Computational Physics
Lecture 10 – Introduction to Finite Difference Methods

Pat Scott

Department of Physics, Imperial College

October 30, 2018

Slides available from https://bb.imperial.ac.uk/

Outline

Goals

By the end of today's lecture, you should

- Be able to calculate the derivative of a function numerically (i.e. from the function's values only, without an analytical expression for the derivative)
- Understand what a finite difference scheme is
- Have a feeling for how to build finite difference schemes from each other
- Be able to identify different ODE problems as either initial value or boundary value
- Be able to implement the simplest finite difference scheme (Euler's method) numerically

Outline

The simplest option: forward difference

The standard definition of the derivative dy/dx or $y'(x)$ is

$$\frac{dy(x)}{dx} \equiv \lim_{h\to 0} \frac{y(x+h)-y(x)}{h} \equiv \lim_{\Delta x\to 0} \frac{\Delta y}{\Delta x} \,. \tag{1}$$

We can implement

$$\tilde{y}'_f(x) \equiv \frac{y(x+h)-y(x)}{h} \,. \tag{2}$$

directly on a computer, and take h to be pretty small

→ this is a **forward difference scheme (FDS)**

→ example of a **finite difference approximation**

The simplest option: forward difference

The standard definition of the derivative dy/dx or $y'(x)$ is

$$\frac{dy(x)}{dx} \equiv \lim_{h\to 0} \frac{y(x+h)-y(x)}{h} \equiv \lim_{\Delta x\to 0} \frac{\Delta y}{\Delta x} \,. \qquad (1)$$

We can implement

$$\tilde{y}'_f(x) \equiv \frac{y(x+h)-y(x)}{h} \,. \qquad (2)$$

directly on a computer, and take h to be pretty small
→ this is a **forward difference scheme (FDS)**
→ example of a **finite difference approximation**

Note notation: $\sim$ indicates 'numerical approximation to'

The simplest option: forward difference

$$\tilde{y}'_f(x) \equiv \frac{y(x+h) - y(x)}{h} \tag{3}$$

Problems: we can't take h to zero numerically, for two reasons:

- divide by zero
- finite accuracy of the difference in the numerator

This induces an error in our estimate of $y'(x)$. To see what sort of error, Taylor expand $y(x+h)$ around x:

$$y(x+h) = y(x) + y'(x)\,h + \frac{y''(x)}{2}\,h^2 + \cdots\,, \tag{4}$$

Rearrange to get $y'(x)$:

$$y'(x) = \frac{y(x+h) - y(x)}{h} - \frac{y''(x)}{2}\,h + \ldots = \tilde{y}'_f(x) - \mathcal{O}(h)\,. \tag{5}$$

The simplest option: forward difference

$$\tilde{y}'_f(x) \equiv \frac{y(x+h) - y(x)}{h} \tag{3}$$

Problems: we can't take h to zero numerically, for two reasons:

- divide by zero
- finite accuracy of the difference in the numerator

This induces an error in our estimate of $y'(x)$. To see what sort of error, Taylor expand $y(x+h)$ around x:

$$y(x+h) = y(x) + y'(x)\,h + \frac{y''(x)}{2}\,h^2 + \cdots\,, \tag{4}$$

Rearrange to get $y'(x)$: $\Longrightarrow$ truncation error is $\mathcal{O}(h)$.

$$y'(x) = \frac{y(x+h) - y(x)}{h} - \frac{y''(x)}{2}\,h + \ldots = \tilde{y}'_f(x) - \mathcal{O}(h)\,. \tag{5}$$

The simplest option: forward difference

$$\tilde{y}'_f(x) \equiv \frac{y(x+h) - y(x)}{h} \tag{3}$$

Problems: we can't take h to zero numerically, for two reasons:

- divide by zero
- finite accuracy of the difference in the numerator

This induces an error in our estimate of $y'(x)$. To see what sort of error, Taylor expand $y(x+h)$ around x:

$$y(x+h) = y(x) + y'(x)\,h + \frac{y''(x)}{2}\,h^2 + \cdots\,, \tag{4}$$

Rearrange to get $y'(x)$: $\Longrightarrow$ truncation error is $\mathcal{O}(h)$. Meaning?

