Lecture 11: rounding errors + solving IVPs with FDs

– Computers can only store numbers within a limited precision. Therefore, for most real numbers such as $1/3$, π and $\sqrt{2}$, there will always be a round-off error (the difference between the actual value and the computer representation).

Example: Euler's number (e)

$$e = \lim_{x \to 0} (1+x)^{1/x}$$

– To compute e very accurately, we can take x very small. However, if x gets close to the so-called machine accuracy, we will lose precision: the computer will round $1+x$ to 1.

• Rounding errors in numerical differentiation

– Let's say we want to use, e.g., a forward FD to approximate some $y'(t)$:

$$y'(t) \approx \frac{\Delta y}{\Delta t} = \frac{y(t+h) - y(t)}{h}$$

– In practice, however, we can only compute Δy up to machine accuracy $\varepsilon_m \approx 10^{-16}$.
Therefore, what we actually get is

$$\underset{\text{ideal}}{\frac{\Delta y}{h}} \longrightarrow \overbrace{\frac{\tilde{\Delta} y}{h}}^{\text{rounded difference}}_{\text{real}} = \frac{\overbrace{\Delta y}^{\text{exact difference}} \pm \overbrace{\Theta(\varepsilon_m)}^{\text{round-off}}}{h} = \frac{\Delta y}{h} \pm \Theta\left(\frac{\varepsilon_m}{h}\right)$$

→ Remember: for the forward FD, we have $y'(t) - \frac{\Delta y}{h} = \Theta(h)$ (1st-order convergence)

Hence: $\frac{\tilde{\Delta} y}{h} = \underbrace{y' - \Theta(h)}_{\Delta y/h} \pm \Theta\left(\frac{\varepsilon_m}{h}\right) \Rightarrow \underbrace{y' - \frac{\tilde{\Delta} y}{h}}_{\text{actual error}} = \underbrace{\Theta(h)}_{\text{truncation error}} \pm \underbrace{\Theta\left(\frac{\varepsilon_m}{h}\right)}_{\text{rounding error}}$

— We see that the total error is composed of a truncation error (numerical) and a rounding error. As $h \to 0$, the former decreases, while the latter increases. As long as h is large enough, the rounding error $\Theta(\varepsilon_m/h)$ will be negligible. As we reduce h, however, the two errors will at some point become of comparable magnitude! This will happen when

$$\underbrace{\Theta(h)}_{\text{truncation}} = \underbrace{\Theta\left(\frac{\varepsilon_m}{h}\right)}_{\text{rounding}}, \text{ that is, when } \Theta(h^2) = \Theta(\varepsilon_m), \text{ or } \boxed{h = \Theta(\sqrt{\varepsilon_m})}$$

For the standard $\varepsilon_m = 10^{-16}$, we would then expect the convergence to break down when $h = \Theta(\sqrt{10^{-16}}) = \Theta(10^{-8})$. Mind that this result is specific to the forward/backward formulas. What if we use the central formula?

– In that case, we have second-order (theoretical) convergence, as seen previously. Thus, we have a truncation error of $\Theta(h^2)$, and a rounding error of $\Theta(\varepsilon_m/h)$. Those will become comparable when

$$\Theta(h^2) = \Theta(\varepsilon_m/h),\ \text{that is, } \Theta(h^3) = \Theta(\varepsilon_m),\ \text{or } \boxed{h = \Theta\left(\sqrt[3]{\varepsilon_m}\right)}$$

→ For $\varepsilon_m = 10^{-16}$, the convergence should break down when $h = \Theta(10^{-5.33})$

▸ General case

– In general, an FD formula to approximate the n-th derivative can be written as

$$y^{(n)}(t) = \frac{\sum_{i=0}^{N} \alpha_i y_i}{h^n} + \Theta(h^p)$$

– Since round-off happens in the numerator, we will have

$$y^{(n)} = \frac{\sum_{i=0}^{N} \alpha_i y_i + \Theta(\varepsilon_m)}{h^n} + \Theta(h^p)$$

rounding error

Therefore, the convergence should break down when

$$\Theta\left(\frac{\varepsilon_m}{h^n}\right) = \Theta(h^p), \text{ that is, when } \boxed{h = \Theta\left(\varepsilon_m^{\frac{1}{n+p}}\right)}$$

<u>Example</u>: $y''(t) = \dfrac{y(t+h) - 2y(t) + y(t-h)}{h^{2}} + \Theta(h^{2})$ n p

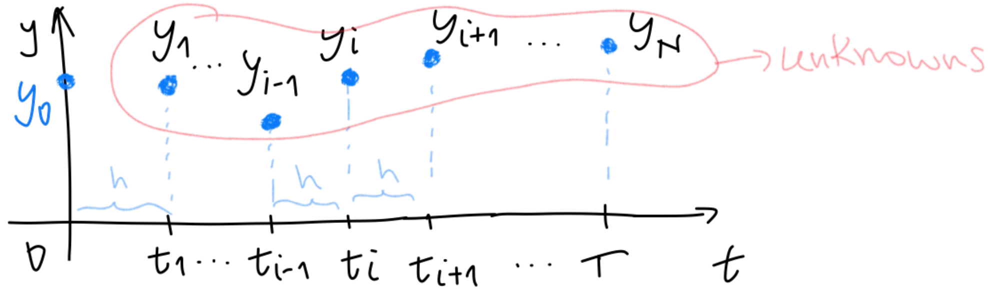
$$\Rightarrow h_{\text{break down}} = \Theta\left(\varepsilon_m^{\frac{1}{2+2}}\right) = \Theta\left(\sqrt[4]{10^{-16}}\right) = \Theta(10^{-4})$$

• Using finite differences to solve IVPs

– As an example, let's finally consider the pendulum problem:

$$y'' + a\sin y = 0, \quad a = g/L, \quad y(0) = y_0, \quad y'(0) = v_0$$

– To use FDs, we will consider time steps:



- We know $y(0)=y_0$, and we want to compute (approximate) $y_1, y_2, \dots, y_N$.

Remember that $y'(t_i) \approx \dfrac{y_{i+1}-y_i}{h}$. Therefore:

$\overbrace{y'(0)}^{v_0} \approx \dfrac{y_1-y_0}{h}$, but $y'(0)=v_0$. Hence: $\boxed{y_1 \approx \overset{\text{given}}{y_0} + \overset{\text{given}}{h}\,\underset{\text{given}}{v_0}}$

- How do we now compute $y_2, y_3, \dots$?

→ Remember that $y''(t_i) \approx \dfrac{y_{i+1}-2y_i+y_{i-1}}{h^2}$. Insert this into the ODE at t_i:

$$\frac{y_{i+1}-2y_i+y_{i-1}}{h^2} + a\sin y_i \approx 0 \Rightarrow \underbrace{\boxed{y_{i+1} \approx 2y_i - y_{i-1} - h^2 a \sin y_i}}_{\text{general formula (algorithm)}}$$

This gives us a formula to compute the next value y_{i+1} in terms of past values y_i, y_{i-1}:

* $i=1$: $y_2 \approx 2\overset{\text{known}}{y_1} - \overset{\text{given}}{y_0} - \overset{\text{given}}{h^2}\,\overset{\text{given}}{a} \sin y_1$ = Known (because y_0 and y_1 are known at this point)

* $i=2$: $y_3 \approx 2\underset{\text{known}}{y_2} - \underset{\text{known}}{y_1} - h^2 a \sin y_2$ = Known (y_2 and y_1 are now both known)

⋮