MA579H Scientific Computing

Numerics of first order ODEs-II

Rafikul Alam
Department of Mathematics
IIT Guwahati

Lecture outline

- Runge-Kutta method for ODEs
- Vector version of Euler's method for systems of first order ODEs.
- Converting higher order ODEs into a system of first order ODEs.

Second order Runge-Kutta method/Heun's method

Integrating $y' = f(x, y)$ on $[x_j, x_{j+1}]$, we have

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

Second order Runge-Kutta method/Heun's method

Integrating $y' = f(x, y)$ on $[x_j, x_{j+1}]$, we have

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

the trapezoid quadrature rule gives

$$\int_{x_j}^{x_{j+1}} f(t, y(t))dt \approx \frac{h}{2}[f(x_j, y(x_j)) + f(x_{j+1}, y(x_{j+1}))].$$

Second order Runge-Kutta method/Heun's method

Integrating $y' = f(x, y)$ on $[x_j, x_{j+1}]$, we have

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

the trapezoid quadrature rule gives

$$\int_{x_j}^{x_{j+1}} f(t, y(t))dt \approx \frac{h}{2}[f(x_j, y(x_j)) + f(x_{j+1}, y(x_{j+1}))].$$

This yields implicit trapezoid method

$$y_{j+1} = y_j + \frac{h}{2}[f(x_j, y_j) + f(x_{j+1}, y_{j+1})], \;\; j = 0 : n-1.$$

Second order Runge-Kutta method/Heun's method

Integrating $y' = f(x, y)$ on $[x_j, x_{j+1}]$, we have

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

the trapezoid quadrature rule gives

$$\int_{x_j}^{x_{j+1}} f(t, y(t))dt \approx \frac{h}{2}[f(x_j, y(x_j)) + f(x_{j+1}, y(x_{j+1}))].$$

This yields implicit trapezoid method

$$y_{j+1} = y_j + \frac{h}{2}[f(x_j, y_j) + f(x_{j+1}, y_{j+1})], \quad j = 0 : n-1.$$

To obtain an explicit method, we approximate y_{j+1} by forward Euler's method, which yields the second order Runge-Kutta (RK) method

Second order Runge-Kutta method/Heun's method

Integrating $y' = f(x, y)$ on $[x_j, x_{j+1}]$, we have

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

the trapezoid quadrature rule gives

$$\int_{x_j}^{x_{j+1}} f(t, y(t))dt \approx \frac{h}{2}[f(x_j, y(x_j)) + f(x_{j+1}, y(x_{j+1}))].$$

This yields implicit trapezoid method

$$y_{j+1} = y_j + \frac{h}{2}[f(x_j, y_j) + f(x_{j+1}, y_{j+1})], \;\; j = 0 : n-1.$$

To obtain an explicit method, we approximate y_{j+1} by forward Euler's method, which yields the second order Runge-Kutta (RK) method

$$\begin{aligned} Y_{j+1} &= y_j + hf(x_j, y_j) \\ y_{j+1} &= y_j + \frac{h}{2}[f(x_j, y_j) + f(x_{j+1}, Y_{j+1})], \;\; j = 0 : n-1. \end{aligned}$$

The second order RK method is also known as Heun's method.

Second order Runge-Kutta method

Again, consider

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

Second order Runge-Kutta method

Again, consider

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

By the midpoint quadrature rule

$$y(x_{j+1}) - y(x_j) \approx hf(x_{j+\frac{1}{2}}, y(x_{j+\frac{1}{2}})).$$

This gives the implicit midpoint method

$$y_{j+1} = y_j + hf(x_{j+1/2}, y_{j+1/2}), \;\; j = 0 : n-1,$$

where $x_{j+1/2} := (x_{j+1} + x_j)/2 = x_j + h/2$ and $y_{j+1/2} \approx y(x_{j+\frac{1}{2}})$.

