Overview

1. Root finding
   - Recap
   - Newton–Raphson
2. Numerical integration
   - Newton–Cotes formulae
   - Accuracy analysis
   - Romberg integration
3. Ordinary differential equations
   - The Euler method
   - Higher-order ODEs
4. Summary

Next quiz 24/26 March!

Recap / Outline

- Start by bracketing the root

Recap / Outline

- Start by bracketing the root
- Bisection: halve the interval — always works!

Recap / Outline

- Start by bracketing the root
- Bisection: halve the interval — always works!
- False position: interpolate to guess next point
- Secant: similar to false position, but does not rely on bracketing. May fly off to outer space!

Recap / Outline

- Start by bracketing the root
- Bisection: halve the interval — always works!
- False position: interpolate to guess next point
- Secant: similar to false position, but does not rely on bracketing. May fly off to outer space!
- Bisection has linear convergence
- Secant is superlinear
- False position is usually superlinear

Newton–Raphson

Use Taylor expansion to estimate the root

- Very good if you are close enough

Newton–Raphson

Use Taylor expansion to estimate the root

- Very good if you are close enough
- Most useful if you have analytical expression for $f'(x)$
- Much less useful in 1-d if you must use numerical derivative: Secant is better, Ridders is much better!

Newton–Raphson

Use Taylor expansion to estimate the root

- Very good if you are close enough
- Most useful if you have analytical expression for $f'(x)$
- Much less useful in 1-d if you must use numerical derivative: Secant is better, Ridders is much better!
- Almost only general method on the market in n-dim

Newton–Raphson

Use Taylor expansion to estimate the root

- Very good if you are close enough
- Most useful if you have analytical expression for $f'(x)$
- Much less useful in 1-d if you must use numerical derivative: Secant is better, Ridders is much better!
- Almost only general method on the market in n-dim
- Can go horribly wrong!

Simple principle:

$$f(x+\varepsilon) = f(x) + \varepsilon f'(x) + \frac{\varepsilon^2}{2} f''(x) + \ldots$$

Ignore higher order terms, find $\varepsilon : f(x+\varepsilon) = 0$

Newton–Raphson

Use Taylor expansion to estimate the root

- Very good if you are close enough
- Most useful if you have analytical expression for $f'(x)$
- Much less useful in 1-d if you must use numerical derivative: Secant is better, Ridders is much better!
- Almost only general method on the market in n-dim
- Can go horribly wrong!

Simple principle:

$$f(x+\varepsilon) = f(x) + \varepsilon f'(x) + \frac{\varepsilon^2}{2} f''(x) + \dots$$

Ignore higher order terms, find $\varepsilon : f(x+\varepsilon) = 0$

$$\Longrightarrow \quad \varepsilon = -f(x)/f'(x)$$

Newton–Raphson in more dimensions

We have N equations for N variables:

$$f_i(x_1, \ldots, x_N) = 0\,, \quad i = 1, \ldots, N$$

Newton–Raphson in more dimensions

We have N equations for N variables:

$$f_i(x_1, \ldots, x_N) = 0\,, \quad i = 1, \ldots, N$$

Taylor expansion:

$$f_i(\vec{x} + \vec{\delta}) = f_i(\vec{x}) + \sum_{j=1}^{N} \frac{\partial f_i}{\partial x_j}\delta_j + \mathcal{O}(\delta^2)$$

Newton–Raphson in more dimensions

We have N equations for N variables:

$$f_i(x_1, \ldots, x_N) = 0\,, \quad i = 1, \ldots, N$$

Taylor expansion:

$$f_i(\vec{x} + \vec{\delta}) = f_i(\vec{x}) + \sum_{j=1}^{N} \frac{\partial f_i}{\partial x_j} \delta_j + \mathcal{O}(\delta^2)$$

In vector–matrix notation:

$$F(x + \delta) = F(x) + J \cdot \delta\,, \quad J_{ij} \equiv \frac{\partial f_i}{\partial x_j} = \text{Jacobian}$$

Solve matrix eq $J\delta = -F$ for the vector δ.

