UNIVERSITI TEKNOLOGI MALAYSIA

SME 3023 Applied Numerical Methods

Ordinary Differential Equations

Abu Hasan Abdullah

Faculty of Mechanical Engineering

Sept 2012

Outline

Outline

1. Introduction
2. Sample Engineering Problems

Outline

1. Introduction
2. Sample Engineering Problems
3. Order of ODE

Outline

1. Introduction
2. Sample Engineering Problems
3. Order of ODE
4. Linearization of ODE

Outline

1. Introduction
2. Sample Engineering Problems
3. Order of ODE
4. Linearization of ODE
5. Non-computer Methods for Solving ODE

Outline

1. Introduction
2. Sample Engineering Problems
3. Order of ODE
4. Linearization of ODE
5. Non-computer Methods for Solving ODE
6. ODE - Initial Value Problem
    - Euler’s Method
    - Heun’s Method
    - Runge-Kutta (RK) Methods
    - Problems of Order n

Outline

Outline

1. Introduction
2. Sample Engineering Problems
3. Order of ODE
4. Linearization of ODE
5. Non-computer Methods for Solving ODE
6. ODE - Initial Value Problem
    - Euler's Method
    - Heun's Method
    - Runge-Kutta (RK) Methods
    - Problems of Order n
7. ODE - Boundary Value Problem
    - Shooting Method
    - Finite Difference Method
8. References

Introduction

- Newton's second law to compute velocity v of a falling mass m as a function of time t can be written as

$$\frac{dv}{dt} = g - \frac{c}{m}v \tag{1}$$

  where g is *gravitational constant*, and c is *drag coefficient*. Eq. (1) is composed of
  - unknown function, v, and
  - its derivatives, dv/dt

  Such equation is called *differential equation* or *rate equation*. Also, in Eq. (1)
  - v is the *dependent variable*
  - t is the *independent variable*
- When function involves *ONE* independent variable, the equation is called the *ordinary differential equation*, e.g.

$$\frac{dv}{dt} = g - \frac{c}{m}v$$

- When function involves *TWO or MORE* independent variables, the equation is called the *partial differential equation*, e.g.

$$\rho\frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} + \rho f_x$$

Sample Engineering Problems

- Newton's second law of motion

$$\frac{dv}{dt} = \frac{F}{m} \tag{2}$$

where v is *velocity*, F is *force* and m is *mass*

- Fourier's heat law

$$q = -k\frac{dT}{dx} \tag{3}$$

where q is *heat flux*, k is *thermal conductivity* and T is *temperature*

- Fick's law of diffusion

$$J = -D\frac{dc}{dx} \tag{4}$$

where J is *mass flux*, D is *diffusion coefficient* and c is *concentration*

- Faraday's law (voltage drop across an inductor)

$$\Delta V_t = L\frac{di}{dt} \tag{5}$$

where ΔV_t is *voltage drop*, L is *inductance* and i is *current*

Order of ODE

- Differential equations are classified as to the order:
  - First order equation—highest derivative is *first* derivative

$$\frac{dv}{dt} = g - \frac{c}{m}v$$

  - Second order equation—highest derivative is *second* derivative, for example equation describing position x of mass-spring system with damping

$$m\frac{d^2x}{dt^2} + c\frac{dx}{dt} + kx = 0 \tag{6}$$

  where c is *gravitational constant* and k is *mass* and c is *drag coefficient*
  - Similarly, n^{th} order differential equation would include n^{th} derivative.
- Higher order differential equation can be reduced to a system of first order equations. For Eq. (6) above, we do it by defining

$$y = \frac{dx}{dt} \tag{7}$$

Order of ODE

- Eq. (7) is differentiated to yield

$$\frac{dy}{dt} = \frac{d^2x}{dt^2} \tag{8}$$

- Eqs. (7) and (8) are then substituted into Eq. (6) to yield

$$m\frac{dy}{dt} + cy + kx = 0 \tag{9}$$

- Eq. (9) can be re-arranged into

$$\frac{dy}{dt} = -\frac{cy + kx}{m} = 0 \tag{10}$$

  Thus Eqs. (7) and (10) are two first order equations that are *equivalent* to the original second order equation, i.e. Eq. (6).

Linearization of ODE

- A linear ODE is one that fit the general form

$$a_n(x)y^{(n)} + \ldots + a_1(x)y' + a_0(x)y = f(x) \tag{11}$$

where $y^{(n)}$ is n^{th} derivative of y w.r.t. x, a is specified function of x and c is specified function of x. Eq. (11) is *linear* because there are NO products or nonlinear function of dependent variable y and its derivatives.

- Motion of a swinging pendulum, Figure 1, is governed by

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\sin\theta = 0 \tag{12}$$

Eq. (12) is *nonlinear* because of the term $\sin\theta$.

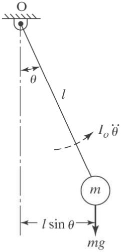

Figure 1 : Motion of swinging pendulum.

Linearization of ODE

- To obtain solution to Eq. (12) we *linearize* it by assuming, for small values of θ,

$$\sin\theta \approx \theta \tag{13}$$

which, on substitution into Eq. (12), yields a linear form

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\theta = 0 \tag{14}$$

- Why linear ODE are important?
  Because they can be solved analytically. In contrast, most nonlinear equations cannot be solved exactly!!

