h(y^*_{n+1})$	Exact	
0	1.00000	-1.00000	-0.10000	0.85000	-0.72250	-0.07225	1.000000	0.000000
0.1	0.90925	-0.82674	-0.08267	0.78524	-0.61660	-0.06166	0.909091	-0.000159
0.2	0.83358	-0.69486	-0.06949	0.72935	-0.53195	-0.05320	0.833333	-0.000248
0.3	0.76953	-0.59217	-0.05922	0.68070	-0.46335	-0.04634	0.769231	-0.000295
0.4	0.71460	-0.51066	-0.05107	0.63800	-0.40705	-0.04070	0.714286	-0.000317
0.5	0.66699	-0.44488	-0.04449	0.60026	-0.36031	-0.03603	0.666667	-0.000324
0.6	0.62532	-0.39103	-0.03910	0.56667	-0.32111	-0.03211	0.625000	-0.000321
0.7	0.58855	-0.34639	-0.03464	0.53659	-0.28793	-0.02879	0.588235	-0.000314
0.8	0.55586	-0.30898	-0.03090	0.50951	-0.25960	-0.02596	0.555556	-0.000303
0.9	0.52661	-0.27731	-0.02773	0.48501	-0.23523	-0.02352	0.526316	-0.000291
1	0.50028	-0.25028	-0.02503	0.46274	-0.21412	-0.02141	0.500000	-0.000278

