EAD 115

Numerical Solution of Engineering and Scientific Problems

David M. Rocke
Department of Applied Science
Ordinary Differential Equations

• ODE: solve for functions of one variable
• Possibly multiple equations and multiple functions, but usually one equation in one variable
• Functions of more than one variable can appear in partial differential equations (PDE’s)
• Some ODE’s can be solved analytically, but most cannot
\[ F = ma \]

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

Analytical

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

Numerical

\[ v_{i+1} = v_i + \left( g - \frac{c}{m} v_i \right) \Delta t \]

Physical law

ODE

Solution
\[ y = -0.5x^4 + 4x^3 - 10x^2 + 8.5x + 1 \]
\[ \frac{dy}{dx} = -2x^3 + 12x^2 - 20x + 8.5 \]

simple form of ODE
\[ y = \int -2x^3 + 12x^2 - 20x + 8.5 \, dx \]
\[ y = -0.5x^4 + 4x^3 - 10x^2 + 8.5x + c \]

another condition is needed to determine \( y \)
\[ x = 0, \ y = 1 \Rightarrow c = 1 \]

initial condition

often has a physical interpretation
Initial/Boundary Value Problem

• An *initial value problem* is an ODE in which the specifications that make the solution unique occur at a single value of the independent variable x or t
• A *boundary value problem* specifies the conditions at a number of different x or t values
Consider an ODE of the form
\[
\frac{dy}{dx} = f(x, y) \quad \text{with initial conditions}
\]
We can trace out a solution starting at \((x_0, y_0)\)
\[
y_{i+1} = y_i + \phi h
\]
where
\[
x_{i+1} - x_i = h
\]
\[
\phi = \frac{dy}{dx}
\]
The diagram shows a graph with the axes labeled $y$ and $x$. The equation $y_{i+1} = y_i + \phi h$ is indicated, which represents the change in $y$ from $x_i$ to $x_{i+1}$ with a step size $h$.

- The slope at $x_i$ is denoted by $\phi$.
- The step size $h$ is shown between $x_i$ and $x_{i+1}$.
Runge-Kutta methods

• Euler’s method is the simplest of these one-step methods
• Improved slope estimates can improve the result
• These methods are called in general Runge-Kutta or RK methods
\[
\frac{dy}{dx} = f(x, y)
\]

\[
y_{i+1} = y_i + \phi h
\]

\[
\phi = \frac{dy}{dx}\bigg|_{(x_i, y_i)} = f(x_i, y_i)
\]

Euler's method
Integrate between 0 and 4, with $y = 1$ at $x = 0$

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

The exact solution is

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

Euler's method is somewhat silly since we have better methods like Simpson's rule, but serves as an illustration
Errors in Euler’s Method

• Errors are
  – *local* to each step
  – *global* or accumulated

• Errors are caused by
  – truncation (when h is large)
  – roundoff (when h is small and number of steps is large)
Euler's Method Global Error for 5 Step Sizes

- Exact
- $h=0.5$
- $h=0.25$
- $h=0.125$
- $h=0.0625$
- $h=0.03125$

Graph showing the global error for different step sizes in Euler's method.
Euler's Method Local Error for 5 Step Sizes
Euler’s Method

• Is simple to implement
• Can be sufficiently accurate for many practical tasks if the step size is small enough
• No step size will result in a highly accurate result
• Higher order methods are needed
set integration range
xi = 0
xf = 4
initialize variables
x = xi
y = 1
set step size and determine
number of calculation steps
dx = 0.5
nc = (xf - xi)/dx
output initial condition
PRINT x, y
loop to implement Euler’s method
and display results
DO i = 1, nc
dydx = -2x^3 + 12x^2 - 20x + 8.5
y = y + dydx * dx
x = x + dx
PRINT x, y
END DO
(a) Main or “Driver” Program

Assign values for
y = initial value dependent variable
xi = initial value independent variable
xf = final value independent variable
dx = calculation step size
xout = output interval

x = xi
m = 0
xp_m = x
yp_m = y
DO
xend = x + xout
IF (xend > xf) THEN xend = xf
h = dx
CALL Integrator (x, y, h, xend)
m = m + 1
xp_m = x
yp_m = y
IF (x ≥ xf) EXIT
END DO
DISPLAY RESULTS
END

(b) Routine to Take One Output Step

SUB Integrator (x, y, h, xend)
  DO
    IF (xend − x < h) THEN h = xend − x
    CALL Euler (x, y, h, ynew)
    y = ynew
    IF (x ≥ xend) EXIT
  END DO
END SUB

(c) Euler’s Method for a Single ODE

SUB Euler (x, y, h, ynew)
  CALL Derivs(x, y, dydx)
  ynew = y + dydx * h
  x = x + h
END SUB

(d) Routine to Determine Derivative

SUB Derivs (x, y, dydx)
dydx = ...
END SUB
Prediction is very difficult, especially about the future."

