Truncation Errors in Euler's Method

Really want to understand **global error**: the (absolute) error after k timesteps

$$GE_k = y(t_k) - y_k^{EM}$$

Use Taylor Series on $y(t_k + h)$, assuming $y \in C^2 \implies$

$$\begin{aligned} y(t_{k+1}) &= y(t_k) + hy'(t_k) + h^2 y''(\xi)/2 && (1) \\ &= y(t_k) + hf(t_k, y(t_k)) + h^2 y''(\xi)/2 && (2) \end{aligned}$$

Euler's Method is just

$$y_{k+1} = y_k + hf(t_k, y_k) \qquad (3)$$

(2)-(3) $\implies$

$$GE_{k+1} = GE_k + h[f(t_k, y(t_k)) - f(t_k, y_k)] + h^2 y''(\xi)/2 \qquad (4)$$

Mean Value Theorem in second argument $\implies$

$$f(t_k, y(t_k)) - f(t_k, y_k) = \frac{\partial f}{\partial y}|_\xi \, (y(t_k) - y_k) \tag{5}$$
$$\equiv J(\xi) GE_k \tag{6}$$

or could use Lipshitz bound

$$f(t_k, y(t_k)) - f(t_k, y_k) \leq L \; GE_k$$

(4) and (6) $\implies$

$$GE_{k+1} = (1 + hJ) GE_k + h^2 y''(\xi)/2 \tag{7}$$

or

$$GE_{k+1} \leq (1 + hL) GE_k + h^2 y''(\xi)/2$$

Error propagation

In words, this difference equation says
Global error at t_{k+1} = the Global error at t_k, propagated with factor $(1 + hJ)$
plus the new error (the local error) made at each step

$(1 + hJ)$ determines the **error propagation** at finite h
$\rightarrow$ the **numerical stability of method** cf. `BadRecurrence.m`
We want numerical errors to dampen out, if possible,
$\implies$ we want $|\, 1 + hJ \,| < 1$
Note: since $h > 0$, errors must grow if $J > 0$

Diverging solutions

If $J > 0 \implies$ nearby solutions are diverging

Absolute errors will grow in a region where $J > 0$

If J is not too large, IVP may still be fairly well-conditioned.

If the solution is also growing, relative errors may be OK, even if absolute errors are growing

$\implies$ may get useful information from numerical solution

If $J \gg 0$, IVP is ill-conditioned

$\implies$ don't expect accurate numerical solution, using any method!

More precisely: if $J(t_f - t_0) \gg 1$

Contractive solutions

If $J < 0 \implies$ nearby solutions are contractive
In this case, the local error made at each step is damped thereafter
Sketch:

We aim for accurate solutions in this case.

Convergence of Euler's Method

Convergence determined by what happens to the error as $h \to 0$
Numerical stability determined by what happens to the error at finite h.

We would like to prove that $GE_k \to 0$ as $h \to 0$.

i.e. that the Forward Error $\to 0$ as $h \to 0$.
We do it in 2 steps:

1. show the Backward Error (residual) $\to 0$ as $h \to 0$: **consistency**
2. show the condition number of the **difference equation** stays bounded as $h \to 0$: **0-stability**

then use a Big Theorem

$$\boxed{\textbf{Consistency + Stability} \to \textbf{Convergence}}$$

that is true for ODE methods and time-dependent linear PDE methods!

For ODEs: any sensible method is consistent but not all are 0-stable!

Consistency

A method is **consistent** if the exact solution satisfies the difference equation in the limit $h \to 0, n \to \infty$ with $t = t_0 + nh$ fixed.

To check this, we find the **residual** R by subsitituting the exact solution into the method:

for Euler's Method

$$R = y(t_{n+1}) - y(t_n) - hf(t_n, y(t_n)) = \frac{1}{2}h^2 y''(\xi)$$

R is called the **local truncation error LTE**.
For 1-step methods, the LTE is the same as the Local Error.

Backward error

We want this residual to measure the backward error: is the exact solution a solution of the difference equation with a different f? The answer is YES if we replace f with $f + \frac{R}{h}$ — so the backward error is really R/h.

