**Ordinary Differential Equations - Initial Value Problem**

- Taylor’s series method
- Euler method
- Runge Kutta method
- Computer-based solutions
  - IVPAG subroutine (Fortran IMSL subroutine that solves an initial-value problem for ordinary differential equations using either Adams-Moulton’s or Gear’s BDF method)
  - MATLAB solution of initial value problems for ordinary differential equations (ode23, ode45, ode113, ode15s, ode23s, ode23t, ode23tb))
Classification of ODE

ODE can be classified in different ways
- **Order**
  - First order ODE
  - Second order ODE
  - \(N^{th}\) order ODE
- **Linearity**
  - Linear ODE
  - Nonlinear ODE
- **Auxiliary conditions**
  - Initial value problems
  - Boundary value problems

Classification of the Methods

Numerical Methods for solving ODE

- **Single-Step Methods**
  - Euler, Runge-Kutta: single step methods
  - Estimates of the solution at a particular step are entirely based on information on the previous step

- **Multiple-Step Methods**
  - Adam-Moulton method: multi-step method
  - Estimates of the solution at a particular step are based on information on more than one step

Taylor Series Method

The problem to be solved is a first order ODE

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

Estimates of the solution at different base points

\[
y(x_0 + h), \quad y(x_0 + 2h), \quad y(x_0 + 3h), \quad \ldots
\]

are computed using truncated Taylor series expansions

Taylor Series Expansion

Truncated Taylor Series Expansion

\[
y(x_0 + h) = y(x_0) + \sum_{k=0}^n \frac{h^k}{k!} \frac{d^k y}{dx^k} \bigg|_{x=x_0, y=y_0}
\]

\[
= y(x_0) + h \frac{dy}{dx} \bigg|_{x=x_0, y=y_0} + \frac{h^2}{2!} \frac{d^2 y}{dx^2} \bigg|_{x=x_0, y=y_0} + \ldots + \frac{h^n}{n!} \frac{d^n y}{dx^n} \bigg|_{x=x_0, y=y_0}
\]

\(n^{th}\) order Taylor series method uses \(n^{th}\) order Truncated Taylor series expansion
Euler Method

- First order Taylor series method is known as Euler Method.
- Only the constant term and linear term are used in Euler method.
- The error due to the use of the truncated Taylor series is of order $O(h^2)$.

Leonhard Euler
(Basel, April 15, 1707 – September 18, 1783)

- $y = f(x)$
- $\frac{dy}{dx} + \nabla \rho = 0$

First Order Taylor Series Method ... (Euler Method)

\[ y(x_n + h) = y(x_n) + h \frac{dy}{dx} \bigg|_{x=x_n} + o(h^2) \]

Notation:
- $x_n = x_0 + nh$
- $y_n = y(x_n)$
- $\frac{dy}{dx} \bigg|_{x=x_n} = f(x_n, y_n)$

Euler Method
\[ y_{i+1} = y_i + h \cdot f(x_i, y_i) \]

Problem:
Given the first order ODE \[ \dot{y}(x) = f(x, y) \]
with the initial condition \[ y_0 = y(x_0) \]
Determine \[ y_i = y(x_0 + ih) \quad \text{for} \quad i = 1, 2, \ldots \]

Euler Method:
\[ y_0 = y(x_0) \]
\[ y_{i+1} = y_i + h \cdot f(x_i, y_i) \quad \text{for} \quad i = 1, 2, \ldots \]

Interpretation of Euler Method
Example 1

Use Euler method to solve the ODE

\[
\frac{dy}{dx} = 1 + x^2, \quad y(1) = -4
\]

to determine \(y(1.01), y(1.02)\) and \(y(1.03)\)
Example 1

\[ f(x, y) = 1 + x^2, \quad x_0 = 1, \quad y_0 = -4, \quad h = 0.01 \]