Non-computer Methods for Solving ODE

- Without computers, ODE are solved with *analytical* integration techniques in which Eq. (1) could be multiplied by dt and then integrated to yield

  $$v = \int \left(g - \frac{c}{m} v \right) dt \tag{15}$$

  RHS of Eq. (15) is an *indefinite integral* because limits of integration are not specified
- Eq. (15) can be solved analytically, assuming $v = 0$ at $t = 0$, to yield

  $$v(t) = \frac{gm}{c}\left(1 - e^{-(c/m)t}\right)$$

ODE - Initial Value Problem

Mathematical Background

- A solution of an ODE is a *specific* function of independent variable and parameters that satisfy the original differential equation. Let say we were given a forth order polynomial

  $$y = -0.5x^4 + 4x^3 - 10x^2 + 8.5x + 1 \tag{16}$$

  Differentiating Eq. (16) gives us an ODE

  $$\frac{dy}{dx} = -2x^3 + 12x^2 - 20x + 8.5 \tag{17}$$

  Eq. (17) also describes the behaviour of a polynomial but in a different manner than Eq. (16)—i.e. it gives rate of change of y w.r.t. x, which is a slope.

ODE - Initial Value Problem

Mathematical Background

- But our objective here is to determine the original function (Eq. (16)) given the ODE (Eq. (17)). So we go back to Eq. (17) and solve analytically

  $$y = \int (-2x^3 + 12x^2 - 20x + 8.5)dx$$

  which leads to

  $$y = -0.5x^4 + 4x^3 - 10x^2 + 8.5x + C \tag{18}$$

  Eq. (18) is identical to Eq. (16) with one notable exception—1 in Eq. (16) being replaced by C in Eq. (18) and C is the constant of integration which is arbitrary.
- Therefore, to specify the solution *completely*, an ODE is usually accompanied by auxiliary condition(s)
- For first order ODE, this auxiliary condition, called initial value, is required to determine the *constant of integration* and obtain a *unique* solution.

- In order to get back Eq. (16), Eq. (18) needs the initial value that $x = 0, y = 1$. Substituting the initial value into Eq.(18) yields

$$1 = -0.5(0)^4 + 4(0)^3 - 10(0)^2 + 8.5x + C \tag{19}$$

or

$$C = 1 \tag{20}$$

Hence,

$$y = -0.5x^4 + 4x^3 - 10x^2 + 8.5x + 1$$

ODE - Initial Value Problem

Basis of Numerical Solution

- Basis of numerical method devoted to solving ODE of the form

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

  is

  $$\begin{aligned}\text{New value} &= \text{Old value} \\ &\quad + (\text{Slope} \times \text{Step size})\end{aligned}$$

  which can be mathematically expressed as

  $$y_{i+1} = y_i + \phi h \tag{21}$$

  where ϕ is the slope estimate. This method is known as *one-step* method, Figure 2.

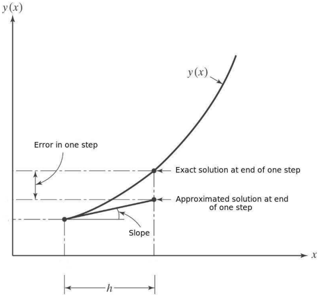

Figure 2 : One-step method.

ODE - Initial Value Problem

Euler's Method

- The first derivative provides a direct estimate of the slope at x_i

  $$\phi = f(x_i, y_i)$$

  where $f(x_i, y_i)$ is *differential equation evaluated at x_i and y_i*. Substituting this into Eq. (21) yields

  $$y_{i+1} = y_i + f(x_i, y_i)h \tag{22}$$

  Eq. (22) represents the Euler's (or Cauchy-Euler) method.

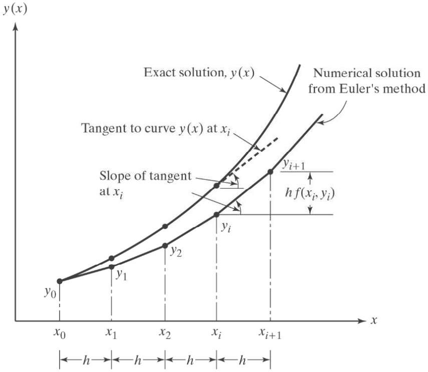

Figure 3 : Euler's method.

Problem Statement:

Use Euler's method to solve

$$\frac{dy}{dx} = -2x^3 + 12x^2 - 20x + 8.5$$

from $x = 0$ to $x = 4$ with step size of 0.5. Initial condition at $x = 0$ is $y = 1$.

Solution:

At $x = 0.5$, Eq. (22), with initial condition $y(0) = 1$, yield

$$\begin{aligned} y(0.5) &= y(0) + f(0,1) \times 0.5 \\ &= 1.0 + [-2(0)^3 + 12(0)^2 - 20(0) + 8.5] \times 0.5 = 5.25 \end{aligned}$$

True solution at $x = 0.5$ is

$$y = -0.5(0.5)^4 + 4(0.5)^3 - 10(0.5)^2 + 8.5(0.5) + 1 = 3.21875$$

$$\therefore E_x = 3.21875 - 5.25 = -2.03125 \quad \text{or} \quad R_x = -63.1\%$$

Solution: continued...