$$y'(x) = \frac{y(x+h) - y(x)}{h} - \frac{y''(x)}{2}\,h + \ldots = \tilde{y}'_f(x) - \mathcal{O}(h)\,. \tag{5}$$

Backwards difference scheme

This was the forwards difference scheme:

$$\tilde{y}'_f(x) \equiv \frac{y(x+h) - y(x)}{h} . \tag{6}$$

We can construct an equivalent-order scheme in the opposite (backwards) direction too:

$$\tilde{y}'_b(x) \equiv \frac{y(x) - y(x-h)}{h} . \tag{7}$$

Two different answers with the same order error.

Backwards difference scheme

This was the forwards difference scheme:

$$\tilde{y}'_f(x) \equiv \frac{y(x+h) - y(x)}{h} \,. \tag{6}$$

We can construct an equivalent-order scheme in the opposite (backwards) direction too:

$$\tilde{y}'_b(x) \equiv \frac{y(x) - y(x-h)}{h} \,. \tag{7}$$

Two different answers with the same order error. Which to choose?

Central difference scheme

Two answers at the same order...

$\Longrightarrow$ we can average them to bootstrap our way up to the next level of accuracy:

$$\tilde{y}'_c(x) \equiv \frac{\tilde{y}'_f(x) + \tilde{y}'_b(x)}{2} = \frac{y(x+h) - y(x-h)}{2h}. \tag{8}$$

This is the **central difference scheme.**

Central difference scheme

$$\tilde{y}'_c(x) \equiv \frac{\tilde{y}'_f(x) + \tilde{y}'_b(x)}{2} = \frac{y(x+h) - y(x-h)}{2h} \,. \tag{9}$$

To see the order of the error, expand to 3rd order:

$$y(x+h) \quad = y(x) + y'(x)\,h + \tfrac{1}{2}y''(x)\,h^2 + \tfrac{1}{3!}y'''(x)\,h^3 + \ldots \tag{10}$$
$$y(x-h) \quad = y(x) - y'(x)\,h + \tfrac{1}{2}y''(x)\,h^2 - \tfrac{1}{3!}y'''(x)\,h^3 + \ldots \tag{11}$$

We can use this to define

$$y(x+h) - y(x-h) = 2\,y'(x)\,h + \mathcal{O}(h^3), \tag{12}$$

so

$$y'(x) = \frac{y(x+h) - y(x-h)}{2h} - \mathcal{O}(h^2) \equiv \tilde{y}'_c(x) - \mathcal{O}(h^2), \tag{13}$$

Central difference scheme

$$\tilde{y}'_c(x) \equiv \frac{\tilde{y}'_f(x) + \tilde{y}'_b(x)}{2} = \frac{y(x+h) - y(x-h)}{2h}\,. \tag{9}$$

To see the order of the error, expand to 3rd order:

$$\begin{aligned} y(x+h) &= y(x) + y'(x)\,h + \tfrac{1}{2}y''(x)\,h^2 + \tfrac{1}{3!}y'''(x)\,h^3 + \ldots \quad (10)\\ y(x-h) &= y(x) - y'(x)\,h + \tfrac{1}{2}y''(x)\,h^2 - \tfrac{1}{3!}y'''(x)\,h^3 + \ldots \quad (11)\end{aligned}$$

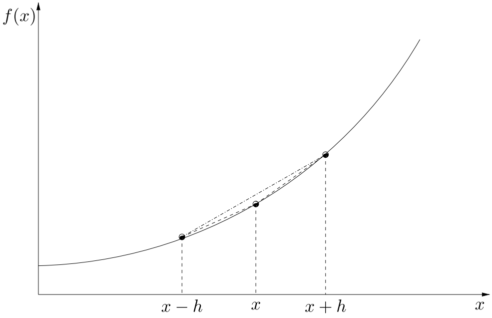
We can use this to define

$$y(x+h) - y(x-h) = 2\,y'(x)\,h + \mathcal{O}(h^3), \tag{12}$$

so $\Longrightarrow$ truncation error is $\mathcal{O}(h^2)$.

$$y'(x) = \frac{y(x+h) - y(x-h)}{2h} - \mathcal{O}(h^2) \equiv \tilde{y}'_c(x) - \mathcal{O}(h^2), \tag{13}$$

Comparison of schemes



In general, we can build finite difference combinations to get any higher-order derivative to any accuracy

$\rightarrow$ requires using y at more values of x

$\rightarrow$ $\tilde{y}^n$ requires $n+1$ points

$\rightarrow$ compare with cubic splines: $\tilde{y}''$ with 4 points

Outline

The problem

Solve

$$\frac{\mathrm{d}y}{\mathrm{d}x} = f(x, y) \tag{14}$$

for y(x).

