17. Ordinary differential equations (ODEs)

Summary of the previous lecture

- Numerical integration (quadrature)
- “Interpolate then integrate”
- Newton–Cotes methods with equally-spaced points
- Error analysis

Goals for today

- Ordinary Differential Equations (ODEs)
- Solutions of ODEs
- Alternative points of view
- Numerical methods:
    - Euler method

Ordinary differential equations

- What is an **ordinary differential equation** (ODE)?

Ordinary differential equations

- What is an **ordinary differential equation** (ODE)?
- An ODE defines a function x
- By relating x to one or more of its derivatives

Ordinary differential equations

- What is an **ordinary differential equation** (ODE)?
- An ODE defines a function x
- By relating x to one or more of its derivatives
- E.g. $\dot{x} = -\lambda x$

Ordinary differential equations

- What is an **ordinary differential equation** (ODE)?
- An ODE defines a function x
- By relating x to one or more of its derivatives
- E.g. $\dot{x} = -\lambda x$
- This means $\dot{x}(t) = -\lambda x(t)$ for all t

Ordinary differential equations

- What is an **ordinary differential equation** (ODE)?
- An ODE defines a function x
- By relating x to one or more of its derivatives
- E.g. $\dot{x} = -\lambda x$
- This means $\dot{x}(t) = -\lambda x(t)$ for all t
- This models e.g. radioactive decay: $x(t)$ is the proportion of nuclei that are still radioactive at time t

Ordinary differential equations

- What is an **ordinary differential equation** (ODE)?
- An ODE defines a function x
- By relating x to one or more of its derivatives
- E.g. $\dot{x} = -\lambda x$
- This means $\dot{x}(t) = -\lambda x(t)$ for all t
- This models e.g. radioactive decay: $x(t)$ is the proportion of nuclei that are still radioactive at time t
- We need to start from **initial condition** $x(t_0) = x_0$

Solving an ODE

- A more general ODE looks like

$$\dot{x} = f(x)$$

Solving an ODE

- A more general ODE looks like

$$\dot{x} = f(x)$$

- This means

$$\dot{x}(t) = f(x(t)) \quad \forall t \in [t_0, t_{\text{final}}]$$

Solving an ODE

- A more general ODE looks like

$$\dot{x} = f(x)$$

- This means

$$\dot{x}(t) = f(x(t)) \quad \forall t \in [t_0, t_{\text{final}}]$$

- This tells us *how fast* the solution changes if we are at a given current value

Solving an ODE

- A more general ODE looks like

$$\dot{x} = f(x)$$

- This means

$$\dot{x}(t) = f(x(t)) \quad \forall t \in [t_0, t_{\text{final}}]$$

- This tells us *how fast* the solution changes if we are at a given current value
- Together with the initial condition, this **implicitly** determines the value of $x(t)$ at all times t

Solving an ODE

- A more general ODE looks like

$$\dot{x} = f(x)$$

- This means

$$\dot{x}(t) = f(x(t)) \quad \forall t \in [t_0, t_{\text{final}}]$$

- This tells us *how fast* the solution changes if we are at a given current value
- Together with the initial condition, this **implicitly** determines the value of $x(t)$ at all times t
- The solution is a **function** $t \mapsto x(t)$ for $t \in [t_0, t_{\text{final}}]$

Existence and uniqueness

- Does an ODE *really* specify a solution $x(t)$?

Existence and uniqueness

- Does an ODE *really* specify a solution $x(t)$?
- **Yes** (locally)!: Existence and uniqueness theorem

Existence and uniqueness

- Does an ODE *really* specify a solution $x(t)$?
- **Yes** (locally)!: Existence and uniqueness theorem
- A sufficient condition for local existence and uniqueness is $f \in C^1$ (continuous first derivative)

The meaning of an ODE

- Start at the initial condition (known)
- The ODE gives us the instantaneous derivative (slope) of the solution at that point

The meaning of an ODE

- Start at the initial condition (known)
- The ODE gives us the instantaneous derivative (slope) of the solution at that point
- So we have a **tangent vector** to the solution at $t = t_0$
- This tells us the *direction* and *speed* to move

The meaning of an ODE

- Start at the initial condition (known)
- The ODE gives us the instantaneous derivative (slope) of the solution at that point
- So we have a **tangent vector** to the solution at $t = t_0$
- This tells us the *direction* and *speed* to move
- But as soon as we move a bit, we must change to a new direction!