-- Niels Bohr
One of the Mississippi's oddest peculiarities is that of shortening its length from time to time. The Mississippi between Cairo and New Orleans was twelve hundred and fifteen miles long one hundred and seventy-six years ago. It was eleven hundred and eighty after the cut-off of 1722. It was one thousand and forty after the American Bend cut-off. It has lost sixty-seven miles since. Consequently its length is only nine hundred and seventy-three miles at present.
In the space of one hundred and seventy-six years the Lower Mississippi has shortened itself two hundred and forty-two miles. This is an average of a trifle over one mile and a third per year. Therefore, any calm person, who is not blind or idiotic, can see that in the Old Oolitic Silurian Period, just a million years ago next November, the Lower Mississippi River was upward of one million three hundred thousand miles long, and stuck out over the Gulf of Mexico like a fishing-rod. And by the same token any person can see that seven hundred and forty-two years from now the Lower Mississippi will be only a mile and three-quarters long, and Cairo and New Orleans will have joined their streets together, and be plodding comfortably along under a single mayor and a mutual board of aldermen.
There is something fascinating about science. One gets such wholesale returns of conjecture out of such a trifling investment of fact.

—Mark Twain, Life on the Mississippi 173-6 (1883)
Improvements in Euler’s Method

• We could use a higher order Taylor expansion at the current iterate to reduce truncation error
• This results in more analytical complexity due to the need for more derivatives
• Mostly, alternative methods are used to make the extrapolation more accurate
• Extrapolation is a hazardous business!
\[ \frac{dy}{dx} = f(x, y) \]

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

\[ y_{i+1} \approx y_i + f(x_i, y_i)h + \frac{1}{2} \frac{\partial}{\partial x} f(x_i, y_i)h^2 \]

\[ y_{i+1} \approx y_i + f(x_i, y_i)h + \frac{1}{2} \frac{\partial}{\partial x} f(x_i, y_i)h^2 + \frac{1}{6} \frac{\partial^2}{\partial x^2} f(x_i, y_i)h^3 \]

\[ \frac{\partial}{\partial x} f(x, y) = \frac{\partial f(x, y)}{\partial x} + \frac{\partial f(x, y)}{\partial y} \frac{dy}{dx} \]

\[ \frac{\partial^2}{\partial x^2} f(x, y) = \left[ \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial x \partial y} \frac{dy}{dx} \right] \]

\[ + \left[ \frac{\partial^2 f(x, y)}{\partial x \partial y} \frac{dy}{dx} + \frac{\partial^2 f(x, y)}{\partial y^2} \left( \frac{dy}{dx} \right)^2 + \frac{\partial f(x, y)}{\partial y} \frac{d^2 y}{dx^2} \right] \]

\[ = \left[ \frac{\partial^2 f(x, y)}{\partial x^2} + 2 \frac{\partial^2 f(x, y)}{\partial x \partial y} \frac{dy}{dx} \right] + \frac{\partial^2 f(x, y)}{\partial y^2} \left( \frac{dy}{dx} \right)^2 + \frac{\partial f(x, y)}{\partial y} \frac{d^2 y}{dx^2} \]
Heun’s Method

• One problem with Euler’s method is that it uses the derivative at the beginning of the interval to predict the change within the interval.

• Heun’s method uses a better estimate of the change, which is closer to the average derivative in the interval, rather than the initial derivative.

• It is one of a class of predictor-corrector methods.
Slope $= f(x_{i+1}, y_{i+1}^0)$

Slope $= f(x_i, y_i)$

Slope $= \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^0)}{2}$
\[ y'_{i} = f(x_i, y_i) \]
\[ y_{i+1}^0 = y_i + f(x_i, y_i)h \quad \text{Euler Step Predictor Equation} \]
\[ y_{i+1}' = f(x_{i+1}, y_{i+1}^0) \]
\[ y_{i+1}' = \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^0)}{2} \]
\[ y_{i+1} = y_i + \left( \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^0)}{2} \right)h \quad \text{Corrector Equation} \]

Can be iterated
Integrate

\[ y' = 4e^{0.8x} - 0.5y \quad \text{from } x = 0 \text{ to } x = 4 \text{ with stepsize } = 1 \]

\[ x = 0 \text{ then } y = 2 \]

Analytical

\[ y = ae^{0.8x} + be^{-0.5x} \]

\[ y' = 0.8ae^{0.8x} - 0.5be^{-0.5x} \]

\[ 0 = y' - 4e^{0.8x} + 0.5y = (0.8ae^{0.8x} - 0.5be^{-0.5x}) - 4e^{0.8x} + (0.5ae^{0.8x} + 0.5be^{-0.5x}) \]

\[ = (0.8a - 4 + 0.5a)e^{0.8x} \]

\[ a = 4/1.3 \]

\[ a + b = 2 \]

\[ y = \frac{4}{1.3}(e^{0.8x} - e^{-0.5x}) + 2e^{-0.5x} \]
\[ y' = 4e^{0.8x} - 0.5y \quad \text{from } x = 0 \text{ to } x = 4 \text{ with stepsize } = 1 \]

\[ x = 0 \text{ then } y = 2 \]

\[ x_0 = 0 \quad y_0 = 2 \quad y'_0 = 4 - 1 = 3 \]

\[ y_1^0 = 2 + 3(1) = 5 \quad \text{(true value at } x = 1 \text{ is } 6.1946) \]

\[ \varepsilon_t = |6.1946 - 5| / 6.1946 = 0.193 \]

\[ y'_1 = 4e^{0.8} - 0.5(5) = 6.4022 \]

