15. ODEs II: Runge–Kutta methods

Last time

- Review of ODEs
- Euler method and error
- Trapezium method
- Systems of equations

Goals for today

- Systems of equations
- Higher derivatives
- Runge–Kutta methods

Systems of equations

- General system of 1st-order ODEs:

$$\begin{aligned}
\dot{x}_1 &= f_1(t, x_1, \dots, x_n) \\
\dot{x}_2 &= f_2(t, x_1, \dots, x_n) \\
&\vdots \\
\dot{x}_n &= f_n(t, x_1, \dots, x_n)
\end{aligned}$$

Systems of equations

- General system of 1st-order ODEs:

$$\begin{aligned}
\dot{x}_1 &= f_1(t, x_1, \ldots, x_n) \\
\dot{x}_2 &= f_2(t, x_1, \ldots, x_n) \\
&\vdots \\
\dot{x}_n &= f_n(t, x_1, \ldots, x_n)
\end{aligned}$$

- Rewrite in vector form: $\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t))$
- $\mathbf{f} = (f_1, \ldots, f_n)$
- $\mathbf{x}(t) = (x_1(t), x_2(t), \ldots, x_n(t))$
- **f** is a **vector field**

Solving systems of equations

- $\mathbf{x}(t) = (x_1(t), x_2(t), \ldots, x_d(t))$ if d variables
- Taylor expand:

$$
\begin{aligned}
x_i(t_k + h) &= x_i(t_k) + h\, \dot{x}_i(t_k) + \mathcal{O}(h^2) \\
&= x_i(t_k) + h\, f_i(t_k, x_1, \ldots, x_n) + \mathcal{O}(h^2)
\end{aligned}
$$

- So obtain Euler method

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\, \mathbf{f}(\mathbf{x}_k)$$

Solving systems of equations

- $\mathbf{x}(t) = (x_1(t), x_2(t), \dots, x_d(t))$ if d variables
- Taylor expand:

$$\begin{aligned} x_i(t_k + h) &= x_i(t_k) + h\,\dot{x}_i(t_k) + \mathcal{O}(h^2) \\ &= x_i(t_k) + h\,f_i(t_k, x_1, \dots, x_n) + \mathcal{O}(h^2) \end{aligned}$$

- So obtain Euler method

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\,\mathbf{f}(\mathbf{x}_k)$$

- *Same* method but now with vectors
- *Same* code

Higher derivatives

- How treat higher-order equations (higher derivatives)?
- E.g. damped harmonic oscillator

$$\ddot{x} + b\dot{x} + \omega^2 x = 0$$

Higher derivatives

- How treat higher-order equations (higher derivatives)?
- E.g. damped harmonic oscillator

$$\ddot{x} + b\dot{x} + \omega^2 x = 0$$

- There are some special methods for second-order equations
- But usually **reduce** to system of 1st-order equations

Reduction to system of 1st-order equations

- Damped harmonic oscillator example:

$$\ddot{x} + b\dot{x} + \omega^2 x = 0$$

Reduction to system of 1st-order equations

- Damped harmonic oscillator example:

$$\ddot{x} + b\dot{x} + \omega^2 x = 0$$

- Introduce new variable $v := \dot{x}$
- Then $\dot{v} = \ddot{x}$

Reduction to system of 1st-order equations

- Damped harmonic oscillator example:

$$\ddot{x} + b\dot{x} + \omega^2 x = 0$$

- Introduce new variable $v := \dot{x}$
- Then $\dot{v} = \ddot{x}$
- So get system

$$\begin{aligned} \dot{x} &= v \\ \dot{v} &= -bv + \omega^2 x \end{aligned}$$

Euler method

- Recall: To solve $\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t))$
- Can use Euler method:

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h_n \, \mathbf{f}(t_n, \mathbf{x}_{n+1})$$

- Has local truncation error $\mathcal{O}(h^2)$
- Global error $\mathcal{O}(h)$

Reducing error

- How reduce error?
- Trapezium method is better, $\mathcal{O}(h^2)$, but **implicit** so more expensive
- Are there explicit methods of higher order?

Reducing error

- How reduce error?
- Trapezium method is better, $\mathcal{O}(h^2)$, but **implicit** so more expensive
- Are there explicit methods of higher order?

- Yes: Runge–Kutta and multistep methods
- We will look at **Runge–Kutta methods**

Second-order Taylor methods

- Use Taylor expansion to higher order for each step:
- Suppose $\dot{x}(t) = f(t, x(t))$
- Expand step to higher order:

$$x(t+h) = x(t) + h\,\dot{x}(t) + \tfrac{1}{2}h^2\,\ddot{x}(t) + \mathcal{O}(h^3)$$

- How can we deal with $\ddot{x}(t)$?