Summary of the result

<table>
<thead>
<tr>
<th>i</th>
<th>xi</th>
<th>yi</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.00</td>
<td>-4.00</td>
</tr>
<tr>
<td>1</td>
<td>1.01</td>
<td>-3.98</td>
</tr>
<tr>
<td>2</td>
<td>1.02</td>
<td>-3.9595</td>
</tr>
<tr>
<td>3</td>
<td>1.03</td>
<td>-3.9394</td>
</tr>
</tbody>
</table>

Comparison with true value

<table>
<thead>
<tr>
<th>i</th>
<th>xi</th>
<th>yi</th>
<th>True value of yi</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.00</td>
<td>-4.00</td>
<td>-4.00</td>
</tr>
<tr>
<td>1</td>
<td>1.01</td>
<td>-3.98</td>
<td>-3.97996</td>
</tr>
<tr>
<td>2</td>
<td>1.02</td>
<td>-3.9595</td>
<td>-3.95995</td>
</tr>
<tr>
<td>3</td>
<td>1.03</td>
<td>-3.9394</td>
<td>-3.93091</td>
</tr>
</tbody>
</table>

Example 1

\[ f(x, y) = 1 + x^2, \quad x_0 = 1, \quad y_0 = -4, \quad h = 0.01 \]

A graph of the solution of the ODE for 1 < x < 2

Types of Errors

- **Local truncation error:**
  error due to the use of truncated Taylor series to compute \( x(t+h) \) in one step.
- **Global Truncation error**
  accumulated truncation over many steps
- **Round off error:**
  error due to finite number of bits used in representation of numbers. This error could be accumulated and magnified in succeeding steps.
Second Order Taylor Series methods

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

Second order Taylor Series method

\[
y_{i+1} = y_i + h \frac{dy}{dx} + \frac{h^2}{2!} \frac{d^2y}{dx^2} + O(h^3)
\]

\( \frac{d^2y}{dx^2} \) needs to be derived analytically.

Third Order Taylor Series methods

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

Third order Taylor Series method

\[
y_{i+1} = y_i + h \frac{dy}{dx} + \frac{h^2}{2!} \frac{d^2y}{dx^2} + \frac{h^3}{3!} \frac{d^3y}{dx^3} + O(h^4)
\]

\( \frac{d^3y}{dx^3} \) need to be derived analytically.

High Order Taylor Series methods

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

\( n \)th order Taylor Series method

\[
y_{i+1} = y_i + h \frac{dy}{dx} + \frac{h^2}{2!} \frac{d^2y}{dx^2} + \ldots + \frac{h^n}{n!} \frac{d^ny}{dx^n} + O(h^{n+1})
\]

\( \frac{d^2y}{dx^2}, \frac{d^3y}{dx^3}, \ldots, \frac{d^n y}{dx^n} \) need to be derived analytically.

High Order Taylor Series methods

- High order Taylor series methods are more accurate than Euler method.
- The 2nd, 3rd and higher order derivatives need to be derived analytically which may not be easy.
Example 2 - Second order Taylor Series Method

Use Second order Taylor Series method to solve

\[ \frac{dx}{dt} + 2x^2 + t = 1, \quad x(0) = 1, \quad \text{use} \quad h = 0.01 \]

**Question:** What is \( \frac{d^2x(t)}{dt^2} \)?

\[ f(t, x) = 1 - 2x^2 - t, \quad t_0 = 0, \quad x_0 = 1, \quad h = 0.01 \]

\[
\begin{align*}
x_{i+1} = x_i + h(1 - 2x_i^2 - t_i) + \frac{h^2}{2}(-1 - 4x_i(1 - 2x_i^2 - t_i))
\end{align*}
\]

**Step 1:**

\[
\begin{align*}
x_1 = 1 + 0.01(1 - 0 - 2(0^2) + \frac{(0.01)^2}{2}(-1 - 4(1)(-1)) = 0.9901
\end{align*}
\]

**Step 2:**

\[
\begin{align*}
x_2 = 1 + 0.01(1 - 0.01 - 2(0.9901)^2) + \frac{(0.01)^2}{2}(-1 - 4(0.9901)(-1)) = 0.9807
\end{align*}
\]