\[ \overline{y}' = (3 + 6.4022) / 2 = 4.7011 \]

\[ y_1^1 = 2 + (4.7011)(1) = 6.7010 \]

\[ \varepsilon_t = |6.1946 - 6.7010| / 6.1946 = 0.082 \]
\[ y' = 4e^{0.8x} - 0.5y \quad \text{from} \quad x = 0 \quad \text{to} \quad x = 4 \quad \text{with stepsize} = 1 \]

\[ x = 0 \quad \text{then} \quad y = 2 \]

\[ x_0 = 0 \quad \quad y_0 = 2 \quad \quad y'_0 = 4 - 1 = 3 \]

\[ y'_0 = 2 + 3(1) = 5 \quad \quad \text{(true value at} \quad x = 1 \quad \text{is} \quad 6.1946) \]

\[ y'_1 = 2 + (4.7011)(1) = 6.7010 \]

\[ y'_1 = 4e^{0.8} - 0.5(6.7010) = 5.5517 \]

\[ y'_1 = (3 + 5.5517) / 2 = 4.2758 \]

\[ y_1 = 2 + (4.2758)(1) = 6.2758 \]

\[ \varepsilon_t = |6.1946 - 6.2758| / 6.1946 = 0.013 \]
• This will not, in general converge upon iteration to the true value of $y_{i+1}$
• This is because we are at best estimating the actual slope of the secant by the average of the slopes at the two ends, and even were the slopes at the two ends exact, this is not an identity
Integrate between 0 and 4, with \( y = 1 \) at \( x = 0 \)

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

\( x_0 = 0 \quad y_0 = 1 \quad h = 0.5 \)

Euler's Method

\[
x_1 = 0.5 \quad y'_0 = 8.5 \quad y_1^0 = 1 + (8.5)(0.5) = 5.25
\]

Heun's Method

\[
y'_1 = 1.25 \quad \bar{y}'_1 = 4.875 \quad y_1^1 = 1 + (4.875)(0.5) = 3.4375
\]

No iteration needed

(True value is 3.00)
Euler's Method for 5 Step Sizes
Heun's Method for 5 Step Sizes

- Exact
- $h=0.5$
- $h=0.25$
- $h=0.125$
- $h=0.0625$
- $h=0.03125$
Euler's and Heun's Method

- **Exact**
- **h=0.5 Euler**
- **h=0.25 Euler**
- **h=0.5 Heun**
- **h=0.25 Heun**
Euler's Method Global Error for 5 Step Sizes

- **Exact**
- **h=0.5**
- **h=0.25**
- **h=0.125**
- **h=0.0625**
- **h=0.03125**

**Global Error** vs. **x**
Euler's Method Local Error for 5 Step Sizes

- **Exact**
- **h=0.5**
- **h=0.25**
- **h=0.125**
- **h=0.0625**
- **h=0.03125**
Heun's Method Local Error for 5 Step Sizes
Midpoint Method

• Euler’s method approximates the slope of the secant between two points by the slope at the left end of the interval.
• Heun’s method approximates it by the average of the estimated slopes at the endpoints.
• The midpoint method approximates it by the estimated slope at the average of the endpoints.
Integrate by midpoint method

\[ y' = 4e^{0.8x} - 0.5y \quad \text{from } x = 0 \text{ to } x = 4 \text{ with stepsize } = 1 \]

\[ x = 0 \text{ then } y = 2 \]

\[ x_0 = 0 \quad y_0 = 2 \quad y'_0 = 4 - 1 = 3 \]

\[ y_{1/2} = 2 + 3(1/2) = 3.5 \quad \text{(true value at } x = 1/2 \text{ is } 3.7515) \]

\[ y'_{1/2} = 4e^{0.8/2} - 0.5(3.5) = 4.2173 \]

\[ y_1 = 2 + (4.2173)(1) = 6.2173 \]

\[ \varepsilon_t = \left| 6.1946 - 6.2173 \right| / 6.1946 \]

\[ = .02267 / 6.1946 = 0.0366 \]
\[ y'_i = f(x_i) \]
\[ y_{i+1}^0 = y_i + f(x_i)h \quad \text{Euler} \]
\[ y'_{i+1} = f(x_{i+1}) \]
\[ \overline{y}'_i = \frac{f(x_i) + f(x_{i+1})}{2} \]
\[ y_{i+1} = y_i + \left(\frac{f(x_i) + f(x_{i+1})}{2}\right)h \quad \text{Heun} \]
\[ y_{i+1} - y_i = \left(\frac{f(x_i) + f(x_{i+1})}{2}\right)h \quad \text{Trapezoid Rule} \]
\[ y'_i = f(x_i) \]
\[ y^0_{i+1} = y_i + f(x_i)h \]
Euler = Riemann left
\[ y'_{i+1/2} = f(x_{i+1/2}) \]
\[ y_{i+1} = y_i + f(x_{i+1/2})h \]
Midpoint = Riemann midpoint
\[ y_{i+1} - y_i = f(x_{i+1/2})h \]
Integrate between 0 and 4, with \( y = 1 \) at \( x = 0 \)

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

\( x_0 = 0 \quad y_0 = 1 \quad h = 0.5 \)

Euler's Method

\( x_1 = 0.5 \quad y'_0 = 8.5 \quad y^0_1 = 1 + (8.5)(0.5) = 5.25 \)

Heun's Method

\( y'_1 = 1.25 \quad \bar{y}'_1 = 4.875 \quad y^1_1 = 1 + (4.875)(0.5) = 3.4375 \)

Midpoint Method

\( y'_{1/2} = 4.21875 \quad y_1 = 1 + (4.21875)(.5) = 3.109375 \)

(True value is 3.00)
$h = 0.25, 0.125, 0.0625$
Error Analysis

• Euler’s method integrates exactly over an interval so long as the derivative at the beginning is the same as the slope of the secant line.