Working with a single first order ODE

What we have:

1. The derivative function $\frac{\mathrm{d}y}{\mathrm{d}x} \equiv f(x, y(x))$
2. Some sort of boundary condition

What we really care about

1. $y(x)$

Working with a single first order ODE

What we have:

1. The derivative function $\frac{\mathrm{d}y}{\mathrm{d}x} \equiv f(x, y(x))$
2. Some sort of boundary condition

What we really care about

1. $y(x)$

Strategy: use $f(x, y)$ to gradually evolve $y(x)$

1. Start with some $y(x_\mathrm{s})$
2. Calculate the derivative $f(x_\mathrm{s}, y(x_\mathrm{s}))$
3. Use the derivative to extrapolate y some small distance Δx

$$y(x + \Delta x) \approx y(x_\mathrm{s}) + \Delta x\, f(x_\mathrm{s}, y(x_\mathrm{s})) \quad (15)$$

(Schematically; this particular setup is Euler's Method)

The (bigger) problem

Solve

$$\frac{\mathrm{d}^N y}{\mathrm{d}x^N} = f(x, y, y', y'', .., y^{(N-1)}) \tag{16}$$

for y(x).

Any *N*th order ordinary differential equation can be recast as a coupled set of *N* first order ODEs

$$\begin{aligned} y'(x) &= y_1(x, y) \\ y_1'(x) &= y_2(x, y, y_1) \\ &\ldots \\ y_{N-2}'(x) &= y_{N-1}(x, y, y_1, y_2, .., y_{N-2}) \\ y_{N-1}'(x) &= y_N \equiv f(x, y, y_1, y_2, .., y_{N-2}, y_{N-1}) \end{aligned} \tag{17}$$

OK, try it! (2 mins)

Write the ODE

$$\frac{\mathrm{d}^2 y}{\mathrm{d}x^2} + x\frac{\mathrm{d}y}{\mathrm{d}x} = y + 1 \tag{18}$$

as a system of first-order ODEs.

Hint: the general way to start is by defining each lower-order derivative as a new variable, i.e. $u \equiv \frac{\mathrm{d}y}{\mathrm{d}x}$.

So, we can write any set of N first order ODEs (such as from a higher order ODE) as a single first-order vector equation

$$\frac{\mathrm{d}y_i}{\mathrm{d}x}(x) = f_i(x, \vec{y}(x)) \tag{19}$$

where $i = 0..N-1$ and each f_i may individually depend on the full vector $\vec{y} = (y_0, y_1, ..y_{N-1})$

We hence have

1. a vector of derivatives $(\frac{\mathrm{d}y_i}{\mathrm{d}x})$, given by
2. a vector-valued function (f_i) that depends on
3. x directly, and
4. x indirectly, via a vector of auxiliary function values $\vec{y}(x)$

What we have:

1. The set of functions f_i for all i
2. Some sort of boundary conditions

What we really care about

1. $y_0(x)$
2. maybe also the rest of $\vec{y}(x)$

What we have:

1. The set of functions f_i for all i
2. Some sort of boundary conditions

What we really care about

1. $y_0(x)$
2. maybe also the rest of $\vec{y}(x)$

Strategy: use the set of f_is to gradually evolve $\vec{y}(x)$

1. Start with some $\vec{y}(x_\mathrm{s})$
2. Calculate the vector of x derivatives $\vec{f}(x_\mathrm{s}, \vec{y}(x_\mathrm{s}))$
3. Use each f_i to extrapolate each component y_i of $\vec{y}$ some small distance Δx

$$\vec{y}(x + \Delta x) \approx \vec{y}(x_\mathrm{s}) + \Delta x\, \vec{f}(x_\mathrm{s}, \vec{y}(x_\mathrm{s})) \qquad (20)$$