**Step 3:**

\[
\begin{align*}
x_3 = 0.9716
\end{align*}
\]

**Example 2 - Second order Taylor Series Method**

Use Second order Taylor Series method to solve

\[ \frac{dx}{dt} + 2x^2 + t = 1, \quad x(0) = 1, \quad \text{use} \quad h = 0.01 \]

\[
\begin{align*}
x_{i+1} = x_i + h(1 - 2x_i^2 - t_i) + \frac{h^2}{2}(-1 - 4x_i(1 - 2x_i^2 - t_i))
\end{align*}
\]

**Summary of the results**

<table>
<thead>
<tr>
<th>i</th>
<th>t_i</th>
<th>x_i</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.00</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0.01</td>
<td>0.9901</td>
</tr>
<tr>
<td>2</td>
<td>0.02</td>
<td>0.9807</td>
</tr>
<tr>
<td>3</td>
<td>0.03</td>
<td>0.9716</td>
</tr>
</tbody>
</table>
Programming Euler Method

Write a MATLAB program to implement Euler method to solve

\[ \frac{dv}{dt} = 1 - 2v^2 - t \quad v(0) = 1 \]

for \( t_i = 0.01i, \quad i = 1, 2, \ldots, 100 \)

Algorithm Euler Method

1. Input \( t_0, t_f, h \)
2. \( t = t_0 \)
3. \( v = f(t, v) \)
4. \( t = t + h \)
5. \( v = v + h \cdot f(t, v) \)
6. Repeat steps 4 and 5 until \( t = t_f \)
7. Plot \( t \) vs. \( v \)
8. Stop
Programming Euler Method

Plot of the solution

plot(T, V)

Runge-Kutta Method

Second Order Runge Kutta

\[ K_1 = f(x_i, y_i) \]
\[ K_2 = f(x_i + \alpha h, y_i + \beta K_1 h) \]
\[ y_{i+1} = y_i + w_1 K_1 + w_2 K_2 \]

Problem:

Find \( \alpha, \beta, w_1, w_2 \)

such that \( y_{i+1} \) is as accurate as possible.

Runge-Kutta Method

Motivation

- Find accurate methods to solve ODE that does not require calculating high order derivatives.
- The approach is to a formula involving unknown coefficients then determine these coefficients to match as many terms of the Taylor series expansion

Motivation

- Find accurate methods to solve ODE that does not require calculating high order derivatives.
- The approach is to a formula involving unknown coefficients then determine these coefficients to match as many terms of the Taylor series expansion.

Runge-Kutta Method

\[ x(t + h) = x(t) + h f(t, x) + \frac{h^2}{2} f(t + \alpha h, x + \beta h f(t, x)) + \ldots \]
\[ x(t + h) = x(t) + w_1 f(t, x) + w_2 f(t + \alpha h, x + \beta h f(t, x)) \]

\[ f(t + \alpha h, x + \beta h f) = f(t) + \alpha h f_x + \beta h f_f + \frac{1}{2} \left( \frac{\alpha h}{h_x} - \beta h \frac{h}{h_x} \right)^2 f(t, x) \]
\[ x(t + h) = x(t) + (w_1 + w_2) h f(t, x) + \alpha w_1 h f_x + \beta w_2 h f_f + O(h^3) \]
Runge-Kutta Method

\[
x(t+h) = x(t) + h \left[ \int_{t}^{t+h} \frac{1}{2} x''(t) + O(h^2) \right]
\]

\[
x(t+h) = x(t) + (w_1 + w_2) h f(t,x) + \omega w_3 f_2 + \beta w_2 f_{12} + O(h^2)
\]

\[
\Rightarrow w_1 = w_2 = 1, \quad \alpha w_2 = 0.5, \quad \beta w_2 = 0.5
\]