• This requires the derivative to be constant.

• $y = f(x) = ax + b$ fulfills this requirement. The function must be linear.

• If $f(x)$ is quadratic, then Heun’s method and the midpoint method are exact.
\[ f(x) = ax^2 + bx + c \]

\[ f(x_1) - f(x_0) = (ax_1^2 + bx_1 + c) - (ax_0^2 + bx_0 + c) \]

\[ = a(x_1^2 - x_0^2) + b(x_1 - x_0) \]

\[ = (x_1 - x_0)[a(x_1 + x_0) + b] \]

\[ \frac{f(x_1) - f(x_0)}{x_1 - x_0} = a(x_1 + x_0) + b \]

\[ \frac{f'(x_1) + f'(x_0)}{2} = \frac{2ax_1 + b + 2ax_0 + b}{2} = a(x_1 + x_0) + b \]

\[ f'\left((x_1 + x_0)/2\right) = 2a(x_1 + x_0)/2 + b = a(x_1 + x_0) + b \]
Error Analysis

• If the function $f(x)$ is approximated by a Taylor series, then Euler’s method is exact on the first-order term, so the local error is $O(h^2)$

• Heun’s method and the midpoint method are exact on the second-order approximation, so the local error is $O(h^3)$

• Since we are integrating $O(h)$ intervals, the global error is $O(h)$ for Euler and $O(h^2)$ for Heun and the midpoint method
(a) Main or “Driver” Program

Assign values for
y = initial value dependent variable
xi = initial value independent variable
xf = final value independent variable
dx = calculation step size
xout = output interval

x = xi
m = 0
xp_m = x
yp_m = y
DO
xend = x + xout
IF (xend > xf) THEN xend = xf
h = dx
CALL Integrator (x, y, h, xend)
m = m + 1
xp_m = x
yp_m = y
IF (x ≥ xf) EXIT
END DO
DISPLAY RESULTS
END

(b) Routine to Take One Output Step

SUB Integrator (x, y, h, xend)
   DO
      IF (xend - x < h) THEN h = xend - x
      CALL Euler (x, y, h, ynew)
y = ynew
      IF (x ≥ xend) EXIT
   END DO
END SUB

(c) Euler’s Method for a Single ODE

SUB Euler (x, y, h, ynew)
   CALL Derivs(x, y, dydx)
ynew = y + dydx * h
   x = x + h
END SUB

(d) Routine to Determine Derivative

SUB Derivs (x, y, dydx)
dydx = ...
END SUB
(a) Simple Heun without Corrector

SUB Heun (x, y, h, ynew)
   CALL Derivs (x, y, dy1dx)
   ye = y + dy1dx \cdot h
   CALL Derivs(x + h, ye, dy2dx)
   Slope = (dy1dx + dy2dx)/2
   ynew = y + Slope \cdot h
   x = x + h
END SUB

(b) Midpoint Method

SUB Midpoint (x, y, h, ynew)
   CALL Derivs(x, y, dydx)
   ym = y + dydx \cdot h/2
   CALL Derivs (x + h/2, ym, dymdx)
   ynew = y + dymdx \cdot h
   x = x + h
END SUB

(c) Heun with Corrector

SUB HeunIter (x, y, h, ynew)
   es = 0.01
   maxit = 20
   CALL Derivs(x, y, dy1dx)
   ye = y + dy1dx \cdot h
   iter = 0
   DO
      yeold = ye
      CALL Derivs(x + h, ye, dy2dx)
      slope = (dy1dx + dy2dx)/2
      ye = y + slope \cdot h
      iter = iter + 1
      ea = \left| \frac{ye - yeold}{ye} \right| \times 100\%
      IF (ea \leq es OR iter > maxit) EXIT
   END DO
   ynew = ye
   x = x + h
END SUB
General Runge-Kutta Methods

• Achieve accuracy of higher order Taylor series expansions without having to compute additional terms explicitly.