Second order Runge-Kutta method

Again, consider

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

By the midpoint quadrature rule

$$y(x_{j+1}) - y(x_j) \approx hf(x_{j+\frac{1}{2}}, y(x_{j+\frac{1}{2}})).$$

This gives the implicit midpoint method

$$y_{j+1} = y_j + hf(x_{j+1/2}, y_{j+1/2}), \;\; j = 0 : n-1,$$

where $x_{j+1/2} := (x_{j+1} + x_j)/2 = x_j + h/2$ and $y_{j+1/2} \approx y(x_{j+\frac{1}{2}})$.

To obtain an explicit method, we estimate $y_{j+\frac{1}{2}}$ by forward Euler's method, which yields a second order Runge-Kutta (RK) method

Second order Runge-Kutta method

Again, consider

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

By the midpoint quadrature rule

$$y(x_{j+1}) - y(x_j) \approx hf(x_{j+\frac{1}{2}}, y(x_{j+\frac{1}{2}})).$$

This gives the implicit midpoint method

$$y_{j+1} = y_j + hf(x_{j+1/2}, y_{j+1/2}), \ \ j = 0 : n-1,$$

where $x_{j+1/2} := (x_{j+1} + x_j)/2 = x_j + h/2$ and $y_{j+1/2} \approx y(x_{j+\frac{1}{2}})$.

To obtain an explicit method, we estimate $y_{j+\frac{1}{2}}$ by forward Euler's method, which yields a second order Runge-Kutta (RK) method

$$\begin{aligned} y_{j+\frac{1}{2}} &= y_j + \frac{h}{2} f(x_j, y_j) \\ y_{j+1} &= y_j + hf(x_{j+1/2}, y_{j+\frac{1}{2}}), \ \ j = 0 : n-1. \end{aligned}$$

Second order Runge-Kutta: Example

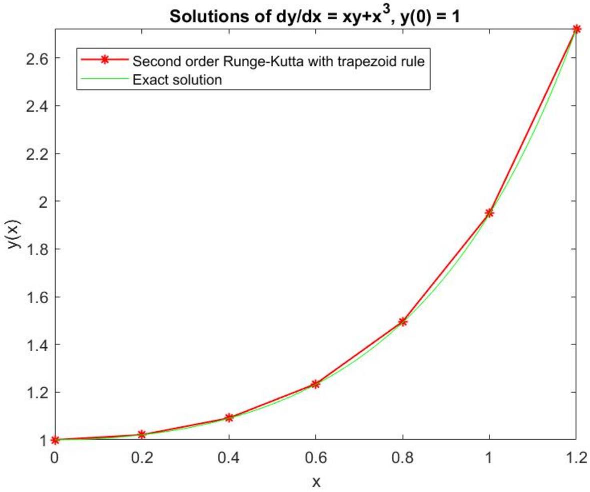

Figure : Exact solution $y(x) = 3e^{x^2/2} - x^2 - 2$ of the non-autonomous ODE $\frac{dy}{dx} = xy + x^3$ satisfying $y(0) = 1$ along with solution via Second order Runge Kutta method with trapezoid rule and $h = 0.2$.

Second order Runge-Kutta: Example

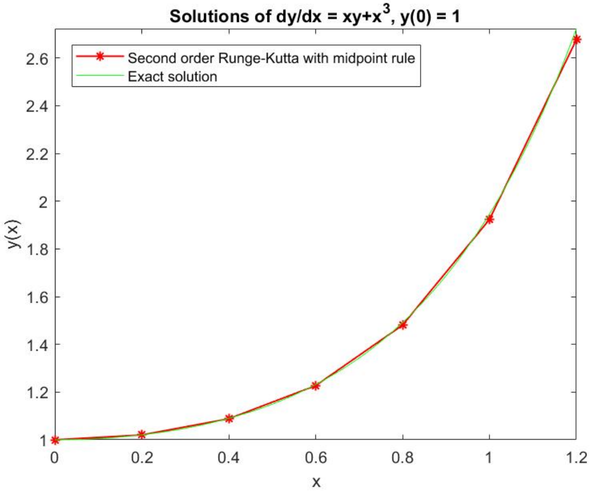

Figure : Exact solution $y(x) = 3e^{x^2/2} - x^2 - 2$ of the non-autonomous ODE $\frac{dy}{dx} = xy + x^3$ satisfying $y(0) = 1$ along with solution via Second order Runge Kutta method with midpoint rule and $h = 0.2$.