Second-order Taylor methods II

- Differentiate the ODE:

$$\ddot{x}(t) = \frac{d}{dt}\left[f(t, x(t))\right]$$

Second-order Taylor methods II

- Differentiate the ODE:

$$\ddot{x}(t) = \frac{d}{dt}\left[f(t, x(t))\right]$$

- Get $\ddot{x}(t) = f_t + f_x \dot{x}$
- Where $f_t := \frac{\partial f}{\partial t}(t, x(t))$ and similarly for f_x

Second-order Taylor methods II

- Differentiate the ODE:

$$\ddot{x}(t) = \frac{d}{dt}\left[f(t, x(t))\right]$$

- Get $\ddot{x}(t) = f_t + f_x \dot{x}$
- Where $f_t := \frac{\partial f}{\partial t}(t, x(t))$ and similarly for f_x
- So

$$x(t+h) \simeq x(t) + h\, f + \tfrac{1}{2} h^2 \left[f_t + f_x\, f\right]$$

- For clarity: wrote f instead of $f(t, x(t))$
- This is useful *only* if we can calculate the derivatives of f (see later)

Back to the trapezium rule

- Let's look at the trapezium rule again:

$$x_{n+1} \simeq x_n + \frac{h}{2}\left[f(x_n) + f(x_{n+1})\right]$$

- Implicit due to $f(x_{n+1})$ on right-hand side
- Could we make this explicit by approximating?

Back to the trapezium rule

- Let's look at the trapezium rule again:

$$x_{n+1} \simeq x_n + \frac{h}{2}\left[f(x_n) + f(x_{n+1})\right]$$

- Implicit due to $f(x_{n+1})$ on right-hand side
- Could we make this explicit by approximating?

- Use the Euler method! i.e. take an **Euler step**:

$$x_{n+1} \simeq x_n + h\, f(x_n)$$

- Use for x_{n+1} on right, which is already multiplied by h!

Runge–Kutta

- This gives a **multistage** method
- Notation: k_i = successive evaluations of f at different points:

$$\begin{aligned} k_1 &:= f(t_n, x_n) \\ k_2 &:= f(t_n + h, x_n + h\, k_1) \\ x_{n+1} &= x_n + \tfrac{h}{2}(k_1 + k_2) \end{aligned}$$

- **Modifed Euler method**

Runge–Kutta

- This gives a **multistage** method
- Notation: k_i = successive evaluations of f at different points:

$$\begin{aligned} k_1 &:= f(t_n, x_n) \\ k_2 &:= f(t_n + h, x_n + h\, k_1) \\ x_{n+1} &= x_n + \tfrac{h}{2}(k_1 + k_2) \end{aligned}$$

- **Modifed Euler method**
- NB: Some references put h in definition of k_i:

$$k_1' := h\, f(t_n, x_n)$$

Order of modified Euler

- Taylor expand:

$$k_2 = f + h\, f_t + h\, k_1\, f_x + \mathcal{O}(h^2)$$

- So

$$\begin{aligned} x_{n+1} &= x_n + \tfrac{h}{2}(k_1 + k_2) \\ &\simeq x_n + \tfrac{h}{2}\left[f + f + hf_t + hk_1 f_x + \mathcal{O}(h^2)\right] \\ &= x_n + h\, f + \tfrac{1}{2}h^2(f_t + f_x f) + \mathcal{O}(h^3) \end{aligned}$$

- 2nd-order Taylor expansion via nested evaluation of f!

General Runge–Kutta methods

- General idea: Match Taylor expansion
- Add more coefficients in more **stages**

$$
\begin{aligned}
k_1 &= f(t_n, x_n) \\
k_2 &= f(t_n + c_1\, h, x_n + a_{11} k_1) \\
k_3 &= f(t_n + c_2\, h, x_n + a_{21} k_1 + a_{22} k_2) \\
&\vdots \\
k_s &= f(t_n + c_1\, h, x_n + a_{s-1,1} k_1 + \cdots + a_{s-1,s-1} k_{s-1}) \\
x_{n+1} &= x_n + h(b_1 k_1 + \cdots + b_s k_s)
\end{aligned}
$$

- Coefficients must satisfy certain constraints

Butcher tableau

- Arrange coefficients in **Butcher tableau**:

$$
\begin{array}{c|ccccc}
0 & & & & & \\
c_1 & a_{11} & & & & \\
c_2 & a_{21} & a_{22} & & & \\
\vdots & & & & & \\
c_{s-1} & a_{s-1,1} & \cdots & a_{s-1,s-1} & & \\
\hline
 & b_1 & b_2 & b_{s-1} & b_s &
\end{array}
$$

- E.g. for modified Euler:

$$
\begin{array}{c|cc}
0 & & \\
1 & 1 & \\
\hline
 & \frac{1}{2} & \frac{1}{2}
\end{array}
$$

RK4

- Elegant and efficient 4th-order method

$$
\begin{aligned}
k_1 &= f(t_n, x_n) \\
k_2 &= f(t_n + h/2, x_n + k_1/2) \\
k_3 &= f(t_n + h/2, x_n + k_2/2) \\
k_4 &= f(t_n + h, x_n + k_3) \\
x_{n+1} &= x_n + \tfrac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)
\end{aligned}
$$

RK4 II

- Simpler to understand and implement as Butcher tableau:

$$
\begin{array}{c|cccc}
0 & & & & \\
\frac{1}{2} & \frac{1}{2} & & & \\
\frac{1}{2} & 0 & \frac{1}{2} & & \\
1 & 0 & 0 & 1 & \\
\hline
 & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6}
\end{array}
$$

Summary

- Runge–Kutta methods: nested Euler steps
- Reproduce Taylor expansions to different orders