Runge-Kutta Methods

- Fourth order Runge-Kutta method

$$y_{n+1} = y_n + 1/6(k_1 + 2k_2 + 2k_3 + k_4)$$

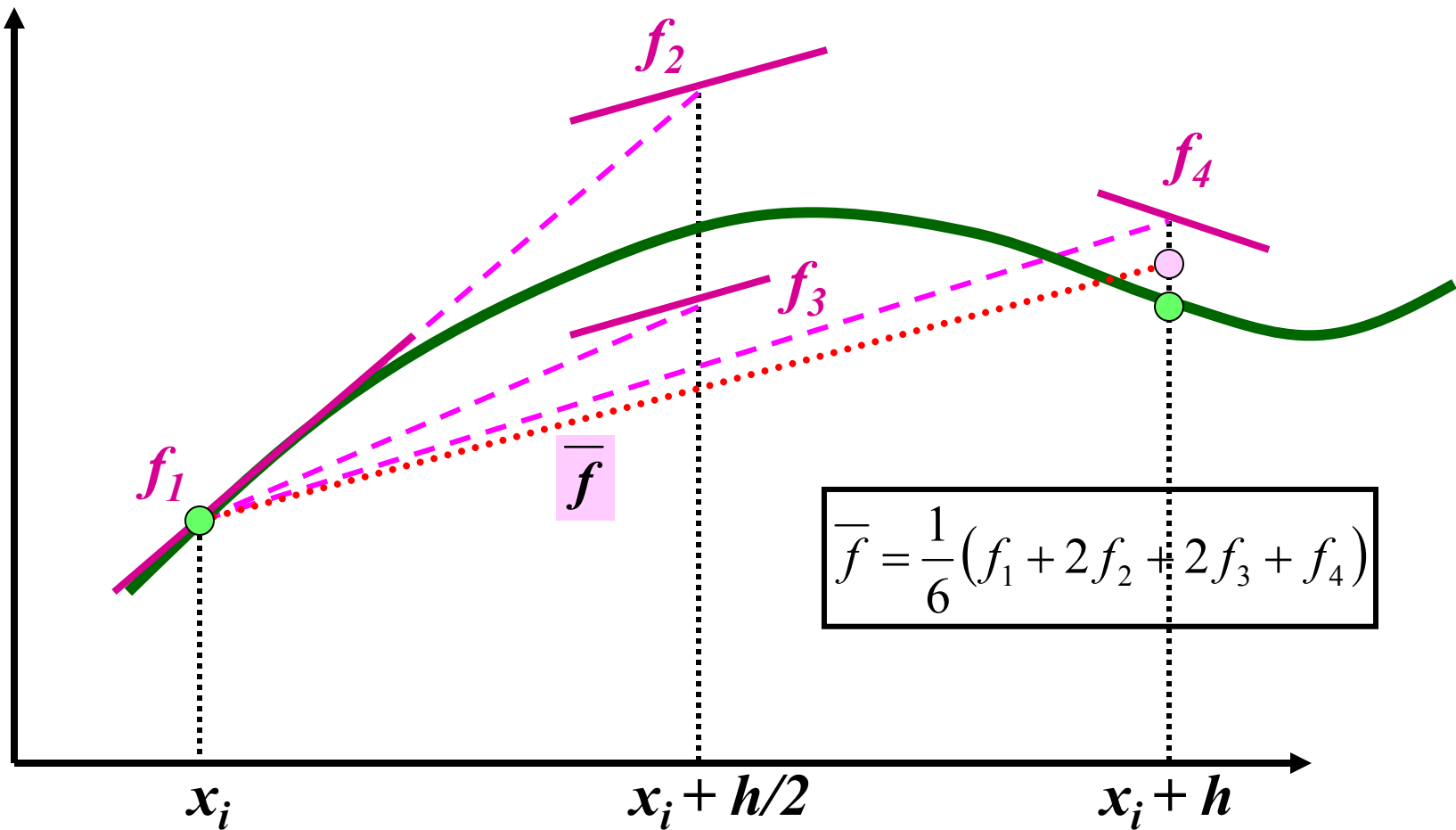
$$k_1 = hf(x, y)$$

$$k_2 = h(f(x+h/2, y+1/2 k_1))$$

$$k_3 = h(f(x+h/2, y+1/2 k_2))$$

$$k_4 = h(f(x+h, y+k_3))$$

4th-order Runge-Kutta Method



Volterra example

- Write a function in standard form

```
function f = rabfox(t,y)
% Computes y' for the Volterra model.
% y(1) is the number of rabbits at time t.
% y(2) is the number of foxes at time t.
global alpha % interaction constant
t % a print statement, just so we can see how
    fast
% the progress is, and what stepsize is being
    used
f(1,1) = 2*y(1) - alpha*y(1)*y(2);
f(2,1) = -y(2) + alpha*y(1)*y(2);
```

Study its solution for various values of encounter

```
% Run the rabbit-fox model for various values of  
% the encounter parameter alpha, plotting each  
% solution.  
global alpha  
for i=2:-1:0,  
alpha = 10^(-i)  
[t,y] = ode45('rabfox',[0:.1:2], [20,10]);  
plot(t,y(:,1),'r',t,y(:,2),'b');  
legend('rabbits','foxes')  
title(sprintf('alpha = %f',alpha));  
pause  
end
```

Runge-Kutta Method (4th Order)

Example

Consider

$$\frac{dy}{dx} = y - x^2$$

Exact Solution

$$y = 2 + 2x + x^2 - e^x$$

The initial condition is:

$$y(0) = 1$$

The step size is:

$$\Delta h = 0.1$$

The 4th Order Runge-Kutta

The example of a single step:

$$k_1 = \Delta h [f(x, y)] = 0.1 f(0, 1) = 0.1(1 - 0^2) = 0.1$$

$$k_2 = \Delta h \left[f \left(x + \frac{1}{2} \Delta h, y + \frac{1}{2} k_1 \right) \right] = 0.1 f(0.05, 1.05) = 0.10475$$

$$k_3 = \Delta h \left[f \left(x + \frac{1}{2} \Delta h, y + \frac{1}{2} k_2 \right) \right] = 0.1 f(0.05, 1. + k_2 / 2) = 0.104988$$

$$k_4 = \Delta h [f(x + \Delta h, y + k_3)] = 0.1 f(0.1, 1.104988) = 0.109499$$

$$y_{n+1} = y_n + \frac{1}{6} [k_1 + 2k_2 + 2k_3 + k_4] = 1.104829$$

Runge-Kutta Method (4th Order)

Example

The values for the 4th order Runge-Kutta method

x	y	f(x,y)	k ₁	f ₂	k ₂	f ₃	k ₃	f ₄	k ₄	Change	Exact
0	1	1	0.1	1.0475	0.10475	1.049875	0.104988	1.094988	0.109499	0.628974	1
0.1	1.104829	1.094829	0.109483	1.13707	0.113707	1.139182	0.113918	1.178747	0.117875	0.682608	1.104829
0.2	1.218597	1.178597	0.11786	1.215027	0.121503	1.216848	0.121685	1.250282	0.125028	0.729263	1.218597
0.3	1.340141	1.250141	0.125014	1.280148	0.128015	1.281648	0.128165	1.308306	0.130831	0.768204	1.340141
0.4	1.468175	1.308175	0.130817	1.331084	0.133108	1.332229	0.133223	1.351398	0.13514	0.79862	1.468175
0.5	1.601278	1.351278	0.135128	1.366342	0.136634	1.367095	0.13671	1.377988	0.137799	0.819614	1.601279
0.6	1.73788	1.37788	0.137788	1.384274	0.138427	1.384594	0.138459	1.38634	0.138634	0.830196	1.737881
0.7	1.876246	1.386246	0.138625	1.383059	0.138306	1.382899	0.13829	1.374536	0.137454	0.82927	1.876247
0.8	2.014458	1.374458	0.137446	1.360681	0.136068	1.359992	0.135999	1.340457	0.134046	0.815626	2.014459
0.9	2.150396	1.340396	0.13404	1.314915	0.131492	1.313641	0.131364	1.28176	0.128176	0.787927	2.150397
1	2.281717	1.281717	0.128172	1.243303	0.12433	1.241382	0.124138	1.195855	0.119586	0.744694	2.281718

The 4th Order Runge-Kutta

The step sizes are:

$$\Delta y = \frac{1}{3} [k_1 + k_2 + k_3]$$

$$\Delta y' = \frac{1}{3\Delta h} [k_1 + 2k_2 + 2k_3 + k_4]$$

The next step would be:

$$y(x + \Delta h) = y(x) + \Delta h y'(x) + \Delta y$$

$$y'(x + \Delta h) = y'(x) + \Delta y'$$

One Step Method

The one-step techniques

- These methods allow us to vary the step size.
- Use only one initial value.
- After each step is completed the past step is “forgotten: We do not use this information.

Explicit and One-Step Methods

Up until this point we have dealt with:

- Euler Method
- Modified Euler/Midpoint
- Runge-Kutta Methods

These methods are called explicit methods, because they use only the information from previous steps.

Moreover these are one-step methods