Numerical integration

Two main approaches

1. Quadrature (classical numerical integration):
   add up integrand at a series of points
2. Transform integral into an ode:

$$I(y) = \int_{y_0}^{y} f(x)dx \quad \Longleftrightarrow \quad \frac{dI}{dy} = f(y)\,, \quad I(y_0) = 0\,.$$

$\longrightarrow$ our main business this semester!

Numerical integration

Two main approaches

1. Quadrature (classical numerical integration):
   add up integrand at a series of points
2. Transform integral into an ode:

$$I(y) = \int_{y_0}^{y} f(x)dx \quad \Longleftrightarrow \quad \frac{dI}{dy} = f(y)\,, \quad I(y_0) = 0\,.$$

$\rightarrow$ our main business this semester!

Other approaches

- use polynomial or rational approximations
- Fourier-type integrals: Fast Fourier Transform (next semester)
- Multidimensional integrals: Monte Carlo (next semester)

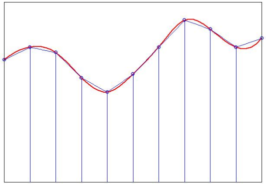
Numerical integration: quadrature

History dates back to Newton, mainstay of numerical methods up to 19th c
Newton–Cotes formulae (trapezium, Simpson, Boole,. . .)

Numerical integration: quadrature

History dates back to Newton, mainstay of numerical methods up to 19th c
Newton–Cotes formulae (trapezium, Simpson, Boole,. . .)

Trapezium rule



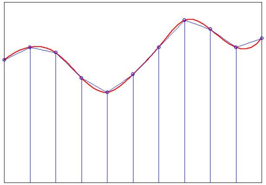
$$\int_a^b f(x)dx = \sum_{n=1}^{N} \int_{x_{n-1}}^{x_n} f(x)dx$$

$$\approx \sum_{n=1}^{N} \left(\frac{1}{2} f_{n-1} + \frac{1}{2} f_n \right) \cdot \delta$$

Numerical integration: quadrature

History dates back to Newton, mainstay of numerical methods up to 19th c
Newton–Cotes formulae (trapezium, Simpson, Boole,. . .)

Trapezium rule



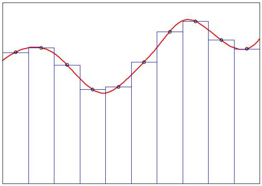
$$\int_a^b f(x)dx = \sum_{n=1}^{N} \int_{x_{n-1}}^{x_n} f(x)dx$$

$$\approx \sum_{n=1}^{N} \left(\frac{1}{2} f_{n-1} + \frac{1}{2} f_n \right) \cdot \delta$$

This is the most useful of all quadrature rules!

Quadrature

Midpoint rule

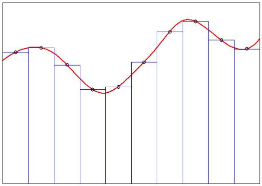


$$\int_a^b f(x)dx = \sum_{n=1}^{N} \int_{x_{n-1}}^{x_n} f(x)dx$$

$$\approx \sum_{n=1}^{N} f_{n-1/2} \cdot \delta$$

Quadrature

Midpoint rule



$$\int_a^b f(x)dx = \sum_{n=1}^{N} \int_{x_{n-1}}^{x_n} f(x)dx$$

$$\approx \sum_{n=1}^{N} f_{n-1/2} \cdot \delta$$

Simpson's rule

$$\int_{x_{n-1}}^{x_{n+1}} f(x)dx \approx \delta \cdot \left(\frac{1}{3} f_{n-1} + \frac{4}{3} f_n + \frac{1}{3} f_{n+1}\right)$$

$$\int_a^b f(x)dx \approx \delta \cdot \left(\frac{1}{3} f_0 + \frac{4}{3} f_1 + \frac{2}{3} f_2 + \frac{4}{3} f_3 + \ldots + \frac{4}{3} f_{N-1} + \frac{1}{3} f_N\right)$$