• Use the same general formulation as Euler’s method, Heun’s method, and the midpoint method in which the next point is the previous point plus the stepsize times an estimate of the slope.
\[ y' = f(x, y) \]
\[ y_{i+1} = y_i + \varphi(x_i, y_i, h)h \]
\[ \varphi = a_1 k_1 + a_2 k_2 + \cdots + a_n k_n \]
\[ k_1 = f(x_i, y_i) \]
\[ k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h) \]
\[ k_3 = f(x_i + p_2 h, y_i + q_{21} k_1 h + q_{22} k_2 h) \]
\[ \vdots \]
\[ k_n = f(x_i + p_{n-1} h, y_i + q_{n-11} k_1 h + \cdots + q_{n-1n-1} k_{n-1} h) \]
\[ n = 1 \quad \text{Euler's method} \]
\[ n = 2 \quad \text{Heun/Midpoint method} \]
\[ p_1 = 1 \quad q_{11} = 1 \quad a_1 = 1/2 \quad a_2 = 1/2 \]
\[ p_1 = 1/2 \quad q_{11} = 1/2 \quad a_1 = 0 \quad a_2 = 1 \]
Second-Order Runge-Kutta

\[ y' = f(x, y) \]

\[ y_{i+1} = y_i + \varphi(x_i, y_i, h)h \]

\[ \varphi = a_1 k_1 + a_2 k_2 \]

\[ k_1 = f(x_i, y_i) \]

\[ k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h) \]

\[ y_{i+1} = y_i + (a_1 k_1 + a_2 k_2)h \]

\[ y_{i+1} = y_i + f(x_i, y_i)h + \frac{1}{2} \frac{d}{dx} f(x_i, y_i)h^2 \]

\[ \frac{d}{dx} f(x, y) = \frac{\partial f(x, y)}{\partial x} + \frac{\partial f(x, y)}{\partial y} \frac{dy}{dx} \]

\[ y_{i+1} = y_i + f(x_i, y_i)h + \frac{1}{2} \left( \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} \frac{dy}{dx} \right)h^2 \]
\[ y_{i+1} = y_i + (a_1 k_1 + a_2 k_2) h \]
\[ y_{i+1} = y_i + f(x_i, y_i) h + \frac{1}{2} \left( \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} \right) h^2 \]
\[ k_1 = f(x_i, y_i) \]
\[ k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h) \]
\[ y_{i+1} = y_i + a_1 h f(x_i, y_i) + a_2 h f(x_i, y_i) + a_2 h^2 p_1 \frac{\partial f}{\partial x} + a_2 h^2 q_{11} f(x_i, y_i) \frac{\partial f}{\partial y} \]
\[ = y_i + \left[ a_1 f(x_i, y_i) + a_2 f(x_i, y_i) \right] h + \left[ a_2 p_1 \frac{\partial f}{\partial x} + a_2 q_{11} f(x_i, y_i) \frac{\partial f}{\partial y} \right] h^2 \]
\[ 1 = a_1 + a_2 = 2a_2 p_1 = 2a_2 q_{11} \]
\[ a_1 = 1 - a_2 \quad p_1 = q_{11} = 1/2a_2 \]
\[ y_{i+1} = y_i + (a_1 k_1 + a_2 k_2) h \]

\[ k_1 = f(x_i, y_i) \]

\[ k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h) \]

\[ a_1 = 1 - a_2 \quad p_1 = q_{11} = 1 / 2 a_2 \]

\[ a_2 = 1 / 2 \Rightarrow a_1 = 1 / 2 \quad p_1 = q_{11} = 1 \]

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

Heun's Method
\[ y_{i+1} = y_i + (a_1k_1 + a_2k_2)h \]
\[ k_1 = f(x_i, y_i) \]
\[ k_2 = f(x_i + p_1h, y_i + q_{11}k_1h) \]
\[ a_1 = 1 - a_2 \quad p_1 = q_{11} = 1/2a_2 \]
\[ a_2 = 1 \Rightarrow a_1 = 0 \quad p_1 = q_{11} = 1/2 \]
\[ y_{i+1} = y_i + hf(x_i + h/2, y_i + f(x_i, y_i)h/2) \]

Midpoint Method
\[ y_{i+1} = y_i + (a_1 k_1 + a_2 k_2) h \]
\[ k_1 = f(x_i, y_i) \]
\[ k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h) \]
\[ a_1 = 1 - a_2 \quad p_1 = q_{11} = 1/2 a_2 \]
\[ a_2 = 2/3 \implies a_1 = 1/3 \quad p_1 = q_{11} = 3/4 \]
\[ y_{i+1} = y_i + h(f(x_i, y_i) + 2 f(x_i + 3h/4, y_i + 3 f(x_i, y_i)h/4))/3 \]

Ralston's Method
Second-Order Runge-Kutta Methods

![Graph showing comparison of methods with exact solution]

- **Exact**
- **Euler**
- **Heun**
- **Midpoint**
- **Ralston**
Higher-Order Methods

• Euler’s method is RK order 1 and has global error $O(h)$
• Second-order RK methods (Heun, Midpoint, Ralston) have global error $O(h^2)$
• Third-order RK methods have global error $O(h^3)$
• Fourth-order RK methods have global error $O(h^4)$
Derivation of RK Methods

- Second-order RK methods have four constants, and three equations from comparing the Taylor series expansion to the iteration. There is one undetermined constant.