At $x = 1.0$,

$$\begin{aligned} y(1.0) &= y(0.5) + f(0.5, 5.25) \times (0.5) \\ &= 5.25 + \left[-2(0.5)^3 + 12(0.5)^2 - 20(0.5) + 8.5\right] \times 0.5 = 5.875 \end{aligned}$$

True solution at $x = 1.0$ is

$$y = -0.5(1.0)^4 + 4(1.0)^3 - 10(1.0)^2 + 8.5(1.0) + 1 = 3.0$$

$$\therefore E_x = 3.0 - 5.875 = -2.875 \quad \text{or} \quad R_x = -95.8\%$$

Computation is repeated and results tabulated

x	y_{true}	y_{Euler}	Error (%)
0.0	1.00000	1.00000	0.0
0.5	3.21875	5.25000	-63.1
1.0	3.00000	5.87500	-95.8
1.5	2.21875	5.12500	131.0
$\vdots$	$\vdots$	$\vdots$	$\vdots$
4.0	3.00000	7.00000	-133.3

ODE - Initial Value Problem

Example 1: Euler's Method

Solution: continued...

Using Matlab to solve the same involves the following steps:

1. Write the ODE into `dydx.m`
2. Optionally, write the known solution into `ytrue.m`
3. Matlab script to solve the ODE is written in `myeuler.m`

Matlab codes

dydx.m

```
function df=dydx(x)
df = -2*x.^3 + 12*x.^2 - 20*x + 8.5;
```

ytrue.m

```
function y=ytrue(x)
y = -0.5*x.^4 + 4*x.^3 - 10*x.^2 + 8.5*x + 1;
```

ODE - Initial Value Problem

Example 1: Euler's Method

Solution: continued…

`myeuler.m`

```
x0 = 0.0; xn = 4.0; xstep=0.01;
x  = [x0:xstep:xn]; n  = length(x);
xt = linspace(x(1),x(n),100); yt  = ytrue(x);
plot(x,yt,'r'); hold on
y(1) = 1.0;      % This is INITIAL VALUE
for i=2:n
    y(i) = y(i-1) + dydx(x(i-1))*xstep;
end
err = abs((yt-y)./yt)*100;
plot(x,y,'b')
[x' y' yt' err']
```

ODE - Initial Value Problem

Euler's Method - Postscript

- First order technique such as Euler's method demands great computational effort to obtain acceptable error level
- Simplicity of Euler's method makes it attractive, easy to program, suitable for quick analyses.

ODE - Initial Value Problem

Heun's Method

- An attempt to improve Euler's method
- Determines two derivatives for the interval or steps
    - at beginning of interval
    - at end of interval

These two derivatives are then averaged to get an improved estimate of the slope.

- Slope at beginning of interval

$$y_i' = f(x_i, y_i) \tag{23}$$

is used to extrapolate linearly to y_{i+1}^o

$$y_{i+1}^o = y_i + f(x_i, y_i) \times h \tag{24}$$

y_{i+1}^o is an intermediate prediction called *predictor*.

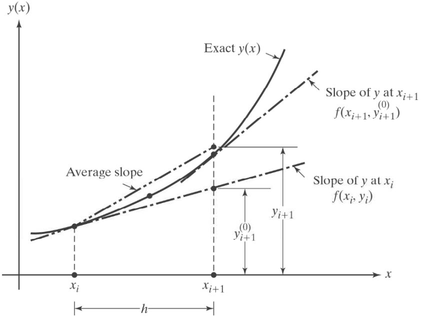

Figure 4 : Heun's method.

ODE - Initial Value Problem

Heun's Method

- Slope at end of interval is estimated using the predictor in Eq. (24)

$$y'_{i+1} = f(x_{i+1}, y^o_{i+1}) \tag{25}$$

- The two slopes—Eqs. (23) and (25)—are combined to obtain average slope for the interval

$$\bar{y} = \frac{y'_i + y'_{i+1}}{2} = \frac{f(x_i, y_i) + f(x_{i+1}, y^o_{i+1})}{2} \tag{26}$$

- The average slope, Eq. (26), is then used to extrapolate linearly from y_i to y_{i+1} using Euler's method

$$y_{i+1} = y_i + \left[\frac{f(x_i, y_i) + f(x_{i+1}, y^o_{i+1})}{2}\right] \times h \tag{27}$$

  Eq. (27) is known as the *corrector* equation

- Heun's method is the basis of many other *predictor-corrector* methods

Problem Statement:

Use Heun's method to solve

$$\frac{dy}{dx} = 4e^{0.8x} - 0.5y$$

from $x = 0$ to $x = 4$ with step size of 1.0. Initial condition at $x = 0$ is $y = 2$.

Note: Analytical solution is

$$y = \frac{4}{1.3}\left(e^{0.8x} - e^{-0.5x}\right) + 2e^{-0.5x}$$

Solution:

Slope at initial point (x_0, y_0)

$$y'_0 = 4e^0 - 0.5(2) = 3$$

Use predictor equation (Eq. (24)) to obtain estimate of y at 1.0

$$y_1^o = 2 + 3(1) = 5$$

Use value y_i^o to predict slope at end of interval

$$y'_1 = f(x_1, y_1^o) = 4e^{0.8(1)} - 0.5(5) = 6.402164$$

Average slope over the interval $0 \le x \le 1$ $x = 0$ and $x = 1$ is

$$y' = \frac{3 + 6.402164}{2} = 4.701082$$

This average slope is then fed into the corrector equation (Eq. (27)) to give an estimate at $x = 1$

$$y_1 = 2 + 4.701082(1) = 6.701082$$

ODE - Initial Value Problem

Example 2: Heun's Method

Solution: continued…

Computation is repeated and results tabulated

x	y_{true}	y_{Heun}	Error (%)
0.0	2.0000000	2.0000000	-0.00
1.0	6.1946314	6.70108190	-8.18
2.0	14.8439219	16.3197819	-9.94
3.0	33.6771718	37.1992489	-10.46
4.0	75.3389626	83.3377674	-10.62

Show the effect of reduced step size.