Accuracy

Consider one step of the integral using trapezium rule:

$$\int_{x_{n-1}}^{x_n} f(x)dx \approx \frac{\delta}{2}(f_{n-1} + f_n)$$

Accuracy

Consider one step of the integral using trapezium rule:

$$\int_{x_{n-1}}^{x_n} f(x)dx \approx \frac{\delta}{2}(f_{n-1} + f_n)$$

Taylor expand the integrand around x_n:

$$f(x_n + \varepsilon) = f(x_n) + \varepsilon f'(x_n) + \frac{\varepsilon^2}{2} f''(x_n) + \mathcal{O}(\varepsilon^3)$$

Accuracy

Consider one step of the integral using trapezium rule:

$$\int_{x_{n-1}}^{x_n} f(x)dx \approx \frac{\delta}{2}(f_{n-1} + f_n)$$

Taylor expand the integrand around x_n:

$$f(x_n + \varepsilon) = f(x_n) + \varepsilon f'(x_n) + \frac{\varepsilon^2}{2} f''(x_n) + \mathcal{O}(\varepsilon^3)$$

The sub-integral is

$$\begin{aligned}\int_{x_{n-1}}^{x_n} f(x)dx = \int_{-\delta}^{0} f(x_n + \varepsilon)d\varepsilon = \int_{-\delta}^{0} \left[f_n + \varepsilon f_n' + \frac{\varepsilon^2}{2} f_n'' + \mathcal{O}(\varepsilon^3)\right] \\ = \delta f_n - \frac{\delta^2}{2} f_n' + \frac{\delta^3}{6} f_n'' + \mathcal{O}(\delta^4)\end{aligned} \qquad \text{(I)}$$

Accuracy

Now Taylor expand f_{n-1} around x_n:

$$f(x_n - \delta) = f(x_n) - \delta f'(x_n) + \frac{\delta^2}{2} f''(x_n) + \mathcal{O}(\delta^3)$$

Accuracy

Now Taylor expand f_{n-1} around x_n:

$$f(x_n - \delta) = f(x_n) - \delta f'(x_n) + \frac{\delta^2}{2} f''(x_n) + \mathcal{O}(\delta^3)$$

The trapezium approximation then becomes

$$\begin{aligned} \frac{\delta}{2}(f_{n-1} + f_n) &= \frac{\delta}{2} f_n - \frac{\delta}{2} \delta f_n' + \frac{\delta}{2} \frac{\delta^2}{2} f_n' + \frac{\delta}{2} f_n + \mathcal{O}(\delta^4) \\ &= \delta f_n - \frac{\delta^2}{2} f_n' + \frac{\delta^3}{4} f_n'' + \mathcal{O}(\delta^4) \end{aligned} \qquad \text{(T)}$$

Accuracy

Now Taylor expand f_{n-1} around x_n:

$$f(x_n - \delta) = f(x_n) - \delta f'(x_n) + \frac{\delta^2}{2} f''(x_n) + \mathcal{O}(\delta^3)$$

The trapezium approximation then becomes

$$\begin{aligned} \frac{\delta}{2}(f_{n-1} + f_n) &= \frac{\delta}{2} f_n - \frac{\delta}{2}\delta f_n' + \frac{\delta}{2}\frac{\delta^2}{2} f_n' + \frac{\delta}{2} f_n + \mathcal{O}(\delta^4) \\ &= \delta f_n - \frac{\delta^2}{2} f_n' + \frac{\delta^3}{4} f_n'' + \mathcal{O}(\delta^4) \end{aligned} \qquad \text{(T)}$$