- Third-order methods have six equations with eight undetermined constants, so two are arbitrary.
Third-Order Runge-Kutta

\[ y' = f(x, y) \]

\[ y_{i+1} = y_i + \phi(x_i, y_i, h)h \]

\[ \phi = a_1k_1 + a_2k_2 + a_3k_3 \]

\[ k_1 = f(x_i, y_i) \]

\[ k_2 = f(x_i + p_1h, y_i + q_{11}k_1h) \]

\[ k_3 = f(x_i + p_2h, y_i + q_{21}k_1h + q_{22}k_2h) \]

\[ y_{i+1} = y_i + (a_1k_1 + a_2k_2 + a_3k_3)h \]

\[ y_{i+1} \approx y_i + f(x_i, y_i)h + \frac{1}{2} \frac{d}{dx} f(x_i, y_i)h^2 + \frac{1}{6} \frac{d^2}{dx^2} f(x_i, y_i)h^3 \]

\[ \frac{d}{dx} f(x, y) = \frac{\partial f(x, y)}{\partial x} + \frac{\partial f(x, y)}{\partial y} \frac{dy}{dx} \]

\[ \frac{\partial^2}{\partial x^2} f(x, y) = \left[ \frac{\partial^2 f(x, y)}{\partial x^2} + 2 \frac{\partial^2 f(x, y)}{\partial x \partial y} \frac{dy}{dx} \right] + \frac{\partial^2 f(x, y)}{\partial y^2} \left( \frac{dy}{dx} \right)^2 + \frac{\partial f(x, y)}{\partial y} \frac{d^2 y}{dx^2} \]
Common Third-Order Method

\[ k_1 = f(x_i, y_i) \]

\[ k_2 = f(x_i + h/2, y_i + k_1 h / 2) \]

\[ k_3 = f(x_i + h, y_i - k_1 h + 2k_2 h) \]

\[ y_{i+1} = y_i + (k_1 + 4k_2 + k_3)h / 6 \]

Reduces to Simpson's Rule
Standard Fourth-Order Method

\[ k_1 = f(x_i, y_i) \]
\[ k_2 = f(x_i + h/2, y_i + k_1 h/2) \]
\[ k_3 = f(x_i + h/2, y_i + k_2 h/2) \]
\[ k_4 = f(x_i + h, y_i + k_3 h) \]

\[ y_{i+1} = y_i + (k_1 + 2k_2 + 2k_3 + k_4)h / 6 \]

Reduces to Simpson's Rule
Butcher's Fifth-Order Method

\[ k_1 = f(x_i, y_i) \]

\[ k_2 = f(x_i + h/4, y_i + k_1h/4) \]

\[ k_3 = f(x_i + h/4, y_i + k_1h/8 + k_2h/8) \]

\[ k_4 = f(x_i + h/2, y_i - k_2h/2 + k_3h) \]

\[ k_5 = f(x_i + 3h/4, y_i + 3k_1h/16 + 9k_4h/16) \]

\[ k_6 = f(x_i + h, y_i - 3k_1h/7 + 2k_2h/7 + 12k_3h/7 - 12k_4h/7 + 8k_5h/7) \]

\[ y_{i+1} = y_i + \left(7k_1 + 32k_3 + 12k_4 + 32k_5 + 7k_6\right)h/90 \]
Comparing RK Methods

- Accuracy depends on the step size and the order
- Computational effort is usually measured in function evaluations
- Up to order 4, an order-$m$ RK method requires $m(b-a)/h$ function evaluations
- Butcher’s order 5 method requires $6(b-a)/h$
SUB RK4 (x, y, h, ynew)
   CALL Derivs(x, y, k1)
ym = y + k1 \cdot h/2
   CALL Derivs(x + h/2, ym, k2)
ym = y + k2 \cdot h/2
   CALL Derivs(x + h/2, ym, k3)
ye = y + k3 \cdot h
   CALL Derivs(x + h, ye, k4)
slope = (k1 + 2(k2 + k3) + k4)/6
ynew = y + slope \cdot h
x = x + h
END SUB
Systems of ODE’s

• We track multiple responses $y_1, y_2, \ldots, y_n$, each of which depends on a single variable $x$ and on possibly all of the other responses

• We also need $n$ initial conditions at $x = x_0$
\[
\frac{dy_1}{dx} = f_1(x, y_1, y_2, \ldots, y_n)
\]
\[
\frac{dy_2}{dx} = f_2(x, y_1, y_2, \ldots, y_n)
\]
\[
\vdots
\]
\[
\frac{dy_n}{dx} = f_n(x, y_1, y_2, \ldots, y_n)
\]

Euler's Method

\[
y_j^{(i+1)} = y_j^{(i)} + f_j(x, y_1^{(i)}, y_2^{(i)}, \ldots, y_n^{(i)})h
\]
\[ \frac{dy_1}{dx} = -0.5y_1 \quad \frac{dy_2}{dx} = 4 - 0.3y_2 - 0.1y_1 \]

Integrate $x = 0$ to $x = 2$

with initial values $y_1 = 4, y_2 = 6, h = 0.5$

<table>
<thead>
<tr>
<th>$x$</th>
<th>$Y_1$</th>
<th>$Y_2$</th>
<th>$Y_1'$</th>
<th>$Y_2'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>4</td>
<td>6</td>
<td>-2.0</td>
<td>1.63</td>
</tr>
<tr>
<td>0.5</td>
<td>3</td>
<td>6.9</td>
<td>-1.5</td>
<td>1.4605</td>
</tr>
<tr>
<td>1.0</td>
<td>2.25</td>
<td>7.715</td>
<td>-1.125</td>
<td>1.2977</td>
</tr>
<tr>
<td>1.5</td>
<td>1.6875</td>
<td>8.4453</td>
<td>-0.8438</td>
<td>1.1452</td>
</tr>
<tr>
<td>2.0</td>
<td>1.2656</td>
<td>9.0941</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
RK Methods for ODE Systems