The meaning of an ODE

- Start at the initial condition (known)
- The ODE gives us the instantaneous derivative (slope) of the solution at that point
- So we have a **tangent vector** to the solution at $t = t_0$
- This tells us the *direction* and *speed* to move
- But as soon as we move a bit, we must change to a new direction!
- An ODE makes precise the idea of changing direction in a continuous way at every “infinitesimal step”

Euler method

- This suggests a **numerical method**

Euler method

- This suggests a **numerical method**
- We literally try to follow the above prescription

Euler method

- This suggests a **numerical method**
- We literally try to follow the above prescription
- We need a method to approximate the true, unknown solution
- One possible way is to creep forward in small **time steps**

Time stepping

- Take equal time steps of length h (for now), giving nodes

$$t_n := t_0 + n\,h$$

Time stepping

- Take equal time steps of length h (for now), giving nodes

$$t_n := t_0 + n\,h$$

- We want to **approximate** the true solution $x(t_n)$ at the nth node

Time stepping

- Take equal time steps of length h (for now), giving nodes

$$t_n := t_0 + n\,h$$

- We want to **approximate** the true solution $x(t_n)$ at the nth node
- Call the approximation there x_n

Time stepping

- Take equal time steps of length h (for now), giving nodes

$$t_n := t_0 + n\,h$$

- We want to **approximate** the true solution $x(t_n)$ at the nth node
- Call the approximation there x_n
- We want to calculate a sequence of approximate values $x_0, x_1, \ldots, x_N$ at nodes $t_0, t_1, \ldots, t_N$

Collaboration I

The Euler method

Suppose we want to solve the ODE $\dot{x} = f(x)$ with $x(t_0) = x_0$.

1. Fix a small time step h. How can we approximate $x(t_0 + h)$?
2. What is the simplest solution? This is the **Euler method**.
3. How good is that?
4. How could we improve that solution?

The Euler method

- The simplest solution method is the **Euler method**

The Euler method

- The simplest solution method is the **Euler method**
- We expand in a **Taylor series**:

$$x(t_n + h) \simeq x(t_n) + h\,\dot{x}(t_n) + \cdots$$

The Euler method

- The simplest solution method is the **Euler method**
- We expand in a **Taylor series**:

$$x(t_n + h) \simeq x(t_n) + h\,\dot{x}(t_n) + \cdots$$

- Since $\dot{x}(t) = f(x(t))$ we obtain the algorithm

$$x_{n+1} = x_n + h\,f(x_n) \quad \text{– Euler method}$$

The Euler method

- The simplest solution method is the **Euler method**
- We expand in a **Taylor series**:

$$x(t_n + h) \simeq x(t_n) + h\,\dot{x}(t_n) + \cdots$$

- Since $\dot{x}(t) = f(x(t))$ we obtain the algorithm

$$x_{n+1} = x_n + h\,f(x_n) \quad \text{– Euler method}$$

- Stepping this forward in time gives a sequence of approximations $x_1, x_2, \ldots$ to the true solutions $x(t_1), x(t_2), \ldots$

The Euler method

- The simplest solution method is the **Euler method**
- We expand in a **Taylor series**:

$$x(t_n + h) \simeq x(t_n) + h\,\dot{x}(t_n) + \cdots$$

- Since $\dot{x}(t) = f(x(t))$ we obtain the algorithm

$$x_{n+1} = x_n + h\,f(x_n) \quad \text{– Euler method}$$

- Stepping this forward in time gives a sequence of approximations $x_1, x_2, \ldots$ to the true solutions $x(t_1), x(t_2), \ldots$
- Note that any error on one step propagates to future steps