Compare with (I):

$$\int_{x_{n-1}}^{x_n} f(x)dx = \delta f_n - \frac{\delta^2}{2} f_n' + \frac{\delta^3}{6} f_n'' + \mathcal{O}(\delta^4)$$

Accuracy of trapezium rule

Accuracy of single step

$$\frac{\delta}{2}(f_{n-1} + f_n) = \int_{x_{n-1}}^{x_n} f(x)dx + \frac{\delta^3}{12} f_n'' + \mathcal{O}(\delta^4) = \int_{x_{n-1}}^{x_n} f(x)dx + \mathcal{O}(\delta^3)$$

Accuracy of trapezium rule

Accuracy of single step

$$\frac{\delta}{2}(f_{n-1} + f_n) = \int_{x_{n-1}}^{x_n} f(x)dx + \frac{\delta^3}{12} f_n'' + \mathcal{O}(\delta^4) = \int_{x_{n-1}}^{x_n} f(x)dx + \mathcal{O}(\delta^3)$$

Accuracy of entire integral

What is the error on $I = \int_a^b f(x)dx$?

Accuracy of trapezium rule

Accuracy of single step

$$\frac{\delta}{2}(f_{n-1} + f_n) = \int_{x_{n-1}}^{x_n} f(x)dx + \frac{\delta^3}{12} f_n'' + \mathcal{O}(\delta^4) = \int_{x_{n-1}}^{x_n} f(x)dx + \mathcal{O}(\delta^3)$$

Accuracy of entire integral

What is the error on $I = \int_a^b f(x)dx$?

— sum up the error on each step!

— we cannot assume any cancellations

$$I_T = I + \sum_{n=1}^{N} \mathcal{O}(\delta^3) = I + N\mathcal{O}(\delta^3)$$

Accuracy of trapezium rule

Accuracy of single step

$$\frac{\delta}{2}(f_{n-1} + f_n) = \int_{x_{n-1}}^{x_n} f(x)dx + \frac{\delta^3}{12} f_n'' + \mathcal{O}(\delta^4) = \int_{x_{n-1}}^{x_n} f(x)dx + \mathcal{O}(\delta^3)$$

Accuracy of entire integral

What is the error on $I = \int_a^b f(x)dx$?

— sum up the error on each step!

— we cannot assume any cancellations

$$I_T = I + \sum_{n=1}^{N} \mathcal{O}(\delta^3) = I + N\mathcal{O}(\delta^3)$$

But $N = (b-a)/\delta$, so $N\mathcal{O}(\delta^3) = (b-a)\mathcal{O}(\delta^2) = \mathcal{O}(\delta^2)$

Accuracy of trapezium rule

Accuracy of single step

$$\frac{\delta}{2}(f_{n-1} + f_n) = \int_{x_{n-1}}^{x_n} f(x)dx + \frac{\delta^3}{12} f_n'' + \mathcal{O}(\delta^4) = \int_{x_{n-1}}^{x_n} f(x)dx + \mathcal{O}(\delta^3)$$

Accuracy of entire integral

What is the error on $I = \int_a^b f(x)dx$?

— sum up the error on each step!

— we cannot assume any cancellations

$$N = (b-a)/\delta \quad \Longrightarrow \quad N\mathcal{O}(\delta^3) = (b-a)\mathcal{O}(\delta^2) = \mathcal{O}(\delta^2)$$

$$I_T = I + N\mathcal{O}(\delta^3) = \mathcal{O}(\delta^2)$$

Accuracy: Midpoint and Simpson

Midpoint rule

The best here is to Taylor expand integrand around midpoint, which give

$$\int_{x_{n-1}}^{x_n} f(x)dx = \delta \cdot f_{n-1/2} + \frac{\delta^3}{24} f''_{n-1/2} + \mathcal{O}(\delta^5)$$

$$\implies \quad I_M = I + \mathcal{O}(\delta^2)$$

Midpoint and trapezium are both accurate up to $\mathcal{O}(\delta^2)$ or $\mathcal{O}(1/N^2)$.