Fourth order Runge-Kutta methods

Finally, consider again

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

Fourth order Runge-Kutta methods

Finally, consider again

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

Then Simpson quadrature rule together with four stages of forward Euler's method yields the 4th order RK method

Fourth order Runge-Kutta methods

Finally, consider again

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

Then Simpson quadrature rule together with four stages of forward Euler's method yields the 4th order RK method

$$\begin{aligned}
Y_1 &= y_j \\
Y_2 &= y_j + \frac{h}{2} f(x_j, Y_1) \\
Y_3 &= y_j + \frac{h}{2} f(x_{j+1/2}, Y_2) \\
Y_4 &= y_j + hf(x_{j+1/2}, Y_3) \\
y_{j+1} &= y_j + \frac{h}{6}[f(x_j, Y_1) + 2f(x_{j+1/2}, Y_2) + 2f(x_{j+1/2}, Y_3) + f(x_{j+1}, Y_4)]
\end{aligned}$$

for $j = 0 : n - 1$.

Fourth order Runge-Kutta methods

Finally, consider again

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

Then Simpson quadrature rule together with four stages of forward Euler's method yields the 4th order RK method

$$\begin{aligned}
Y_1 &= y_j \\
Y_2 &= y_j + \frac{h}{2} f(x_j, Y_1) \\
Y_3 &= y_j + \frac{h}{2} f(x_{j+1/2}, Y_2) \\
Y_4 &= y_j + hf(x_{j+1/2}, Y_3) \\
y_{j+1} &= y_j + \frac{h}{6}[f(x_j, Y_1) + 2f(x_{j+1/2}, Y_2) + 2f(x_{j+1/2}, Y_3) + f(x_{j+1}, Y_4)]
\end{aligned}$$

for $j = 0 : n-1$. The 4th order RK method achieves $\mathcal{O}(h^4)$ accuracy and is the best method.

Fourth order Runge-Kutta methods

Finally, consider again

$$y(x_{j+1}) - y(x_j) = \int_{x_j}^{x_{j+1}} f(t, y(t))dt.$$

Then Simpson quadrature rule together with four stages of forward Euler's method yields the 4th order RK method

$$\begin{aligned}
Y_1 &= y_j \\
Y_2 &= y_j + \frac{h}{2} f(x_j, Y_1) \\
Y_3 &= y_j + \frac{h}{2} f(x_{j+1/2}, Y_2) \\
Y_4 &= y_j + hf(x_{j+1/2}, Y_3) \\
y_{j+1} &= y_j + \frac{h}{6}[f(x_j, Y_1) + 2f(x_{j+1/2}, Y_2) + 2f(x_{j+1/2}, Y_3) + f(x_{j+1}, Y_4)]
\end{aligned}$$

for $j = 0 : n-1$. The 4th order RK method achieves $\mathcal{O}(h^4)$ accuracy and is the best method. MATLAB ode45 command implements 4th order RK method.

Fourth order Runge-Kutta: Example

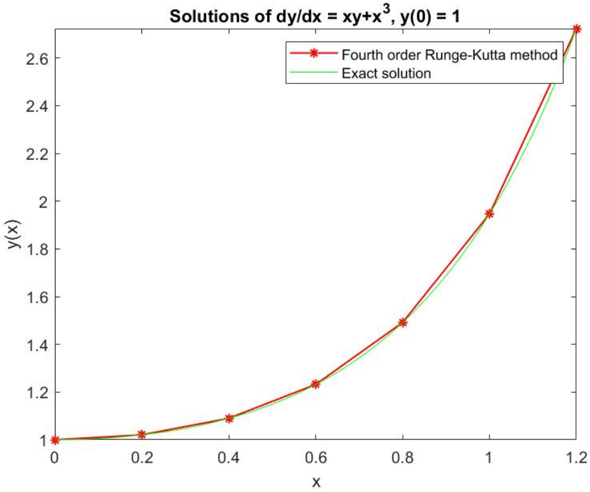

Figure : Exact solution $y(x) = 3e^{x^2/2} - x^2 - 2$ of the non-autonomous ODE $\frac{dy}{dx} = xy + x^3$ satisfying $y(0) = 1$ along with solution via Fourth order Runge Kutta method and $h = 0.2$.