ODE - Initial Value Problem

Example 2: Heun's Method

Solution: continued. . .

Using Matlab to solve the same involves the following steps:

1. Write the ODE into `dydx1.m`
2. Optionally, write the 'known' solution into `ytrue1.m`
3. Matlab script to solve the ODE is written in `myheun.m`

Matlab codes

dydx1.m

```
function df=dydx1(x,y)
df = 4*exp(0.8*x) - 0.5*y;
```

ytrue1.m

```
function y=ytrue1(x)
y = (4/1.3)*(exp(0.8*x) - exp(-0.5*x)) + 2*exp(-0.5*x);
```

ODE - Initial Value Problem

Example 2: Heun's Method

Solution: continued. . .

myheun.m

```
x0=0.0; xn=4.0; xstep=0.25; x=[x0:xstep:xn]; n=length(x);
yt = ytrue1(x);
plot(x,yt,'r'); hold on
x(1)=0.0; y(1)=2.0;      % INITIAL CONDITION
for i=2:n
    islope = dydx1(x(i-1),y(i-1));
    predictor = y(i-1) + islope*xstep;
    eslope = dydx1(x(i),predictor);
    avslope = (islope + eslope)/2;
    y(i) = y(i-1) + avslope*xstep;
end
err = abs((yt-y)./yt)*100;
plot(x,y,'b')
[x' y' yt' err']
```

- Advantage—achieve accuracy of a Taylor series approach without requiring calculation of higher derivatives. Generalized form

$$y_{i+1} = y_i + \phi(x_i, y_i, h) \times h \tag{28}$$

where $\phi(x_i, y_i, h)$ is an *increment function* which is a representative slope over the interval and given by

$$\phi = a_1 k_1 + a_2 k_2 + \ldots + a_n k_n \tag{29}$$

- k's in Eq. (29) are *recurrence relationships* i.e. k_1 appears in k_2 which appears in k_3 and so forth

$$k_1 = f(x_i, y_i) \tag{30}$$
$$k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h) \tag{31}$$
$$k_3 = f(x_i + p_2 h, y_i + q_{21} k_1 h + q_{22} k_2 h) \tag{32}$$
$$\ldots$$
$$k_n = f(x_i + p_{n-1} h, y_i + q_{n-1,1} k_1 h + q_{n-2,2} k_2 h + \ldots + q_{n-1,n-1} k_{n-1} h) \tag{33}$$

ODE - Initial Value Problem

Runge-Kutta (RK) Methods

- Various types of RK methods can be devised by employing different numbers of terms in the increment function as specified by n
- Once n is chosen, values for a's, p's and q's are evaluated by setting Eq. (28) equal to terms in a Taylor series expansion.
- First order RK method, when $n = 1$, is actually Euler's method.
- Second order RK method, when $n = 2$, is exact if solution to ODE is quadratic.

ODE - Initial Value Problem

Runge-Kutta Order 1 (RK1) Method

- For $n = 1$, we can write, using Eq. (28), the first order RK (RK1) method as

  $$y_{i+1} = y_i + (a_1 k_1)h \tag{34}$$

  where

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

  With $a_1 = 1$, the first order RK method

  $$y_{i+1} = y_i + f(x_i, y_i)h \tag{35}$$

  can be seen to be the same as Euler's method.

- For $n = 2$, we can write, using Eq. (28), the second order RK (RK2) method as

$$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2)h \tag{36}$$

where

$$k_1 = f(x_i, y_i) \tag{37}$$

$$k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h) \tag{38}$$

To use Eq. (36), values for a_1, a_2, p_1 and q_{11} must first be determined.

- Recall second order Taylor series for y_{i+1} in terms of y_i and $f(x_i, y_i)$

$$y_{i+1} = y_i + f(x_i, y_i)h + f'(x_i, y_i)\frac{h^2}{2!} \tag{39}$$

where $f'(x_i, y_i)$, by *chain-rule differentiation*, yields

$$f'(x_i, y_i) = \frac{\partial f(x,y)}{\partial x} + \frac{\partial f(x,y)}{\partial y}\frac{dy}{dx} \tag{40}$$

- Substituting Eq. (40) into Eq. (39) yields

$$y_{i+1} = y_i + f(x_i, y_i)h + \left(\frac{\partial f}{\partial x} + \frac{\partial f}{\partial y}\frac{dy}{dx} \right) \frac{h^2}{2!} \tag{41}$$

- Taylor series for two-variable function is

$$g(x + r, y + s) = g(x, y) + r\frac{\partial g}{\partial x} + s\frac{\partial g}{\partial y} + \ldots$$

and apply this to expand Eq. (38)

$$f(x_i + p_1 h, y_i + q_{11} k_1 h) = f(x_i, y_i) + p_1 h\frac{\partial f}{\partial x} + q_{11} k_1 h\frac{\partial f}{\partial y} + O(h^2) \tag{42}$$

ODE - Initial Value Problem

Runge-Kutta Order 2 (RK2) Methods

- Eqs. (42) and (37) are then substituted into Eq. (36) to yield

$$
\begin{aligned}
y_{i+1} = y_i + a_1 h f(x_i, y_i) + a_2 h f(x_i, y_i) + a_2 p_1 h^2 \frac{\partial f}{\partial x} + \\
a_2 q_{11} h^2 f(x_i, y_i) \frac{\partial f}{\partial y} + O(h^3)
\end{aligned}
\tag{43}
$$

  and collecting terms we get

$$
\begin{aligned}
y_{i+1} = y_i + \Big[a_1 f(x_i, y_i) + a_2 f(x_i, y_i) \Big] h + \\
\Big[a_2 p_1 \frac{\partial f}{\partial x} + a_2 q_{11} f(x_i, y_i) \frac{\partial f}{\partial y} \Big] h^2 + O(h^3)
\end{aligned}
\tag{44}
$$