Accuracy: Midpoint and Simpson

Midpoint rule

The best here is to Taylor expand integrand around midpoint, which give

$$\int_{x_{n-1}}^{x_n} f(x)dx = \delta \cdot f_{n-1/2} + \frac{\delta^3}{24} f''_{n-1/2} + \mathcal{O}(\delta^5)$$

$$\implies \quad I_M = I + \mathcal{O}(\delta^2)$$

Midpoint and trapezium are both accurate up to $\mathcal{O}(\delta^2)$ or $\mathcal{O}(1/N^2)$.

Simpson's rule

$$\int_{x_{n-1}}^{x_{n+1}} f(x)dx = 2\delta \cdot f_n + \frac{\delta^3}{3} f''_n + \frac{\delta^5}{60} f^{(4)} + \mathcal{O}(\delta^7)$$

$$\delta \cdot \left(\frac{1}{3} f_{n-1} + \frac{4}{3} f_n + \frac{1}{3} f_{n+1}\right) = 2\delta \cdot f_n + \frac{\delta^3}{3} f''_n + \frac{\delta^5}{36} f^{(4)} + \mathcal{O}(\delta^7)$$

$$I_S = I + \mathcal{O}(\delta^4) = I + \mathcal{O}(1/N^4)$$

Romberg integration

Basic idea:

- evaluate I_T (trapezium rule) with N points (stepsize δ)
- then evaluate with $2N$ points (include all midpoints) — stepsize $\delta/2$
- then evaluate with $4N$ points, etc.

This gives a series of numbers depending on δ
$\longrightarrow$ extrapolate as polynomial in δ to $\delta \to 0$.

Romberg integration

Basic idea:

- evaluate I_T (trapezium rule) with N points (stepsize δ)
- then evaluate with $2N$ points (include all midpoints) — stepsize $\delta/2$
- then evaluate with $4N$ points, etc.

This gives a series of numbers depending on δ
$\longrightarrow$ extrapolate as polynomial in δ to $\delta \to 0$.

The series is in fact a polynomial in δ^2 (all odd terms cancel)

The first order extrapolation gives Simpsons rule!

Ordinary differential equations

Consider again the integral $I = \int_a^b f(x)dx$.

Ordinary differential equations

Consider again the integral $I = \int_a^b f(x)dx$. Introduce the function

$$I(x) = \int_a^x f(\xi)d\xi \quad \Longrightarrow I(a) = 0, \; I(b) = I \, .$$

Ordinary differential equations

Consider again the integral $I = \int_a^b f(x)dx$. Introduce the function

$$I(x) = \int_a^x f(\xi)d\xi \quad \Longrightarrow \; I(a) = 0, \; I(b) = I \, .$$

This function obeys the ODE

$$\frac{dI}{dx} = f(x) \; , \quad I(a) = 0$$

Our integral has been transformed into an ODE inital-value problem!

Ordinary differential equations

Consider again the integral $I = \int_a^b f(x)dx$. Introduce the function

$$I(x) = \int_a^x f(\xi)d\xi \quad \Longrightarrow \; I(a) = 0, \; I(b) = I \, .$$

This function obeys the ODE

$$\frac{dI}{dx} = f(x) \, , \quad I(a) = 0$$

Our integral has been transformed into an ODE inital-value problem!