Taylor's Method

Suppose that the solution $y(x)$ of the IVP $\frac{dy}{dx} = f(x, y),\ y(x_0) = y_0$ belongs to C^{k+1}.

By Taylor's expansion,

$$y(x_i + h) = y(x_i) + hy'(x_i) + \cdots + \frac{h^k}{k!}y^{(k)}(x_i) + \frac{h^{k+1}}{(k+1)!}y^{(k+1)}(\xi)$$

for some $x_i < \xi < x_i + h$.

Taylor's Method

Suppose that the solution $y(x)$ of the IVP $\frac{dy}{dx} = f(x, y),\ y(x_0) = y_0$ belongs to C^{k+1}.

By Taylor's expansion,

$$y(x_i + h) = y(x_i) + hy'(x_i) + \cdots + \frac{h^k}{k!}y^{(k)}(x_i) + \frac{h^{k+1}}{(k+1)!}y^{(k+1)}(\xi)$$

for some $x_i < \xi < x_i + h$.

Now,

$$y'(x) = f(x, y) \Rightarrow y^{(j)}(x) = \frac{d^{(j-1)}f(x, y)}{dx^{j-1}} := f^{(j-1)}(x, y),\ j = 2 : k+1.$$

Therefore,

$$\begin{aligned} y(x_i + h) = y(x_i) + hf(x_i, y(x_i)) + \cdots \quad &+ \quad \frac{h^k}{k!}f^{(k-1)}(x(i), y(x_i)) \\ &+ \quad \frac{h^{k+1}}{(k+1)!}f^{(k)}(\xi, y(\xi)) \end{aligned}$$

Taylor's Method

Taylor's Method of order k:

$$y_{i+1} = y_i + hf(x_i, y_i) + \cdots + \frac{h^k}{k!} f^{(k-1)}(x_i, y_i), \quad \text{for } i = 0 : n.$$

For $k = 1$, this is Forward Euler Method $y_{i+1} = y_i + hf(x_i, y_i), \; i = 0 : n.$

Taylor's Method

Taylor's Method of order k:

$$y_{i+1} = y_i + hf(x_i, y_i) + \cdots + \frac{h^k}{k!} f^{(k-1)}(x_i, y_i), \quad \text{for } i = 0 : n.$$

For $k = 1$, this is Forward Euler Method $y_{i+1} = y_i + hf(x_i, y_i),\ i = 0 : n.$

For $k = 2$, $y_{i+1} = y_i + hf(x_i, y_i) + \frac{h^2}{2} f'(x_i, y_i),\ i = 0 : n.$

Taylor's Method

Taylor's Method of order k:

$$y_{i+1} = y_i + hf(x_i, y_i) + \cdots + \frac{h^k}{k!} f^{(k-1)}(x_i, y_i), \quad \text{for } i = 0 : n.$$

For $k = 1$, this is Forward Euler Method $y_{i+1} = y_i + hf(x_i, y_i), \; i = 0 : n.$

For $k = 2$, $y_{i+1} = y_i + hf(x_i, y_i) + \frac{h^2}{2} f'(x_i, y_i), \; i = 0 : n.$

Since

$$f'(x, y) = \frac{df(x, y)}{dx} = f_x(x, y) + f_y(x, y)\frac{dy}{dx} = f_x(x, y) + f_y(x, y)f(x, y),$$

therefore Taylor's Method of order 2 is

$$y_{i+1} = y_i + hf(x_i, y_i) + \frac{h^2}{2}\left(f_x(x_i, y_i) + f_y(x_i, y_i)f(x_i, y_i)\right), \; i = 0 : n,$$

where clearly the error is $\mathcal{O}(h^3)$.