Convergence of the Euler method

- A proposed numerical method like this is useful only if it is **convergent**:

Convergence of the Euler method

- A proposed numerical method like this is useful only if it is **convergent**:
- Call $x_{n,h}$ the solution x_n with time-step h

Convergence of the Euler method

- A proposed numerical method like this is useful only if it is **convergent**:
- Call $x_{n,h}$ the solution x_n with time-step h
- As $h \to 0$, the Euler method solution $(x_{0,h}, x_{1,h}, \ldots, x_{N(h),h})$ must converge to the true $(x(t_0), x(t_1), \ldots, x(t_N))$

Convergence of the Euler method

- A proposed numerical method like this is useful only if it is **convergent**:
- Call $x_{n,h}$ the solution x_n with time-step h
- As $h \to 0$, the Euler method solution $(x_{0,h}, x_{1,h}, \dots, x_{N(h),h})$ must converge to the true $(x(t_0), x(t_1), \dots, x(t_N))$
- I.e. the maximum distance should $\to 0$ as $h \to 0$

Convergence of the Euler method

- A proposed numerical method like this is useful only if it is **convergent**:
- Call $x_{n,h}$ the solution x_n with time-step h
- As $h \to 0$, the Euler method solution $(x_{0,h}, x_{1,h}, \ldots, x_{N(h),h})$ must converge to the true $(x(t_0), x(t_1), \ldots, x(t_N))$
- I.e. the maximum distance should $\to 0$ as $h \to 0$
- This can be proved correct: See e.g. Iserles, *A First Course in the Numerical Analysis of Differential Equations*

Collaboration II

Rate of convergence

1. What tool could we use to analyse the size of the error in a single Euler step, i.e. the **local error**?
2. What is the size of the local error?
3. Hence what is the **global error** after a given time as a function of h?

Rate of convergence

- We want to find the **rate of convergence** of the error as a function of h

Rate of convergence

- We want to find the **rate of convergence** of the error as a function of h
- Let's take a single step starting at the exact value:

$$
\begin{aligned}
x(t_{n+1}) &= x(t_n + h) \\
&= x(t_n) + h\,\dot{x}(t_n) + \tfrac{1}{2}h^2\ddot{x}(\xi) \\
&= x(t_n) + hf(x(t_n)) + \frac{1}{2}h^2\ddot{x}(\xi)
\end{aligned}
$$

Rate of convergence

- We want to find the **rate of convergence** of the error as a function of h
- Let's take a single step starting at the exact value:

$$\begin{aligned} x(t_{n+1}) &= x(t_n + h) \\ &= x(t_n) + h\,\dot{x}(t_n) + \tfrac{1}{2}h^2\ddot{x}(\xi) \\ &= x(t_n) + hf(x(t_n)) + \frac{1}{2}h^2\ddot{x}(\xi) \end{aligned}$$

- So the *local* error is $\mathcal{O}(h^2)$ at each step – **order 1**

Rate of convergence

- We want to find the **rate of convergence** of the error as a function of h
- Let's take a single step starting at the exact value:

$$\begin{aligned} x(t_{n+1}) &= x(t_n + h) \\ &= x(t_n) + h\,\dot{x}(t_n) + \tfrac{1}{2}h^2\ddot{x}(\xi) \\ &= x(t_n) + hf(x(t_n)) + \frac{1}{2}h^2\ddot{x}(\xi) \end{aligned}$$

- So the *local* error is $\mathcal{O}(h^2)$ at each step – **order 1**
- There are $N \sim \frac{1}{h}$ steps, so we expect the *global* error to be $\mathcal{O}(h)$