- Comparing like terms in Eqs. (41) and (44), the following must hold

$$\begin{aligned} a_1 + a_2 &= 1 \\ a_2 p_1 &= \tfrac{1}{2} \\ a_2 q_{11} &= \tfrac{1}{2} \end{aligned}$$

1. We now have 3 simultaneous equations containing 4 unknown constants—hence, there's no unique solution.
2. However, assuming a value for one of the constants, we can determine the other 3!
3. Consequently, there is a family of second order RK methods rather than a single version!

- The second order RK method version of Eq. (28) is

$$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2)h \tag{45}$$

where

$$k_1 = f(x_i, y_i) \tag{46}$$

$$k_1 = f(x_i + p_1 h, y_i + q_{11} k_1 h) \tag{47}$$

$$a_1 + a_2 = 1 \tag{48}$$

$$a_2 p_1 = \tfrac{1}{2} \tag{49}$$

$$a_2 q_{11} = \tfrac{1}{2} \tag{50}$$

- Suppose we specify a value for a_2, then Eqs. (48)–(50) can be solved simultaneously for

$$a_1 = 1 - a_2 \tag{51}$$

$$p_1 = q_{11} = \frac{1}{2a_2} \tag{52}$$

ODE - Initial Value Problem

Runge-Kutta Order 2 (RK2) Methods

- **Heun's Method with a Single Corrector:** $a_2 = \frac{1}{2}$

  If $a_2 = \frac{1}{2}$, Eqs. (51) and (52) can be solved for $a_1 = \frac{1}{2}$ and $p_1 = q_{11} = 1$. These parameters, when substituted into Eq. (45) yield

  $$y_{i+1} = y_i + (\tfrac{1}{2}k_1 + \tfrac{1}{2}k_2)h \tag{53}$$

  where

  $$k_1 = f(x_i, y_i) \quad \text{: slope at begining of interval} \tag{54}$$

  $$k_2 = f(x_i + h, y_i + k_1 h) \quad \text{: slope at end of interval} \tag{55}$$

ODE - Initial Value Problem

Runge-Kutta Order 2 (RK2) Methods

- **The Midpoint Method:** $a_2 = 1$

  If $a_2 = 1$, then $a_1 = 0$ and $p_1 = q_{11} = \frac{1}{2}$ and Eq. (45) becomes

  $$y_{i+1} = y_i + k_2 h \tag{56}$$

  where

  $$k_1 = f(x_i, y_i) \tag{57}$$

  $$k_2 = f(x_i + \tfrac{1}{2}h, y_i + \tfrac{1}{2}k_1 h) \tag{58}$$

ODE - Initial Value Problem

Runge-Kutta Order 2 (RK2) Methods

- **Ralston's Method:** $a_2 = \frac{2}{3}$

  Choosing $a_2 = \frac{2}{3}$ provides a minimum bound on the truncation error for second order RK algorithms. It leads to $a_1 = \frac{1}{3}$ and $p_1 = q_{11} = \frac{3}{4}$, and yields

$$y_{i+1} = y_i + \left(\tfrac{1}{3}k_1 + \tfrac{2}{3}k_2 \right) h \tag{59}$$

  where

$$k_1 = f(x_i, y_i) \tag{60}$$

$$k_2 = f(x_i + \tfrac{3}{4}h, y_i + \tfrac{3}{4}k_1 h) \tag{61}$$

- For $n = 3$, we can write, using Eq. (28), the third order RK (RK3) method as

$$y_{i+1} = y_i + (a_1k_1 + a_2k_2 + a_3k_3)h \tag{62}$$

where

$$k_1 = f(x_i, y_i) \tag{63}$$
$$k_2 = f(x_i + p_1h, y_i + q_{11}k_1h) \tag{64}$$
$$k_3 = f(x_i + p_2h, y_i + q_{21}k_1h + q_{22}k_2h) \tag{65}$$

To use Eq. (62), values for a_1, a_2, a_3, p_1, p_2, q_{11}, q_{21} and q_{22} must first be determined.

ODE - Initial Value Problem

Runge-Kutta Order 3 (RK3) Methods

- One of the more popular version of Eq. (62) is

$$y_{i+1} = y_i + \tfrac{1}{6}(k_1 + 4k_2 + k_3)h \tag{66}$$

where

$$\begin{aligned} k_1 &= f(x_i, y_i) \\ k_2 &= f(x_i + \tfrac{1}{2}h, y_i + \tfrac{1}{2}hk_1) \\ k_3 &= f(x_i + h, y_i - hk_1 + 2hk_2) \end{aligned}$$

Note: See Section 9.7.3 on page 658 of Rao (2002) for details.

- RK4 methods are by far the most popular, with the *classical fourth order RK method* being the most commonly used:

$$y_{i+1} = y_i + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)h \tag{67}$$

where

$$\begin{aligned}
k_1 &= f(x_i, y_i) \\
k_2 &= f(x_i + \tfrac{1}{2}h, y_i + \tfrac{1}{2}hk_1) \\
k_3 &= f(x_i + \tfrac{1}{2}h, y_i + \tfrac{1}{2}hk_2) \\
k_4 &= f(x_i + h, y_i + hk_3)
\end{aligned}$$

Note: See Section 25.3.3 on page 707 of Chapra and Canale (2006) for details.