Taylor's Method

Example: For the IVP $\frac{dy}{dx} = xy + x^3, \ y(0) = 1,$

$$f(x, y) = xy + x^3 \Rightarrow f_x(x, y) = y + 3x^2 \text{ and } f_y(x, y) = x.$$

Taylor's Method

Example: For the IVP $\frac{dy}{dx} = xy + x^3, \; y(0) = 1,$

$$f(x, y) = xy + x^3 \Rightarrow f_x(x, y) = y + 3x^2 \text{ and } f_y(x, y) = x.$$

Therefore,

$$f'(x, y) = f_x(x, y) + f_y(x, y)f(x, y) = y + 3x^2 + x^2y + x^4.$$

Taylor's Method

Example: For the IVP $\frac{dy}{dx} = xy + x^3,\ y(0) = 1,$

$$f(x,y) = xy + x^3 \Rightarrow f_x(x,y) = y + 3x^2 \text{ and } f_y(x,y) = x.$$

Therefore,

$$f'(x,y) = f_x(x,y) + f_y(x,y)f(x,y) = y + 3x^2 + x^2y + x^4.$$

Hence we have 2nd order Taylor's method

$$\begin{aligned} y_{i+1} &= y_i + hf(x_i, y_i) + \frac{h^2}{2}f'(x_i, y_i) \\ &= y_i + h(x_i y_i + x_i^3) + \frac{h^2}{2}(y_i + 3x_i^2 + x_i^2 y_i + x_i^4). \end{aligned}$$

for $i = 0, \ldots, n$.

Taylor's Method: Example

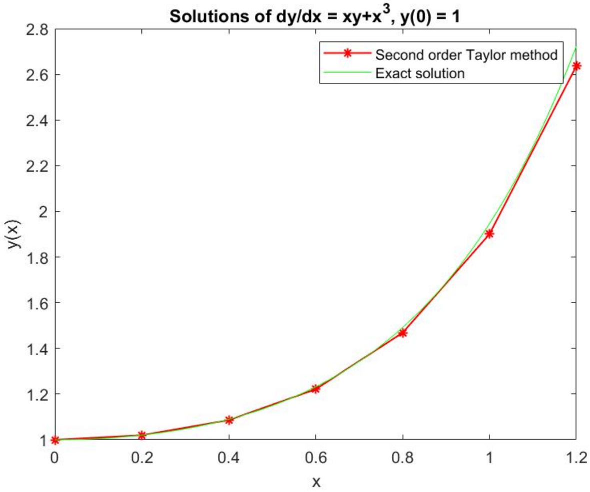

Figure : Exact solution $y(x) = 3e^{x^2/2} - x^2 - 2$ of the non-autonomous ODE $\frac{dy}{dx} = xy + x^3$ satisfying $y(0) = 1$ along with the solution via second order Taylor's method with $h = 0.2$.

Systems of ODE

A first-order system has the form: For $x \in [a, b]$ solve

$$\begin{aligned} y_1' &= f_1(x, y_1, y_2, \ldots, y_n) \\ y_2' &= f_2(x, y_1, y_2, \ldots, y_n) \\ &\vdots \\ y_n' &= f_n(x, y_1, y_2, \ldots, y_n) \end{aligned}$$

Systems of ODE

A first-order system has the form: For $x \in [a, b]$ solve

$$\begin{aligned} y_1' &= f_1(x, y_1, y_2, \ldots, y_n) \\ y_2' &= f_2(x, y_1, y_2, \ldots, y_n) \\ &\vdots \\ y_n' &= f_n(x, y_1, y_2, \ldots, y_n) \end{aligned}$$

Setting $\mathbf{y} := [y_1, \ldots, y_n]^\top$ and defining $\mathbf{f} : [a, b] \times \mathbb{R}^n \longrightarrow \mathbb{R}^n$ by

$$\mathbf{f}(x, \mathbf{y}) := [f_1(x, \mathbf{y}), \ldots, f_n(x, \mathbf{y})]^\top,$$

we have the IVP

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}), \; x \in [a, b] \text{ and } \mathbf{y}(x_0) = \mathbf{y}_0.$$

We can use vector version of Euler method to solve the IVP.