One possible solution
\[
w_1 = 0.5, \quad w_2 = 0.5, \quad \alpha = 1, \quad \beta = 1
\]

Runge-Kutta Method

Second Order Runge Kutta
\[
K_1 = h f(t,x)
\]
\[
K_2 = h f(t+h,x + K_1)
\]
\[
x(t+h) = x(t) + \frac{1}{2} (K_1 + K_2)
\]

Alternative Formula

Second Order Runge Kutta
\[
F_1 = f(t,x)
\]
\[
F_2 = f(t+h,x + hF_1)
\]
\[
x(t+h) = x(t) + \frac{h}{2} (F_1 + F_2)
\]

Runge-Kutta Method

Third Order Runge Kutta (RK3)
\[
K_1 = f(x_1, y_1)
\]
\[
K_2 = f(x_1 + \frac{1}{2} h, y_1 + \frac{1}{2} K_1 h)
\]
\[
K_3 = f(x_1 + \frac{1}{2} h, y_1 - K_1 h + 2 K_2 h)
\]
\[
y(x+h) = y(x) + \frac{1}{6} (K_1 + 4 K_2 + K_3)
\]
Runge-Kutta Method

Fourth Order Runge-Kutta

\[ K_1 = h \cdot f(t, x) \]
\[ K_2 = h \cdot f(t + \frac{1}{2}h, x + \frac{1}{2}K_1) \]
\[ K_3 = h \cdot f(t + \frac{1}{2}h, x + \frac{1}{2}K_2) \]
\[ K_4 = h \cdot f(t + h, x + K_3) \]
\[ x(t + h) = x(t) + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \]

Second order Runge-Kutta Method

Example

Solve the following system to find \( x(1.02) \) using RK2

\[ \dot{x}(t) = 1 + x^2(t) + t^3, \quad x(1) = -4, h = 0.01 \]
Second order Runge-Kutta Method
Example

Solve the following system to find \( x(1.02) \) using RK2
\[
\dot{x}(t) = 1 + x^2(t) + t^3, \quad x(1) = -4, h = 0.01
\]

STEP 1:
\[
K_1 = h \ f(t, x) = 0.01(1 + x^2 + t^3) = 0.18
\]
\[
K_2 = h \ f(t + h, x + K_1) = 0.01(1 + (x + 0.18)^2 + (t + 0.01)^3) = 0.1692
\]
\[
x(1.01) = x(1) + \frac{1}{2}(K_1 + K_2) = -4 + \frac{1}{2}(0.18 + 0.1692) = -3.8254
\]

Second order Runge-Kutta Method
Example

Second order Runge-Kutta Method
Example

STEP 2
\[
K_1 = h \ f(t, x) = 0.01(1 + x^2 + t^3) = 0.1666
\]
\[
K_2 = h \ f(t + h, x + K_1) = 0.01(1 + (x + 0.1666)^2 + (t + 0.01)^3) = 0.1545
\]
\[
x(1.01 + 0.01) = x(1.01) + \frac{1}{2}(K_1 + K_2)
\]
\[
= -3.8254 + \frac{1}{2}(0.1666 + 0.1545) = -3.6648
\]
Second order Runge-Kutta Method

Given:
\[ \frac{dy}{dx} = f(x, y) \]
\[ y(x_0) = y_0 \]

Determine # of steps needed

**RK 2 formula**

\[ K_1 = f(x_i, y_i) \]
\[ K_2 = f(x_i + h, y_i + K_1 h) \]
\[ y_{i+1} = y_i + \frac{h}{2}(K_1 + K_2) \]

Second order Runge-Kutta Method … example

Use the second order Runge-Kutta method to solve the differential equation

\[ \frac{dy}{dx} = 1 + y^2 + x^3, \quad y(1) = -4 \]

to find \( y(1.01), y(1.02) \)

Problem:

\[ \frac{dy}{dx} = 1 + y^2 + x^3, \quad y(1) = -4 \]
\[ f(x, y) = 1 + y^2 + x^3 \]
\[ x_0 = 1, \quad y_0 = -4 \]