ODE - Initial Value Problem

Example 3: Runge-Kutta Order 4 (RK4) Methods

Problem Statement:

Use the classical RK4 method to solve

$$f(x) = \frac{dy}{dx} = -2x^3 + 12x^2 - 20x + 8.5$$

using step size of $h = 0.5$ and initial condition of $y = 1$ at $x = 0$.

Solution:

At $x = 0.0$

$$\begin{aligned}
k_1 &= -2(0.0)^3 + 12(0.0)^2 - 20(0.0) + 8.5 = 8.5000 \\
k_2 &= -2(0.0 + 0.5 * 0.5)^3 + 12(0.0 + 0.5 * 0.5)^2 - 20(0.0 + 0.5 * 0.5) + 8.5 = 4.2188 \\
k_3 &= -2(0.0 + 0.5 * 0.5)^3 + 12(0.0 + 0.5 * 0.5)^2 - 20(0.0 + 0.5 * 0.5) + 8.5 = 4.2188 \\
k_4 &= -2(0.0 + 0.5)^3 + 12(0.0 + 0.5)^2 - 20(0.0 + 0.5) + 8.5 = 1.2500
\end{aligned}$$

Hence, estimate at $x = 0.5$ is

$$\begin{aligned}
y_{i+1} &= y_i + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)h \\
y(0.5) &= 1 + \left(\tfrac{1}{6}\left[8.5 + 2(4.21875) + 2(4.21875) + 1.25\right]\right)0.5 = 3.2188
\end{aligned}$$

ODE - Initial Value Problem

Example 3: Runge-Kutta Order 4 (RK4) Methods

Solution: continued…

At $x = 0.5$

$$\begin{aligned}
k_1 &= -2(0.5)^3 + 12(0.5)^2 - 20(0.5) + 8.5 = 1.25000 \\
k_2 &= -2(0.5 + 0.5 * 0.5)^3 + 12(0.0 + 0.5 * 0.5)^2 - 20(0.0 + 0.5 * 0.5) + 8.5 = -0.5938 \\
k_3 &= -2(0.0 + 0.5 * 0.5)^3 + 12(0.0 + 0.5 * 0.5)^2 - 20(0.0 + 0.5 * 0.5) + 8.5 = -0.5938 \\
k_4 &= -2(0.5 + 0.5)^3 + 12(0.5 + 0.5)^2 - 20(0.5 + 0.5) + 8.5 = -1.50000
\end{aligned}$$

Hence, estimate at $x = 1.0$ is

$$y_{i+1} = y_i + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)h$$

$$y(1.0) = 1 + \left(\frac{1}{6}\left[1.2500 + 2(-0.5938) + 2(-0.5938) - 1.5000\right]\right)0.5 = 3.0000$$

ODE - Initial Value Problem

Example 4: Runge-Kutta Order 4 (RK4) Methods

Problem Statement:

Use the classical RK4 method to solve

$$f(x,y) = \frac{dy}{dx} = 4e^{0.8x} - 0.5y$$

from $x = 0$ to $x = 0.5$ with step size of 0.5 and initial condition of $y(0) = 2$.

Solution:

Using Matlab to solve the above involves the following steps:

1. Write the ODE into `dydx1.m`
2. Optionally, write the 'known' solution into `ytrue1.m`
3. Matlab script to solve the ODE is written in `myrk4xy.m`

ODE - Initial Value Problem

Example 4: Runge-Kutta Order 4 (RK4) Methods

Solution: continued. . .

Matlab codes

dydx1.m

```
function df=dydx1(x,y)
df = 4*exp(0.8*x) - 0.5*y;
```

ytrue1.m

```
function y=ytrue1(x)
y = (4/1.3)*(exp(0.8*x) - exp(-0.5*x)) + 2*exp(-0.5*x);
```

ODE - Initial Value Problem

Example 4: Runge-Kutta Order 4 (RK4) Methods

Solution: continued. . .

myrk4xy.m

```
x0 = 0.0; xn = 0.5; h=0.05;
x  = [x0:h:xn]; n  = length(x);
yt = ytrue1(x);
plot(x,yt,’r’); hold on
x(1) = 0.0; y(1) = 0.5;      % INITIAL CONDITION
for i=2:n
    k1 = dydx1(x(i-1),y(i-1));
    k2 = dydx1(x(i-1)+0.5*h,y(i-1)+0.5*k1*h);
    k3 = dydx1(x(i-1)+0.5*h,y(i-1)+0.5*k2*h);
    k4 = dydx1(x(i-1)+h,y(i-1)+k3*h);
    phi = (k1+2*k2+2*k3+k4)/6;
    y(i) = y(i-1) + phi*h;
end
err = abs((yt-y)./yt)*100;
plot(x,y,’o’)
display(’x y yt err’)
[x’ y’ yt’ err’]
```

ODE - Initial Value Problem

Example 5: Using Matlab's `ode23` Function

Problem Statement:

Use built-in Matlab function `ode23` to numerically solve first order ODE

$$f(x,y) = \frac{dy}{dx} = xy^2 + y$$

Initial condition of at $x = 0$ is $y = 1$.