Vector version of Euler method

Consider the system of first order ODE

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}),\ \mathbf{y}(x_0) = \mathbf{y}_0.$$

Forward Euler's method for the system

$$\mathbf{y}_{j+1} = \mathbf{y}_j + h\,\mathbf{f}(x_j, \mathbf{y}_j),\ \ j = 0 : n-1.$$

Backward Euler's method for the system

$$\mathbf{y}_{j+1} = \mathbf{y}_j + h\,\mathbf{f}(x_{j+1}, \mathbf{y}_{j+1}),\ \ j = 0 : n-1.$$

Vector version of Euler method

Consider the system of first order ODE

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}),\ \mathbf{y}(x_0) = \mathbf{y}_0.$$

Forward Euler's method for the system

$$\mathbf{y}_{j+1} = \mathbf{y}_j + h\,\mathbf{f}(x_j, \mathbf{y}_j),\ \ j = 0 : n-1.$$

Backward Euler's method for the system

$$\mathbf{y}_{j+1} = \mathbf{y}_j + h\,\mathbf{f}(x_{j+1}, \mathbf{y}_{j+1}),\ \ j = 0 : n-1.$$

At each step of backward Euler, we have to solve the nonlinear system $\mathbf{y}_{j+1} - \mathbf{y}_j - h\,\mathbf{f}(x_{j+1}, \mathbf{y}_{j+1}) = 0$ for $\mathbf{y}_{j+1}$.

Vector version of Euler method

Example: $\begin{bmatrix} y_1' \\ y_2' \end{bmatrix} = \begin{bmatrix} y_2^2 - 2y_1 \\ y_1 - y_2 - xy_2^2 \end{bmatrix}, \begin{bmatrix} y_1(0) \\ y_2(0) \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \Longrightarrow \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} xe^{-2x} \\ e^{-x} \end{bmatrix}.$

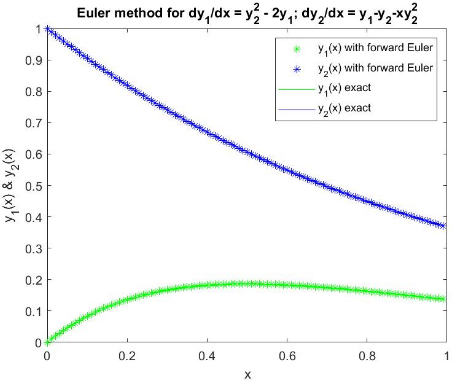

Figure : Exact solutions $y_1(x) = xe^{2x}$ and $y_2(x) = e^{-x}$ of the first order system of ODEs in Example 1 and their respective approximations via forward Euler method with $h = 0.01$.

Higher order equation

Higher order ODE can often be converted to a system of first order ODE. Consider the n-th order ODE

$$y^{(n)} = f(x, y, y', \ldots, y^{(n-1)}), \quad y^{(j)}(x_0) = \alpha_j, \; j = 0 : n-1.$$

Higher order equation

Higher order ODE can often be converted to a system of first order ODE. Consider the n-th order ODE

$$y^{(n)} = f(x, y, y', \ldots, y^{(n-1)}), \;\; y^{(j)}(x_0) = \alpha_j, \; j = 0 : n-1.$$

Introducing new variable $y_1 := y, y_2 := y', \ldots, y_n := y^{(n-1)}$, we have

$$\mathbf{y}' := \begin{bmatrix} y_1' \\ \vdots \\ y_{n-1}' \\ y_n' \end{bmatrix} = \begin{bmatrix} y_2 \\ \vdots \\ y_n \\ f(x, y_1, \ldots, y_n) \end{bmatrix} =: \mathbf{f}(x, \mathbf{y}), \;\; \mathbf{y}'(x_0) = \mathbf{y}_0,$$

where $\mathbf{y} := [y_1, \ldots, y_n]^\top$ and $\mathbf{y}_0 := [\alpha_0, \ldots, \alpha_{n-1}]^\top$.

Motion of a pendulum

The motion of a pendulum of length ℓ is governed by the equation

$$y'' = -\frac{g}{\ell}\sin(y), \; y(0) = a, \; y'(0) = b,$$

where y is angle measured in radian and g is gravity.

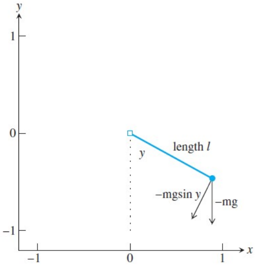

Figure : The component of force along the tangential direction is $F = -mg\sin y$ where y is the angle made by the pendulum bob with the vertical axis.