\( h = 0.01 \)

Use RK 2 to find \( y(1.01), y(1.02) \)
Problem:
\[ \frac{dy}{dx} = 1 + y^2 + x^2, \quad y(1) = -4 \]

Step 1:
\[ K_1 = f(x_0, y_0) = (1 + y_0^2 + x_0^2) = 18.0 \]
\[ K_2 = f(x_0 + h, y_0 + K_1 h) = (1 + (y_0 + 0.18)^2 + (x_0 + 0.01)^2) = 16.92 \]
\[ y_1 = y_0 + \frac{h}{2}(K_1 + K_2) = 4 - \frac{0.01}{2} (18 + 16.92) = -3.8254 \]

Step 2:
\[ K_1 = f(x_1, y_1) = (1 + y_1^2 + x_1^2) = 16.66 \]
\[ K_2 = f(x_1 + h, y_1 + K_1 h) = (1 + (y_1 + 0.1666)^2 + (x_1 + 0.01)^2) = 15.45 \]
\[ y_2 = y_1 + \frac{h}{2}(K_1 + K_2) = -3.8254 + \frac{0.01}{2} (16.66 + 15.45) = -3.6648 \]
Second order Runge-Kutta Method ... example

Solution after 100 steps

Fourth order Runge-Kutta Method ... example

\[
\frac{dy}{dx} = 1 + y + x^2 \\
y(0) = 0.5 \\
h = 0.2 \\
Use RK 4 to compute \ y(0.2) \ and \ y(0.4)
\]

Runge-Kutta Method

4th Order algorithm

\[
\begin{align*}
K_1 &= hf(x_n, y_n) \\
K_2 &= hf(x_n + \frac{h}{2}, y_n + \frac{K_1}{2}) \\
K_3 &= hf(x_n + \frac{h}{2}, y_n + \frac{K_2}{2}) \\
K_4 &= hf(x_n + h, y_n + K_3) \\
y_{n+1} &= y_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4)
\end{align*}
\]

Problem:

\[
\frac{dy}{dx} = 1 + y + x^2, \quad y(0) = 0.5
\]

Use RK 4 to find \( y(0.2), y(0.4) \)

Remember - RK4
Fourth order Runge-Kutta Method ... example

Problem :
\[ \frac{dy}{dx} = 1 + y + x^2, \quad y(0) = 0.5 \]
Use RK4 to find \( y(0.2), y(0.4) \)

Step 1:
\[ K_1 = y_x + x + x^2 = 1 + 0.2 + 0.2^2 = 1.44 \]
\[ K_2 = y_x + \frac{1}{2} x + \frac{1}{2} (K_1) = 1 + \frac{1}{2} (0.2) + \frac{1}{2} (1.44) = 1.7893 \]
\[ K_3 = y_x + \frac{1}{2} x + \frac{1}{2} (K_2) = 1 + \frac{1}{2} (0.2) + \frac{1}{2} (1.7893) = 1.9182 \]
\[ K_4 = y_x + x + \frac{1}{2} (K_3) = 1 + 0.2 + \frac{1}{2} (1.9182) = 1.948 \]
\[ y = y_0 + \frac{1}{6} (K_1 + 2K_2 + 2K_3 + K_4) = 0.8293 \]

Step 2:
\[ K_1 = f(x, y) = 1 + y + x^2 \]
\[ K_2 = f(x + \frac{1}{2} h, y + \frac{1}{2} (x + h)) = 1 + (0.2 + 0.8293) + (0.2)^2 = 1.9864 \]
\[ K_3 = f(x + \frac{1}{2} h, y + \frac{1}{2} (x + h)) = 1 + \frac{1}{2} (0.2 + 0.8293) + \frac{1}{2} (1.9864) = 1.9484 \]
\[ K_4 = f(x + h, y + h) = 1 + 0.2 + 0.8293 = 1.9893 \]
\[ y = y_0 + \frac{1}{6} (K_1 + 2K_2 + 2K_3 + K_4) = 0.8293 \]