Solution:

Using Matlab to solve the same involves the following steps:

1. Write the ODE into `myode1.m`
2. Matlab script to solve `myode1.m` is written in `solvemyode1.m`

ODE - Initial Value Problem

Example 5: Using Matlab's `ode23` Function

Solution: (continued. . .)
Matlab codes

myode1.m

```
function df=myode1(x,y)
yprime = x*y.^2 + y;
```

solvemyode1.m

```
xr = [0,0.5];          % set x-range
y0 = 1;                % initial conditions
[x,y] = ode23('myode1',xr,y0)
plot(x,y)
```

ODE - Initial Value Problem

Example 6: Using Matlab's `ode45` Function

Problem Statement:

Use built-in Matlab function `ode45` to solve the first order ODE

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

Initial condition of at $x = 0$ is $y = 1$.

Solution:

Using Matlab to solve the above involves the following steps:

1. Write the ODE into `iv1.m`
2. Matlab script to solve `iv1.m` is written in `solveiv1.m`

ODE - Initial Value Problem

Example 6: Using Matlab's `ode45` Function

Solution: (continued. . .)

Matlab codes

iv1.m

```
function ydot=iv1(t,y)
ydot = -y+t+1;
```

solveiv1.m

```
tspan = [0,1.0];        % set time span
y0 = 1;                 % initial condition
[t,y] = ode45(’iv1’,tspan,y0)
plot(t,y)
```

We will show this kind of problem through an example:

$$\frac{d^2x}{dt^2} + 3\frac{dx}{dt} + x = 5$$

This is n^{th} order problem where, in this case $n = 2$. It can be re-written as a system of n first order ODE. This can be done in the following steps:

1. Let $x_1 = x$, then

   $$\frac{d^2x_1}{dt^2} + 3\frac{dx_1}{dt} + x_1 = 5$$

   or, re-written as

   $$\frac{d^2x_1}{dt^2} = -3\frac{dx_1}{dt} - x_1 + 5 \tag{68}$$

ODE - Initial Value Problem

Problems of Order n

② Next, let

$$\frac{dx_1}{dt} = x_2 \tag{69}$$

This is the first equation of the system.

③ Differentiate x_2 and substitute it into Eq. (69)

$$\frac{dx_2}{dt} = \frac{d^2x_1}{dt^2} = -3\frac{dx_1}{dt} - x_1 + 5 = -3x_2 - x_1 + 5 \tag{70}$$

This is the second equation of the system.

④ System of ODE represented by the first order Eqs. (69) and (70) can be then be written into a user-defined function in Matlab–see Matlab codes `iv0.m`.

Matlab codes

iv0.m

```
function ydot=iv0(t,y)
y1dot = y(2);
y2dot = -3*y(2) - y(1) + 5;
ydot = [y1dot;y2dot];
```

ODE - Initial Value Problem

Example 7: Problems of Order $n = 2$

Problem Statement:

Use built-in Matlab function(s) to solve the classic van de Pol equation

$$\ddot{x} - \mu(1 - x^2)\dot{x} + x = 0$$

where μ is a parameter greater than zero.

Solution:

Choose

$$y_1 = x \qquad \Longrightarrow \qquad \dot{y_1} = \frac{dx}{dt} = y_2$$

$$y_2 = \frac{dx}{dt} \qquad \Longrightarrow \qquad \dot{y_2} = \frac{d^2x}{dt^2} = \mu(1 - y_1^2)y_2 - y_1$$

ODE - Initial Value Problem

Example 7: Problems of Order $n = 2$

Solution: (continued. . .)

Using Matlab to solve the above involves the following steps:

1. Write the system of first order ODE into `vdpol.m`
2. Matlab script to solve `vdpol.m` is written in `solvevdpol.m`

Matlab codes:

vdpol.m

```
function ydot=vdpol(t,y)
mu = 2;           % set to a value greater than zero
y1dot = y(2);
y2dot = mu*(1-y(1)^2)*y(2)-y(1)
ydot = [y1dot; y2dot];
```

solvevdpol.m

```
tspan=[0 20];                     % time span
yo = [2; 0];                      % initial condition
[t,y] = ode45(’vdpol’,tspan,yo);  % solve the ODE-IVP
plot(t,y(:,1),t,y(:,2),’--’)
xlabel(’time’)
ylabel(’y1dot and y2dot’)
title(’van der Pol Solution’)
```

Problem Statement:

Use built-in Matlab function(s) to solve the following third order ODE

$$\frac{d^3y}{dt^3} + 3\frac{d^2y}{dt^2} + 5\frac{dy}{dt} + 7y = 6$$

This is n^{th} order problem with $n = 3$. It can be re-written as a system of 3 first order ODE.

ODE - Initial Value Problem

Example 8: Problems of Order $n = 3$

Solution:

1. Let $x_1 = y$, then

$$\frac{d^3 x_1}{dt^3} + 3\frac{d^2 x_1}{dt^2} + 5\frac{dx_1}{dt} + 7x_1 = 6$$

or, re-written as

$$\frac{d^3 x_1}{dt^3} = -3\frac{d^2 x_1}{dt^2} - 5\frac{dx_1}{dt} - 7x_1 + 6 \tag{71}$$

Solution: continued...

② Let

$$x_2 = \frac{dx_1}{dt} \tag{72}$$

This is the first equation of the system. Differentiate x_2, twice

$$\frac{dx_2}{dt} = \frac{d^2x_1}{dt^2} \tag{73a}$$

$$\frac{d^2x_2}{dt^2} = \frac{d^3x_1}{dt^3} \tag{73b}$$

ODE - Initial Value Problem

Example 8: Problems of Order $n = 3$

Solution: continued. . .