Main advantage

Powerful methods exist for solving ODEs

- we will acquaint ourselves with these methods
- they apply to general, first-order or higher-order odes

$$\frac{dy}{dx} = f(x, y) \, , \qquad \frac{d^2y}{dx^2} + b(x, y)\frac{dy}{dx} + c(x, y) = 0 \, , \quad \text{etc}$$

The Euler method

Let us try to solve the equation

$$\frac{dy}{dx} = f(x) \qquad \text{starting from} \quad y(a) = c \quad x \in [a, b]$$

The Euler method

Let us try to solve the equation

$$\frac{dy}{dx} = f(x) \qquad \text{starting from} \quad y(a) = c \quad x \in [a, b]$$

We divide the interval into N segments with length $\varepsilon = (b - a)/N$
Number the points

$$x_0 = a, x_1 = a + \varepsilon, \ldots, x_N = b$$

We want to compute the values

$$y_n = y(x_n) = y(x_0 + n\varepsilon) \quad \forall n = 1, \ldots, N$$

We know $y(x_0) = y_0 = c$.

The Euler method

We know $y(x_0) = y_0 = c$.
We can calculate $v_0 = y'(x_0) = f(x_0)$

The Euler method

We know $y(x_0) = y_0 = c$.
We can calculate $v_0 = y'(x_0) = f(x_0)$

Take a small step ε:

$$y_1 = y(x_1) = y(x_0 + \varepsilon) = y(x_0) + \varepsilon y'(x_0) = y_0 + \varepsilon v_0$$
$$v_1 = y'(x_1) = f(x_1)$$

Iterate this to get $y_2, y_3, \ldots, y_N$

The Euler method

We know $y(x_0) = y_0 = c$.
We can calculate $v_0 = y'(x_0) = f(x_0)$

Take a small step ε:

$$y_1 = y(x_1) = y(x_0 + \varepsilon) = y(x_0) + \varepsilon y'(x_0) = y_0 + \varepsilon v_0$$
$$v_1 = y'(x_1) = f(x_1)$$

Iterate this to get $y_2, y_3, \ldots, y_N$

Algorithm

Set $x_0 = a, y_0 = c$, choose N or stepsize ε
Calculate $v_0 = f(x_0)$.
Then, for each $n = 1, \ldots, N$ do

1. $x_n = x_{n-1} + \varepsilon$
2. $y_n = y_{n-1} + \varepsilon v_{n-1}$
3. $v_n = f(x_n)$

Error estimate

Taylor expand as before:

$$\begin{aligned} y^{\text{true}}(x_n) &= y(x_{n-1}) + \varepsilon y'(x_{n-1}) + \frac{\varepsilon^2}{2} y''(x_{n-1}) + \ldots \\ &= y_n^E + \mathcal{O}(\varepsilon^2) \end{aligned} \tag{1}$$

Error estimate

Taylor expand as before:

$$\begin{aligned} y^{\text{true}}(x_n) &= y(x_{n-1}) + \varepsilon y'(x_{n-1}) + \frac{\varepsilon^2}{2} y''(x_{n-1}) + \ldots \\ &= y_n^E + \mathcal{O}(\varepsilon^2) \end{aligned} \tag{1}$$

Each step induces an error of $\mathcal{O}(\varepsilon^2)$.
The number of steps is $N = (b-a)/\varepsilon$ [or $n = (x_n - a)/\varepsilon$]

Real error

$$y^{\text{true}} - y^E \sim N \cdot \mathcal{O}(\varepsilon^2) \sim \mathcal{O}(\varepsilon)$$

Error estimate

Taylor expand as before:

$$y^{\text{true}}(x_n) = y(x_{n-1}) + \varepsilon y'(x_{n-1}) + \frac{\varepsilon^2}{2} y''(x_{n-1}) + \dots \\ = y_n^E + \mathcal{O}(\varepsilon^2) \tag{1}$$

Each step induces an error of $\mathcal{O}(\varepsilon^2)$.
The number of steps is $N = (b-a)/\varepsilon$ [or $n = (x_n - a)/\varepsilon$]

Real error

$$y^{\text{true}} - y^E \sim N \cdot \mathcal{O}(\varepsilon^2) \sim \mathcal{O}(\varepsilon)$$

This analysis holds in exactly the same way for a general ODE

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

The Euler method is accurate up to $\mathcal{O}(\varepsilon)$

Higher order ODEs

Generic second order ODE:

$$\frac{d^2y}{dx^2} + f(x, y)\frac{dy}{dx} + g(x, y) = 0$$

How would we solve this numerically?