Motion of a pendulum

Setting $y_1 = y$ and $y_2 = y'$, we have the first order system

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= -\frac{g}{\ell}\sin(y_1) \end{aligned}$$

Motion of a pendulum

Setting $y_1 = y$ and $y_2 = y'$, we have the first order system

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= -\frac{g}{\ell}\sin(y_1) \end{aligned}$$

If the pendulum is started from a position straight out to the right then the initial conditions are $y_1(0) = \pi/2$ and $y_2(0) = 0$. Now, considering $\ell = 1$ and $g = 9.81 m/sec^2$, we can test the suitability of Eulers Method as a solver for this system.

Motion of a pendulum

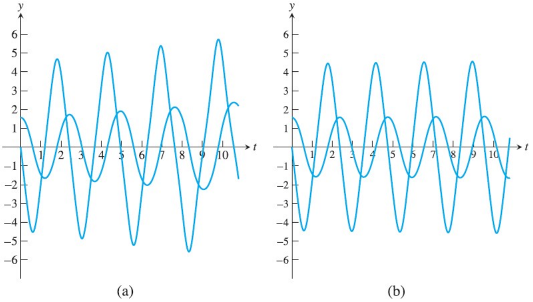

Figure : Euler method for the motion of the pendulum. Smaller oscillations plot angle y_1 and larger oscillations plot angular velocity y_2. (a) h $= 0.01$ is too large showing growing amplitude of oscillations (b) h $= 0.001$ is more accurate.

Orbit of a satellite

Let (x, y) denote the position of a satellite. Then Newton's law of motion yields the second order system of ODE

$$\begin{aligned} m_1 x'' &= -\frac{g m_1 m_2 x}{(x^2 + y^2)^{3/2}} \\ m_1 y'' &= -\frac{g m_1 m_2 y}{(x^2 + y^2)^{3/2}} \end{aligned}$$

To transform it to a system of first order ODEs, let $v_x = x'$ and $v_y = y'$.

Orbit of a satellite

Let (x, y) denote the position of a satellite. Then Newton's law of motion yields the second order system of ODE

$$\begin{aligned} m_1 x'' &= -\frac{g m_1 m_2 x}{(x^2+y^2)^{3/2}} \\ m_1 y'' &= -\frac{g m_1 m_2 y}{(x^2+y^2)^{3/2}} \end{aligned}$$

To transform it to a system of first order ODEs, let $v_x = x'$ and $v_y = y'$. Then the system is rewritten as

$$\begin{aligned} x' &= v_x \\ v_x' &= -\frac{g m_2 x}{(x^2+y^2)^{\frac{3}{2}}} \\ y' &= y_y \\ v_y' &= -\frac{g m_2 y}{(x^2+y^2)^{\frac{3}{2}}} \end{aligned}$$

Orbit of a satellite

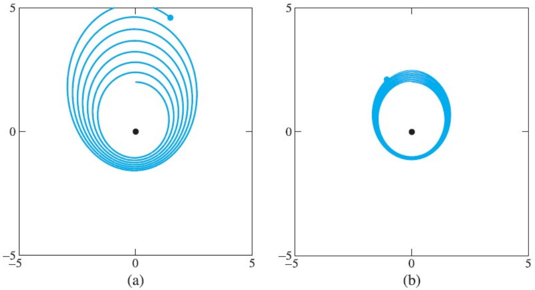

Figure : The one body problem approximated with forward Euler method (a) h = 0.01 (b) h = 0.001.

Orbit of a satellite

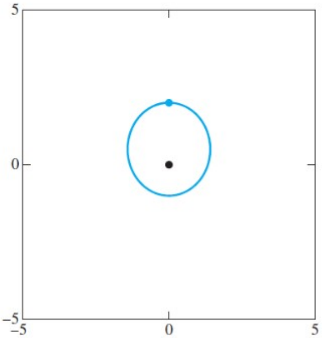

Figure : The one body problem approximated with RK2 using trapezoid rule with h = 0.01.