Inhomogeneous ODEs

- f can also depend explicitly on time:

$$\dot{x}(t) = f(t, x(t))$$

- E.g. if there is periodic external forcing, $\dot{x}(t) = x + \cos(t)$

Inhomogeneous ODEs

- f can also depend explicitly on time:

$$\dot{x}(t) = f(t, x(t))$$

- E.g. if there is periodic external forcing, $\dot{x}(t) = x + \cos(t)$

- Then the Euler method becomes

$$x_{n+1} = x_n + h_n \, f(t_n, x_n)$$

with general step sizes h_n

Systems of ODEs

Systems of equations

- Usually we will have > 1 dependent variables, e.g.

$$\dot{x} = f(x, y)$$
$$\dot{y} = g(x, y)$$

Systems of equations

- Usually we will have > 1 dependent variables, e.g.

$$\begin{aligned} \dot{x} &= f(x, y) \\ \dot{y} &= g(x, y) \end{aligned}$$

- x and y are **coupled** together

Systems of equations

- Usually we will have > 1 dependent variables, e.g.

$$\begin{aligned}\dot{x} &= f(x, y)\\ \dot{y} &= g(x, y)\end{aligned}$$

- x and y are **coupled** together
- So we **cannot** solve the equations independently

Systems of equations

- Usually we will have > 1 dependent variables, e.g.

$$\begin{aligned}\dot{x} &= f(x, y) \\ \dot{y} &= g(x, y)\end{aligned}$$

- x and y are **coupled** together
- So we **cannot** solve the equations independently

- We need to find a numerical method that can solve the system of equations as a whole

Systems of equations

- A general system of 1st-order ODEs is

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

Systems of equations

- A general system of 1st-order ODEs is

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

- Aggregate the variables into a vector $\mathbf{x}(t) := (x_1(t), x_2(t), \ldots, x_n(t))$

Systems of equations

- A general system of 1st-order ODEs is

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

- Aggregate the variables into a vector $\mathbf{x}(t) := (x_1(t), x_2(t), \dots, x_n(t))$
- Define $\mathbf{f} := (f_1, \dots, f_n)$ – vector-valued function

Systems of equations

- A general system of 1st-order ODEs is

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

- Aggregate the variables into a vector $\mathbf{x}(t) := (x_1(t), x_2(t), \dots, x_n(t))$
- Define $\mathbf{f} := (f_1, \dots, f_n)$ – vector-valued function
- The ODE becomes $\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t))$

Systems of equations

- A general system of 1st-order ODEs is

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

- Aggregate the variables into a vector $\mathbf{x}(t) := (x_1(t), x_2(t), \dots, x_n(t))$
- Define $\mathbf{f} := (f_1, \dots, f_n)$ – vector-valued function
- The ODE becomes $\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t))$

Euler for systems of equations

- $\mathbf{x}(t) = (x_1(t), x_2(t), \ldots, x_n(t))$ if n variables

Euler for systems of equations

- $\mathbf{x}(t) = (x_1(t), x_2(t), \dots, x_n(t))$ if n variables
- Taylor expand each variable:

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

Euler for systems of equations

- $\mathbf{x}(t) = (x_1(t), x_2(t), \dots, x_n(t))$ if n variables
- Taylor expand each variable:

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

- So we obtain a vector version of the Euler method

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

Euler for systems of equations

- $\mathbf{x}(t) = (x_1(t), x_2(t), \dots, x_n(t))$ if n variables
- Taylor expand each variable:

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

- So we obtain a vector version of the Euler method

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

- We can implement this with the *same* method, but now with vectors!
- We should re-use the *same* code!

Higher derivatives

- How can we treat higher-order ODEs (with higher derivatives)?