Higher order ODEs

Generic second order ODE:

$$\frac{d^2y}{dx^2} + f(x,y)\frac{dy}{dx} + g(x,y) = 0$$

How would we solve this numerically?

I. Try discretising both derivatives

Use symmetric derivatives for both:

$$\frac{y_{n+1} - 2y_n + y_{n-1}}{\varepsilon^2} + f(x_n, y_n)\frac{y_{n+1} - y_{n-1}}{2\varepsilon} + g(x_n, y_n) = 0$$

$$\implies \quad y_{n+1} = \frac{2y_n - \left(1 - \frac{\varepsilon}{2} f(x_n, y_n)\right) y_{n-1} - g(x_n, y_n)}{1 + \frac{\varepsilon}{2} f(x_n, y_n)}$$

Higher order ODEs

Generic second order ODE:

$$\frac{d^2y}{dx^2} + f(x,y)\frac{dy}{dx} + g(x,y) = 0$$

How would we solve this numerically?

I. Try discretising both derivatives

Use symmetric derivatives for both:

$$\frac{y_{n+1} - 2y_n + y_{n-1}}{\varepsilon^2} + f(x_n, y_n)\frac{y_{n+1} - y_{n-1}}{2\varepsilon} + g(x_n, y_n) = 0$$

$$\Longrightarrow \quad y_{n+1} = \frac{2y_n - \left(1 - \frac{\varepsilon}{2}f(x_n, y_n)\right)y_{n-1} - g(x_n, y_n)}{1 + \frac{\varepsilon}{2}f(x_n, y_n)}$$

- Need to know both y_n and y_{n-1} to compute y_{n+1}
- Can get y_1 from y_0, y_0' using forward derivative, then iterate

Higher order ODEs

Generic second order ODE:

$$\frac{d^2y}{dx^2} + f(x,y)\frac{dy}{dx} + g(x,y) = 0$$

How would we solve this numerically?

I. Try discretising both derivatives

Use symmetric derivatives for both:

$$\frac{y_{n+1} - 2y_n + y_{n-1}}{\varepsilon^2} + f(x_n, y_n)\frac{y_{n+1} - y_{n-1}}{2\varepsilon} + g(x_n, y_n) = 0$$

$$\implies \quad y_{n+1} = \frac{2y_n - \left(1 - \frac{\varepsilon}{2} f(x_n, y_n)\right) y_{n-1} - g(x_n, y_n)}{1 + \frac{\varepsilon}{2} f(x_n, y_n)}$$

Error in derivatives: $\mathcal{O}(\varepsilon^2)$ Error in each step: $\varepsilon^2 \mathcal{O}(\varepsilon^2) = \mathcal{O}(\varepsilon^4)$

Higher order ODEs

Generic second order ODE:

$$\frac{d^2y}{dx^2} + f(x,y)\frac{dy}{dx} + g(x,y) = 0$$

How would we solve this numerically?

I. Try discretising both derivatives

Use symmetric derivatives for both:

$$\frac{y_{n+1} - 2y_n + y_{n-1}}{\varepsilon^2} + f(x_n, y_n)\frac{y_{n+1} - y_{n-1}}{2\varepsilon} + g(x_n, y_n) = 0$$

$$\implies \quad y_{n+1} = \frac{2y_n - \left(1 - \frac{\varepsilon}{2} f(x_n, y_n)\right) y_{n-1} - g(x_n, y_n)}{1 + \frac{\varepsilon}{2} f(x_n, y_n)}$$

Error in derivatives: $\mathcal{O}(\varepsilon^2)$ Error in each step: $\varepsilon^2 \mathcal{O}(\varepsilon^2) = \mathcal{O}(\varepsilon^4)$