Definition

A method is **consistent** if $\lim_{h\to 0} \frac{R}{h} = 0$

In our case, $R/h \sim h \to 0 \implies$ Euler's Method is consistent.

Definition

A method is **consistent of order p** if $\frac{R}{h} = O(h^p)$ as $h \to 0$

$\implies$ Euler's Method is consistent of order 1.

0-stability

Check that the difference equation doesn't go ill-conditioned as $h \to 0$.

Suppose we change initial condition by δ_0 and at step n change the right hand side (RHS) by δ_n,

$$u_{n+1} = u_n + hf(t_n, u_n); u_0 = A$$

$$v_{n+1} = v_n + h[f(t_n, v_n) + \delta_n]; v_0 = A + \delta_0$$

is the change in the solution $v - u$ finite?
Can show (by induction) that

$$|v - u|_n \leq e^{LT}\delta_0 + \frac{e^{LT} - 1}{L}\delta$$

where $T = t_n - t_0$, assuming $|\delta_k| \leq \delta$ no h dependence!
$\implies$ change remains finite as $h \to 0$ so Euler's Method is 0-stable.

Hence convergence

Now combine these two results to show that $|GE_n| \to 0$ as $h, \delta_0 \to 0$.

Definition

A method is **convergent of order** p if $\max |GE_n| = O(h^p)$ as $h, \delta_0 \to 0$

Theorem

A method that is consistent of order p and 0-stable is convergent of order p.

Sketch of proof:
the backward error R/h plays the role of the perturbation δ. Therefore if the discrete condition number stays finite (0-stability) and $R/h = O(h^p)$ as $h \to 0$ (consistent), then as $h, \delta_0 \to 0 \implies |GE_n| = O(h^p)$ (convergent)

An error bound

For Euler's Method, $R/h = \frac{1}{2} h y''(\xi)$
$\implies \delta = \frac{1}{2} hM$ where $|y''(\xi)| \leq M$, assuming $y \in C^2$.
So, for $\delta_0 \to 0$

$$| GE_n | \leq \frac{hM}{2L}[e^{LT} - 1] = Kh$$

an *a priori* error bound! i.e. global error is $O(h)$

Euler's Method is convergent of order 1 (a 'first order method')

Hence if we halve h, expect GE to halve, as we observed.

The bound Kh depends sensitively on LT— this bound is rigorous but very pessimistic: doesn't distinguish between $J \gg 0$ and $J \ll 0$ (both have same Lipshitz constant $L = |J|$)

A tighter bound

We can do better if $J < 0$ (solutions are contractive)

$$| \, GE_n \, | \leq \frac{hM}{2} T = Kh$$

provided h satisfies

$$| \, 1 + hJ \, | < 1$$

i.e. numerical errors are damped

BUT if $J < 0$ and $| \, 1 + hJ \, | > 1$
we get numerical error growth for a well-conditioned IVP!
— a classic case of **numerical instability**

Interval of absolute stability

The interval where Euler Method is numerically stable is called the **interval of absolute stability**

$$| 1 + hJ | < 1$$

$\Longrightarrow$

$$-2 < hJ < 0$$

Only satisfied if $J < 0$ (solutions are contractive) and

$$h < 2/ | J |$$

$\Longrightarrow$ there is an upper bound on stepsize h (depending on J) to ensure numerical stability.
This can limit the efficiency if h is forced to be very small!

Stiff problems

If $J \ll 0 \implies$ need **very small stepsize** to satisfy stability requirement
$\implies$ stepsize restricted by stability not truncation error!

An ODE where the choice of stepsize is restricted by stability, not by accuracy (truncation error) because $J(t_f - t_0) \ll -1$ is called **stiff**.

Usually a problem is stiff if it has **sharp transients**.
Stiff problems require special methods to get around stability restrictions!

Summary: role of J

$J > 0$ nearby solutions diverge		$J < 0$ nearby solutions converge	
$J(t_f - t_0) \gg 1$	$J(t_f - t_0) \approx 1$	$J(t_f - t_0) \approx -1$	$J(t_f - t_0) \ll -1$
don't expect good num. sol.	should be OK esp. rel. error	$h > 2/\lvert J\rvert \implies$ Euler unstable	stiff: need special methods

Effect of roundoff?