Higher derivatives

- How can we treat higher-order ODEs (with higher derivatives)?
- E.g. a damped harmonic oscillator

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

Higher derivatives

- How can we treat higher-order ODEs (with higher derivatives)?
- E.g. a damped harmonic oscillator

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

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

Reduction to a system of 1st-order equations

- Damped harmonic oscillator example:

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

Reduction to a system of 1st-order equations

- Damped harmonic oscillator example:

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

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

Reduction to a system of 1st-order equations

- Damped harmonic oscillator example:

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

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

- So we get the equivalent *system* of 1st-order equations

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

Collaboration III

An alternative viewpoint

Consider the ODE $\dot{x} = f(x)$.

1. Which operation could we use to remove the derivative?
2. Which theorem helps with this and what does it say?
3. Use this to rewrite the ODE $\dot{x} = f(x)$ without the derivative.

ODEs as integrals

- Since there is a derivative in an ODE, it is natural to try to remove it by *integrating*

ODEs as integrals

- Since there is a derivative in an ODE, it is natural to try to remove it by *integrating*
- Let's integrate both sides of the ODE

$$\dot{x}(t) = f(x(t))$$

with respect to time:

ODEs as integrals

- Since there is a derivative in an ODE, it is natural to try to remove it by *integrating*
- Let's integrate both sides of the ODE

$$\dot{x}(t) = f(x(t))$$

with respect to time:

$$\int_{t_0}^{t_1} \dot{x}(s)\,\mathrm{d}s = \int_{t=t_0}^{t_1} f(x(s))\,\mathrm{d}s$$

ODEs as integrals II

- The Fundamental Theorem of Calculus tells us that integration is the inverse operation to differentiation. Hence

$$\int_{t_0}^{t_1} \dot{x}(s)\,\mathsf{d}s = x(t_1) - x(t_0)$$

ODEs as integrals II

- The Fundamental Theorem of Calculus tells us that integration is the inverse operation to differentiation. Hence

$$\int_{t_0}^{t_1} \dot{x}(s)\,\mathsf{d}s = x(t_1) - x(t_0)$$

- Thus

$$x(t_1) = x(t_0) + \int_{t_0}^{t_1} f(x(s))\,\mathsf{d}s$$

ODEs as integrals II

- The Fundamental Theorem of Calculus tells us that integration is the inverse operation to differentiation. Hence

$$\int_{t_0}^{t_1} \dot{x}(s)\,\mathsf{d}s = x(t_1) - x(t_0)$$

- Thus

$$x(t_1) = x(t_0) + \int_{t_0}^{t_1} f(x(s))\,\mathsf{d}s$$

- This seems useless since we still have an equation for x in terms of itself!

ODEs as integrals II

- The Fundamental Theorem of Calculus tells us that integration is the inverse operation to differentiation. Hence

$$\int_{t_0}^{t_1} \dot{x}(s)\,\mathsf{d}s = x(t_1) - x(t_0)$$

- Thus

$$x(t_1) = x(t_0) + \int_{t_0}^{t_1} f(x(s))\,\mathsf{d}s$$

- This seems useless since we still have an equation for x in terms of itself!

Approximating the integral: Euler again

- What is the simplest way to approximate the integral?

Approximating the integral: Euler again

- What is the simplest way to approximate the integral?
- Approximate $x(s) \simeq x(t_0)$ for *all* $s \in [t_0, t_1]$

Approximating the integral: Euler again

- What is the simplest way to approximate the integral?
- Approximate $x(s) \simeq x(t_0)$ for *all* $s \in [t_0, t_1]$
- So $x_1 - x_0 = \int_{t_0}^{t_1} f(x_0)\, ds = hf(x_0)$
- So Euler is just the rectangular rule for integration!

Improving over Euler

- We will see in the problem set that… Euler is *bad*

Improving over Euler

- We will see in the problem set that… Euler is *bad*
- How could we improve it?

Summary

- ODEs: the solution is a **function**
- Equivalent viewpoint of ODEs as integral equationsw

- Numerical methods
- Approximate solutions via **time stepping**
- Euler method

- Higher-order ODEs as systems of 1st-order ODEs