• We describe the common order 4 method.
• First determine slopes at initial value for all variables, this gives a set of \( n k_1 \) values.
• Then use these to estimate a set of functional values at the midpoint and slopes
• Use these to get improved midpoint values and slopes
• Use these to get estimate of value and slope at end
• Combine for final projection
\[
\frac{dy_1}{dx} = -0.5y_1 \quad \frac{dy_2}{dx} = 4 - 0.3y_2 - 0.1y_1
\]

Integrate \( x = 0 \) to \( x = 2 \)

with initial values \( y_1 = 4 \), \( y_2 = 6 \), \( h = 0.5 \)

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

\( k_2 = f(x_i + h/2, y_i + k_1 h/2) \)

\( k_3 = f(x_i + h/2, y_i + k_2 h/2) \)

\( k_4 = f(x_i + h, y_i + k_3 h) \)

\( y_{i+1} = y_i + (k_1 + 2k_2 + 2k_3 + k_4)h/6 \)
\[
\frac{dy_1}{dx} = -0.5 y_1 \quad \frac{dy_2}{dx} = 4 - 0.3 y_2 - 0.1 y_1
\]
\[
x = 0, 2 \quad y_1 = 4 y_2 = 6, \; h = 0.5
\]
\[
k_{1,1} = f_1(0, 4, 6) = -2
\]
\[
k_{1,2} = f_2(0, 4, 6) = 4 - 0.3(6) - 0.1(4) = 1.8
\]
\[
k_{2,1} = f_1(0.25, 4 + (-2)(0.5) / 2, 6 + (1.8)(0.5) / 2)
\]
\[
= f_1(0.25, 3.5, 6.45) = -1.75
\]
\[
k_{2,2} = f_2(0.25, 3.5, 6.45) = 1.715
\]
\[
\frac{dy_1}{dx} = -0.5y_1, \quad \frac{dy_2}{dx} = 4 - 0.3y_2 - 0.1y_1
\]

\[
x = 0.2, \quad y_1 = 4, \quad y_2 = 6, \quad h = 0.5
\]

\[
k_{1,1} = -2, \quad k_{1,2} = 1.8, \quad k_{2,1} = -1.75, \quad k_{2,2} = 1.715
\]

\[
k_{3,1} = f_1(0.25, 4 + (-1.75)(0.5) / 2, 6 + (1.715)(0.5) / 2)
\]

\[
= f_1(0.25, 3.5625, 6.42875) = -1.78125
\]

\[
k_{3,2} = f_2(0.25, 3.5625, 6.42875) = 1.715125
\]

\[
k_{4,1} = f_1(0.5, 4 + (-1.78125)(0.5), 6 + (1.715125)(0.5))
\]

\[
= f_1(0.5, 3.109375, 6.857563) = -1.554688
\]

\[
k_{4,2} = f_2(0.5, 3.109375, 6.857563) = 1.631794
\]
\[
\begin{align*}
\frac{dy_1}{dx} &= -0.5y_1 && \frac{dy_2}{dx} = 4 - 0.3y_2 - 0.1y_1 \\
x &= 0, 2 && y_1 = 4, y_2 = 6, h = 0.5 \\
k_{1,1} &= -2 && k_{1,2} = 1.8 && k_{2,1} = -1.75 && k_{2,2} = 1.715 \\
k_{3,1} &= -1.78125 && k_{3,2} = 1.715125 \\
k_{4,1} &= -1.554688 && k_{4,2} = 1.631794 \\
y_1(0.5) &= 4 + (-2 + 2(-1.75) + 2(-1.78125) + (-1.554688))(0.5) / 6 \\
&= 3.115234 \\
y_2(0.5) &= 6 + (1.8 + 2(1.715) + 2(1.715125) + 1.631794)(0.5) / 6 \\
&= 6.857670
\end{align*}
\]
Adaptive RK Methods

• A fixed step size may be overkill for some regions of a function and may be too large to be accurate for others
• Adaptive methods use different step sizes for different regions of the function
• Several methods for accomplishing this
  – Use different step sizes but same order
  – Use different orders
Adaptive RK or Step-Halving

- Predict over step with order 4 RK, obtain prediction $y_1$
- Predict with two steps of half step size to obtain prediction $y_2$
- Difference $\Delta = y_2 - y_1$ is an estimate of the error that can be used to control step size adjustment
- $y_2^* = y_2 + \Delta/15$ is fifth order accurate
\[ y' = 4e^{0.8x} - 0.5y \]
\[ x = 0 \text{ to } 2, \quad h = 2 \quad y(0) = 2 \]

True value at 2 is 14.84392

Full step prediction

\[ k_1 = f(x_i, y_i) = f(0, 2) = 3 \]
\[ k_2 = f(x_i + h/2, y_i + k_1 h/2) = f(1, 5) = 6.402164 \]
\[ k_3 = f(x_i + h/2, y_i + k_2 h/2) = f(1, 8.402164) = 4.701082 \]
\[ k_4 = f(x_i + h, y_i + k_3 h) = f(2, 11.40216) = 14.11105 \]
\[ y_{i+1} = y_i + (k_1 + 2k_2 + 2k_3 + k_4)h/6 = 15.10585 \]
\[ y' = 4e^{0.8x} - 0.5y \]

\[ x = 0 \text{ to } 2, \ h = 2, \ y(0) = 2 \]