Total error: $\mathcal{O}(\varepsilon^3)$

Higher order ODE

Generic second order ODE:

$$\frac{d^2y}{dx^2} + f(x,y)\frac{dy}{dx} + g(x,y) = 0$$

II. Make it into two first order ODEs

Define 'new' function $z(x) = y'(x)$

$$\frac{dy}{dx} = z\,, \qquad \frac{dz}{dx} = -f(x,y)z - g(x,y)$$

Higher order ODE

Generic second order ODE:

$$\frac{d^2y}{dx^2} + f(x,y)\frac{dy}{dx} + g(x,y) = 0$$

II. Make it into two first order ODEs

Define 'new' function $z(x) = y'(x)$

$$\frac{dy}{dx} = z\,, \qquad \frac{dz}{dx} = -f(x,y)z - g(x,y)$$

This system can be solved using the Euler method:

$$\begin{aligned} y_{n+1} &= y_n + \varepsilon z_n \\ z_{n+1} &= z_n - \varepsilon\big[f(x_n,y_n)z_n - g(x_n,y_n)\big] \end{aligned}$$

Higher order ODE

Generic second order ODE:

$$\frac{d^2y}{dx^2} + f(x,y)\frac{dy}{dx} + g(x,y) = 0$$

II. Make it into two first order ODEs

Define ‘new’ function $z(x) = y'(x)$

$$\frac{dy}{dx} = z\,, \qquad \frac{dz}{dx} = -f(x,y)z - g(x,y)$$

This system can be solved using the Euler method:

$$y_{n+1} = y_n + \varepsilon z_n$$
$$z_{n+1} = z_n - \varepsilon\big[f(x_n,y_n)z_n - g(x_n,y_n)\big]$$

Error in each step: $\mathcal{O}(\varepsilon^2)$
Total error: $\mathcal{O}(\varepsilon)$

Molecular dynamics

The classical equations of motion for a collection of interacting particles (eg, molecules or planets) will be

- coupled second-order ODEs for positions $x_i(t)$ (Lagrange), or
- coupled first-order ODEs for positions $x_i(t)$ and momenta $p_i(t)$ (Hamilton)

Solving such coupled equations numerically is molecular dynamics.

Molecular dynamics

The classical equations of motion for a collection of interacting particles (eg, molecules or planets) will be

- coupled second-order ODEs for positions $x_i(t)$ (Lagrange), or
- coupled first-order ODEs for positions $x_i(t)$ and momenta $p_i(t)$ (Hamilton)

Solving such coupled equations numerically is molecular dynamics.

Reducing everything to first-order ODEs corresponds to the 'Hamiltonian' approach. This has several advantages, first and foremost:

You can use the same methods for all equations

Molecular dynamics

The classical equations of motion for a collection of interacting particles (eg, molecules or planets) will be

- coupled second-order ODEs for positions $x_i(t)$ (Lagrange), or
- coupled first-order ODEs for positions $x_i(t)$ and momenta $p_i(t)$ (Hamilton)

Solving such coupled equations numerically is molecular dynamics.

Reducing everything to first-order ODEs corresponds to the 'Hamiltonian' approach. This has several advantages, first and foremost:

You can use the same methods for all equations

Black-box solvers (`ode45` etc) require as 'input' a set of first-order equations with initial values.

You may still need physical or mathematical insight to choose the most appropriate method!

Summary

- Trapezium, midpoint: $\mathcal{O}(\delta^2)$; Simpson: $\mathcal{O}(\delta^4)$
- Trapezium (or midpoint) easier to extend with Romberg integration

Summary

- Trapezium, midpoint: $\mathcal{O}(\delta^2)$; Simpson: $\mathcal{O}(\delta^4)$
- Trapezium (or midpoint) easier to extend with Romberg integration
- Euler method: simple iterative procedure, accurate to $\mathcal{O}(\varepsilon)$
- Higher-order ODEs can be reduced to coupled first-order ODEs