3. Let

$$\frac{dx_2}{dt} = x_3$$

This is the second equation of the system. But from Eq. (73a)

$$\frac{dx_2}{dt} = \frac{d^2x_1}{dt^2}$$

Thus

$$x_3 = \frac{d^2x_1}{dt^2} \tag{74}$$

Solution: continued…

4. Differentiate x_3

$$\frac{dx_3}{dt} = \frac{d^3x_1}{dt^3} \tag{75}$$

Subsituting Eqs. (75), (74) and (72) into Eq. (71) and making substitution $dx_3/dt = d^3x_1/dt^3$, etc. yields

$$\frac{dx_3}{dt} = -3x_3 - 5x_2 - 7x_1 + 6 \tag{76}$$

This is the third equation of the system.

ODE - Initial Value Problem

Example 8: Problems of Order $n = 3$

Solution: continued...

Thus, the third order ODE

$$\frac{d^3y}{dt^3} + 3\frac{d^2y}{dt^2} + 5\frac{dy}{dt} + 7y = 6$$

can be re-written as a system of 3 first order ODE:

$$\frac{dx_1}{dt} = x_2; \qquad \frac{dx_2}{dt} = x_3; \qquad \frac{dx_3}{dt} = -3x_3 - 5x_2 - 7x_1 + 6$$

ODE - Initial Value Problem

Example 8: Problems of Order $n = 3$

Solution: continued…

Using Matlab to solve the above involves the following steps:

1. Write a Matlab user-defined function called, say `iv3.m`
2. Matlab script to solve `iv3.m` is written in `solveiv3.m`

Matlab codes:

iv3.m

```
function ydot=iv3(t,x)
x1dot = x(2);
x2dot = x(3);
x3dot = -3*x(3) - 5*x(2) - 7*x(1) + 6;
xdot = [x1dot;x2dot;x3dot];
```

solveiv3.m

```
tspan=[0 20];                      % time span
xo = [2; 0];                       % initial condition
[t,x] = ode45(’iv3’,tspan,xo);     % solve the ODE-IVP
plot(t,x(:,1),t,x(:,2),t,x(:,3),’--’)
xlabel(’time’)
ylabel(’x1dot and x2dot x3dot’)
title(’ODE of Order 3’)
```

ODE - Boundary Value Problem

IVP, BVP & Methods of Numerical Solution

- Differences between ODE-IVP and ODE-BVP are:

ODE - Initial Value Problem (ODE-IVP)	ODE - Boundary Value Problem (ODE-BVP)
- conditions are specified at the ***same*** value of the independent variable - for n^{th} order ODE, n conditions are required.	- conditions are specified at the ***different*** values of the independent variable - because these conditions are often specified at the extreme points or boundaries of a system, they are referred to as ODE-BVP.

- ODE-BVP may be classified into
  - linear or nonlinear,
  - separated or mixed,
  - specified at two points or more.
- Two general approaches to numerical solution of ODE-BVP are
  - shooting method,
  - finite-difference method

ODE - Boundary Value Problem

Numerical Solution

- Take conservation of energy to develop a heat balance for a long, thin rod. Mathematical model is represented by

$$\frac{d^2T}{dx^2} + h(T_a - T) = 0 \tag{77}$$

where

h : heat transfer coefficient
T_a : temperature of surrounding air

- To solve Eq. (77), appropriate boundary conditions (BC) must be specified. They are

$$T(0) = T_1 \qquad \text{and} \qquad T(L) = T_2$$

With these two BC, Eq. (77) can be solved analytically using calculus. If $L = 10$ m, $T_a = 20^\circ$C, $T_1 = 40^\circ$C, $T_2 = 200^\circ$C and $h = 0.01$, the analytical solution is

$$T = 73.4523e^{0.1x} - 53.4523e^{-0.1x} + 20 \tag{78}$$

Problem Statement:

To show both solution approaches for ODE-BVP, we will solve Eq. (77)

$$\frac{d^2T}{dx^2} + h(T_a - T) = 0$$

for a 10 m rod with $h = 0.001$, $T_a = 20$ and BC: $T(0) = 40$ and $T(10) = 200$

Solution:

Shooting methods is

1. based on converting BVP into equivalent IVP.
   Here, the second order ODE of Eq. (77) is transformed into two first order ODE

   $$\frac{dT}{dx} = z; \qquad \frac{dz}{dx} = h(T - T_a)$$

   We need initial value for z to solve this system of ODE, using, for example, RK4.
2. A trial-and-error approach is then implemented to solve IVP.

ODE - Boundary Value Problem

Numerical Solution: Finite Difference Method

Finite difference methods is . . . (coming soon to classroom near you!)

Bibliography

1. Steven C. Chapra, Raymond P. Canale (2006): *Applied Numerical Methods for Engineers, 5ed*, ISBN 007-124429-8, McGraw-Hill
2. Singiresu S. Rao (2002): *Applied Numerical Methods for Engineers and Scientists*, ISBN 0-13-089480-X, Prentice Hall
3. David Kincaid and Ward Cheney (1991): *Numerical Analysis: Mathematics of Scientific Computing*, ISBN 0-534-13014-3, Brooks/Cole Publishing Co.
4. Steven C. Chapra (2005): *Applied Numerical Methods with MATLAB for Engineers and Scientists*, ISBN 007-124484-0, McGraw-Hill
5. William J. Palm III (2005): *Introduction to Matlab 7 for Engineers*, ISBN 007-123262-1, McGraw-Hill
6. John D. Anderson, Jr. (1995): *Computational Fluid Dynamics–The Basics with Applications*, ISBN 007-113210-4, McGraw-Hill