Summary of the solution

<table>
<thead>
<tr>
<th>i</th>
<th>x_i</th>
<th>y_i</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0</td>
<td>0.5</td>
</tr>
<tr>
<td>1</td>
<td>0.2</td>
<td>0.8293</td>
</tr>
<tr>
<td>2</td>
<td>0.4</td>
<td>1.2141</td>
</tr>
</tbody>
</table>
Runge-Kutta Method ... Summary

- Runge Kutta methods generate accurate solution without the need to calculate high order derivatives.
- Second order RK have local truncation error of order \( O(h^2) \)
- Fourth order RK have local truncation error of order \( O(h^3) \)
- \( N \) function evaluations are needed in \( N \) th order RK method.

Computer-based solutions

- IPAG subroutine (Fortran IMSL subroutine that Solves an initial-value problem for ordinary differential equations using either Adams-Moulton's or Gear's BDF method)
- MATLAB solution of initial value problems for ordinary differential equations (ode23, ode45, ode113, ode15s, ode23s, ode23t, ode23tb)
Computer-based solutions

Example 1

Euler's equation for the motion of a rigid body subject to external forces is

\[ f_1 = g \frac{d}{dt} \left( \frac{d}{dt} \phi_1 \right) = 0 \]
\[ f_1 = -g \frac{d}{dt} \phi_2 \]
\[ f_2 = -0.1 \frac{d}{dt} \phi_3 \]

SUBROUTINE PGN (N, X, Y, YPRIME)

INTEGER N, NPARAM
PARAMETER (Np=4, NPARAM=40)
REAL X, Y(N), YPRIME(N)
REAL YPRIME(1) = Y(1)*Y(2)
YPRIME(2) = -T(i)*Y(3)
YPRIME(3) = -3.5*Y(1)*Y(2)
RETURN
END

SUBROUTINE FGN (N, X, YD)

INTEGER N
REAL X, YD(N), YD(N,*)
RETURN
END

looking for IMSL document (IVPAG.pdf) for detailed description and other examples

Look for IMSL document (UVPAG.pdf) for detailed description and other examples

See "Example of using UVPAG subroutine"
Computer-based solutions

- MATLAB solution of initial value problems for ordinary differential equations (ode23, ode45, ode113, ode15s, ode23s, ode23t, ode23tb)

ode23, ode45, ode113, ode15s, ode23s, ode23t, ode23tb - Solve initial value problems for ordinary differential equations

Syntax

\[
[T, Y] = \text{solver}(odefun, tspan, y0)
\]

\[
[T, Y] = \text{solver}(odefun, tspan, y0, options)
\]

\[
[T, Y, TE, YE, IE] = \text{solver}(odefun, tspan, y0, options)
\]

sol = solver(odefun, [t0 tf], y0, options)

Arguments ....

For detailed description, see the site:

Example 1

An example of a system of ODEs.

\[
\begin{align*}
y'_1 &= -2y_1 + y_2 \\
y'_2 &= -y_1 + y_2 \\
y_2(0) &= 0
\end{align*}
\]

To simulate the system, create a function file containing the equations:

\[
function \ y = system(t, y)
\]

\[
\begin{align*}
y' &= \text{zeros}(2,1) \\
\text{end}
\end{align*}
\]

\[
R(1) = y(1) + y(2)
\]

\[
R(2) = y(2) + y(3)
\]

In this example we obtain the error estimates using the columns command (as done in Example 2) at time 1:

\[
\text{options} = \text{odeset}('\text{RelTol}', 1e-3, '\text{AbsTol}', 1e-3);\]

\[
[t, y, te, ye, ie] = \text{ode15s}('\text{system}', [0 1], [1 0], options)
\]

Plotting the columns of the returned array 'y' versus 't' gives the solution:

\[
p\text{plot}(t, y(1,:), t, y(2,:))
\]