True value at 2 is 14.84392

Full step prediction is 15.10585

Half-step predictions are

\[ y_{i+1} = 2 + (3 + 2(4.217299 + 3.912974) + 5.945677)1/6 \]
\[ = 6.201037 \]

\[ y_{i+2} = 6.201037 + (5.801645 + 2(8.729538 + 7.997565) + 12.712829)1/6 \]
\[ = 14.862484 \]

\[ E_a = (14.862484 - 15.10585)/15 = -0.1622 \]

\[ E_t = 14.84392 - 14.862484 = -0.01857 \]

\[ y^* = 14.862484 + (-0.1622) = 14.84627 \]

\[ E_t = 14.84392 - 14.84627 = -0.00235 \]
Fehlberg/Cash-Karp RK

• Instead of using two different step sizes, we can use two different orders.
• This may use too many function evaluations unless the two orders are coordinated.
• Fehlberg RK uses a fifth order method using the same function evaluations as a fourth order method.
• Coefficients due to Cash and Karp.
\[ y_{i+1}^{(4)} = y_i + \left( \frac{37}{378} k_1 + \frac{250}{621} k_3 + \frac{125}{594} k_4 + \frac{512}{1771} k_6 \right) h \]

\[ y_{i+1}^{(5)} = y_i + \left( \frac{2825}{27648} k_1 + \frac{18575}{48384} k_3 + \frac{13525}{55296} k_4 + \frac{277}{14336} k_5 + \frac{1}{4} k_6 \right) h \]

\[ k_1 = f(x_i, y_i) \]

\[ k_2 = f(x_i + \frac{1}{5} h, y_i + \frac{1}{5} k_1 h) \]

\[ k_3 = f(x_i + \frac{3}{10} h, y_i + \frac{3}{40} k_1 h + \frac{9}{40} k_2 h) \]

\[ k_4 = f(x_i + \frac{3}{5} h, y_i + \frac{3}{10} k_1 h - \frac{9}{10} k_2 h + \frac{6}{5} k_3 h) \]

\[ k_5 = f(x_i + h, y_i - \frac{11}{54} k_1 h + \frac{5}{2} k_2 h - \frac{70}{27} k_3 h + \frac{35}{27} k_4 h) \]

\[ k_6 = f(x_i + \frac{7}{8} h, y_i + \frac{1631}{55296} k_1 h + \frac{175}{512} k_2 h + \frac{575}{13824} k_3 h + \frac{44275}{110592} k_4 h + \frac{253}{4096} k_5 h) \]
Values needed for RK Fehlberg for the example.

<table>
<thead>
<tr>
<th>k</th>
<th>x</th>
<th>y</th>
<th>f(x,y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>k_1</td>
<td>0</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>k_2</td>
<td>0.4</td>
<td>3.2</td>
<td>3.908511</td>
</tr>
<tr>
<td>k_3</td>
<td>0.6</td>
<td>4.20883</td>
<td>4.359883</td>
</tr>
<tr>
<td>k_4</td>
<td>1.2</td>
<td>7.228398</td>
<td>6.832587</td>
</tr>
<tr>
<td>k_5</td>
<td>2</td>
<td>15.42765</td>
<td>12.09831</td>
</tr>
<tr>
<td>k_6</td>
<td>1.75</td>
<td>12.17686</td>
<td>10.13237</td>
</tr>
</tbody>
</table>
\[
y^{(4)}_{i+1} = 2 + \left( \frac{37}{378} \right) \left( 3 + \frac{250}{621} \right) 4.359883 + \frac{125}{594} \left( 6.832587 + \frac{512}{1771} \right) 10.1337 \right) h \\
= 14.83192 \\
y^{(5)}_{i+1} = y_i + \left( \frac{2825}{27648} \right) \left( 3 + \frac{18575}{48384} \right) 4.359883 + \frac{13525}{55296} \left( 6.832587 + \frac{277}{14336} \right) 12.09831 + \frac{1}{4} \left( 10.1337 \right) \right) h \\
= 14.83677 \\
E_a = 14.83677 - 14.83192 = .004842
\]
Step Size Control

• First we specify desired accuracy
• Relative error can be a problem if the function is near 0
• Absolute error takes no account of the scale of the function
• One method is to let the desired accuracy depend on a multiple of both the function and its derivative
\[ \Delta_{\text{new}} = \varepsilon y_{\text{scale}} \]
\[ y_{\text{scale}} = |y| \]
\[ y_{\text{scale}} = |y| + \left| h \frac{dy}{dx} \right| \]
\[ h_{\text{new}} = h_{\text{present}} \left| \frac{\Delta_{\text{new}}}{\Delta_{\text{present}}} \right|^\alpha \]

\( \alpha = .2 \) when the step size is increased
\( \alpha = .25 \) when the step size is decreased

This is one scheme of many for adaptive step size
Example:

\[
\frac{dy}{dx} + 0.6y = 10\exp\left(-\frac{(x - 2)^2}{2(0.075)^2}\right)
\]

\(y(0) = 0.5\)

General Solution

\(y = 0.5\exp(-0.6x)\)