In the error bound for Euler's Method (assuming worst case for roundoff), just replace $\frac{hM}{2}$ with

$$\frac{hM}{2} + K_2 \frac{u}{h}$$

$\implies$ an optimal stepsize, as in `ForwardDifference.m`,

$$h_{\text{opt}} \approx K u^{1/2}$$

If roundoff errors add randomly: get extra error due to roundoff $\sim u n^{1/2} \sim u h^{-1/2}$ (instead of u/h)
$\rightarrow$ an optimal stepsize, with

$$h_{\text{opt}} \approx K u^{2/3}$$

Usually truncation errors dominate in solving ODEs - just remember : don't set h too low!

Summary of Euler's Method properties

Global error	$\leq Bh = O(h)$
Local error	$O(h^2)$
Stability interval	$-2 < hJ < 0$
Roundoff	$\implies h_{opt} \approx u^{2/3}$

i.e. a convergent first order method

finite stability interval $\implies$ suitable for nonstiff problems

Beyond Euler's method

Although it's easy to understand, Euler's method is not accurate enough to be useful for ODEs.

Definition

A method is of **pth order** if local error $= O(h^{p+1})$

$\implies$ global error $= O(h^p)$ (provided it's numerically stable)

Hence Euler's method is a **first order method**

We press on to look for **higher-order methods**

Is higher order worthwhile?

Set absolute tolerance τ and suppose $t_f - t_0 = T$

For Euler's Method: Global Error $\sim h \implies h \sim \tau$
number of function evals $= 1$ per step $\times n$ steps $= \frac{T}{h} \sim T\tau^{-1}$

For RK2 (second order method): Global Error $\sim h^2 \implies h \sim \tau^{1/2}$

number of function evals $= 2$ per step $\times n$ steps $= \frac{2T}{h} \sim 2T\tau^{-1/2}$
RK2 is more efficient (fewer function evaluations) if $2T\tau^{-1/2} < T\tau^{-1}$
$\implies \tau < 1/4$

Moral: For small enough tolerances, a higher order method is worthwhile

There are 2 broad classes of higher order methods: 1-step and multistep
For each class, there are:

- **explicit methods** : easier to program; relatively cheap per step
  fine for nonstiff problems
- **implicit methods** : harder to program; relatively expensive per step
  but some can solve stiff ODEs

→ 4 kinds of higher-order methods

1. explicit 1-step methods (**Runge-Kutta methods**)
2. explicit multistep methods (**Adams methods**)
3. multistep stiff solvers (**BDF methods**)
4. 1-step stiff solvers (**Implicit RK methods**) — **beyond this subject**

Runge-Kutta methods

Let's start from the quadrature rule derivation of Euler's method and try to improve it:

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f(\tau, y(\tau))\, d\tau$$

and approximate it, not by LH rectangle rule, but by trapezoid rule (quadrature rule with truncation error $\sim h^3$)

$$y_{n+1} = y_n + \frac{h}{2}[f(t_n, y_n) + f(t_{n+1}, y(t_n + h))] + O(h^3)$$

but we don't know $y(t_n + h)$! Since we already have an error $\sim h^3$, it's good enough to approximate it with error $\sim h^2$!

$\implies$ we can use Euler to get $Y (\equiv y(t_n + h)$ on RHS):

$$Y = y_n + hf(t_n, y_n) + O(h^2)$$

Explicit trapezoid method

Putting this together:

$$y_{n+1} = y_n + \frac{h}{2}[f(t_n, y_n) + f(t_{n+1}, y_n + hf(t_n, y_n))] + O(h^3)$$

which can be organized sequentially in terms of **slopes** s_i as

$$\begin{aligned} s_1 &= f(t_n, y_n) \\ s_2 &= f(t_n + h, y_n + hs_1) \\ y_{n+1} &= y_n + \frac{h}{2}[s_1 + s_2] \end{aligned}$$

This method:

- is explicit: can compute s_1, s_2, y_{n+1} in turn
- uses 2 function evaluations per step: 1 per **stage**
- tries to get a better idea of the slope field $f(t, y)$ by sampling at points in $[t_n, t_{n+1}]$

End of Lecture 20

End of Week 10