(Just like doing single 1st order ODE evolution, but generalising $f(x, y)$ and y to vectors – example is again Euler's method)

Main types of boundary conditions

1. $\vec{y}(x_\mathrm{s}) = y_\mathrm{start}$
   - Function and all derivatives are defined at some x_s
   - All that remains is to evolve the system forwards and/or backwards from x_s to get $\vec{y}(x)$ for all x of interest
   - This is an **Initial Value Problem**

Main types of boundary conditions

1. $\vec{y}(x_s) = y_{\text{start}}$
   - Function and all derivatives are defined at some x_s
   - All that remains is to evolve the system forwards and/or backwards from x_s to get $\vec{y}(x)$ for all x of interest
   - This is an **Initial Value Problem** – covered in Lecture 12

Main types of boundary conditions

1. $\vec{y}(x_s) = y_{\text{start}}$
   - Function and all derivatives are defined at some x_s
   - All that remains is to evolve the system forwards and/or backwards from x_s to get $\vec{y}(x)$ for all x of interest
   - This is an **Initial Value Problem** – covered in Lecture 12
2. Some components of $\vec{y}$ may be specified at one x_s, others at one (or more) other value(s) of x
   - Not all components may be specified
   - Some may be specified at multiple values of x
   - Gotta set a few, guess, poke around a bit...

Main types of boundary conditions

1. $\vec{y}(x_\mathrm{s}) = y_\mathrm{start}$
   - Function and all derivatives are defined at some x_s
   - All that remains is to evolve the system forwards and/or backwards from x_s to get $\vec{y}(x)$ for all x of interest
   - This is an **Initial Value Problem** – covered in Lecture 12
2. Some components of $\vec{y}$ may be specified at one x_s, others at one (or more) other value(s) of x
   - Not all components may be specified
   - Some may be specified at multiple values of x
   - Gotta set a few, guess, poke around a bit...
   - Hrmmm, nasty – let's put this off until Lecture 14.

Main types of boundary conditions

1. $\vec{y}(x_s) = y_{\rm start}$
   - Function and all derivatives are defined at some x_s
   - All that remains is to evolve the system forwards and/or backwards from x_s to get $\vec{y}(x)$ for all x of interest
   - This is an **Initial Value Problem** – covered in Lecture 12
2. Some components of $\vec{y}$ may be specified at one x_s, others at one (or more) other value(s) of x
   - Not all components may be specified
   - Some may be specified at multiple values of x
   - Gotta set a few, guess, poke around a bit...
   - Hrmmm, nasty – let's put this off until Lecture 14.
3. Some other more complicated auxiliary condition must be satisfied
   - e.g. $n_{\rm cookies}(\rm today) > n_{\rm critical}$ AND $n_{\rm cookies}(\rm yesterday) - n_{\rm cookies}(\rm today) < \mathit{MaxDailyConsumption}$

These last two are examples of **Boundary Value Problems**

Outline

Simplest method is just to use the forward difference method directly:

$$y(x+\Delta x) \approx y(x_\mathrm{s}) + \Delta x\, f(x_\mathrm{s}, y(x_\mathrm{s})) \tag{21}$$

i.e.

$$y(x+h) \approx y(x) + h\, f(x, y(x)) \tag{22}$$

or (note that subscripts denote *steps* here, not indices distinguishing N coupled DEs):

$$\tilde{y}_{n+1} = \tilde{y}_n + f(x_n, \tilde{y}_n)\, h. \tag{23}$$

Some examples of Euler method in action: Jupyter notebook

Housekeeping

- Problem Sheet: combined for Lecs 10–15. Mostly complementary to Project, for extending your understanding, less about preparing you for it (i.e. unlike previous Problem Sheets).
- Reminder: get started on **Assignment** nice and early – due in on **Nov 12**
- Labs continue tomorrow at 9am. Come and ask questions about the Problem Sheets or Assignment.
- Next Tuesday: Evaluating Finite Difference Methods (bring laptops + install Jupyter!)