Last class: Euler's method

$$u_{i+1} = u_i + h\, f(t_i, u_i)$$

Local truncation error

$u' = f(t,u)$, substitute into

$$\frac{u_{i+1} - u_i}{h} - f(t_i, u_i) = 0$$

$$\frac{u(t_i) + u'(t_i)h + \frac{u''(\eta_i)h^2}{2} - u(t_i)}{h} - u'(t_i) = \frac{u''(\eta_i)}{2} h$$

$\uparrow$

$-\tau_i \qquad O(h)$

Consistent: $\tau_i \to 0$ as $h \to 0$

error: $e_i = u_i - u(t_i)$

$$\|e\|_\infty = \max_{0 \le i \le M} |e_i|$$

Convergent: $\|e\|_\infty \to 0$ as $h \to 0$.

Saw anecdotal evidence $\|e\|_\infty = O(h)$

Let's prove this for

$$u' = \lambda u$$

$$u(t_0) = u_0$$

$$u_{i+1} = u_i + h\lambda u_i$$

$$= (1 + h\lambda) u_i$$

$$u(t_{i+1}) = u(t_i) + hf(t_i, u(t_{i+1})) - h\tau_i$$

$$= u(t_i) + h\lambda u(t_{i+1}) - h\tau_i$$

$$e_{i+1} = e_i + \lambda h e_i + h\tau_i$$
$$= (1+\lambda h) e_i + h\tau_i$$

Interpretation: error at the next time step comes from two parts

1) propagation of error from previous time step: $(1+\lambda h)e_i$

2) new error from local truncation: $h\tau_i$

e_0 initial error (e.g. rounding from initial condition)

$$e_1 = (1+\lambda h) e_0 + h\tau_0$$
$$e_2 = (1+\lambda h) e_1 + h\tau_1$$
$$= (1+\lambda h)^2 e_0 + (1+\lambda h) h\tau_0 + h\tau_1$$
$$e_3 = (1+\lambda h)^3 e_0 + (1+\lambda h)^2 h\tau_0 + (1+\lambda h) h\tau_1 + h\tau_2$$
$$= (1+\lambda h)^3 e_0 + \sum_{k=0}^{2} (1+\lambda h)^k h\tau_{2-k}$$
$$e_M = (1+\lambda h)^M e_0 + h\sum_{k=0}^{M-1} (1+\lambda h)^k \tau_{M-k}$$

So we get contributions from initial error, plus each local truncation, each scaled by $(1+\lambda h)^j$ $0 \le j \le M$.

Suppose we can find K independent of h such that

$$\left|(1+\lambda h)^j\right| \le K \quad \text{for} \quad 0 \le j \le M.$$

$$|e_k| \le K|e_0| + K h \sum_{j=0}^{k-1} |\tau_j|$$

$$\le K|e_0| + K \max|\tau_j| \, h M$$

$$= K|e_0| + K T \max|\tau_j|$$

$$\|e\|_\infty \le K\left[|e_0| + K T \max_{0 \le j < M} |\tau_j|\right]$$

$$\tau_j \to 0$$

If $|e_0| \to 0$ then $\|e\|_\infty \to 0$

and we have convergence.

If $e_0 = 0$, $\|e\|_\infty = O(h)$ since $\tau_j = O(h)$.

A more sophisticated proof, based on the same ideas, shows that $f(t,u)$ is continuous and is Lipschitz in u, then Euler's method is convergent (assume $e_0 = 0$)

and the error vanishes $O(h)$.

$$(KT) \max |\tau_j|$$

error vanishes at the same rate as the local truncation error.

Def: A finite difference method is p-th order accurate if (assuming initial error is zero) $\|e\|_\infty = O(h^p)$.
So Euler's method is first order accurate.

Now, about that K

$$\underbrace{(1+h\lambda)}^{j}$$

could be >1, so grows in j. But $0 \le j \le M$

updated to h.

$$|(1+h\lambda)^j| = |1+h\lambda|^j$$

$$\le (1+h|\lambda|)^j$$

$$\le (1+h|\lambda|)^M$$

$$= (1+h|\lambda|)^{T/h}$$

I claim $(1+h|\lambda|) \leq e^{h|\lambda|}$

$$\underbrace{1+x}_{f(x)} \leq \underbrace{e^{x}}_{g(x)} \qquad x \geqslant 0$$

$$f(0)=1 \qquad g(0)=1$$

$$f'(x)=1 \qquad g'(x)=e^{x}>1.$$

So $\quad (g-f)(0)=0$

$$(g-f)'(x) \geqslant 0 \quad \text{for } x \geqslant 0.$$

$$\Rightarrow (g-f)(x) \geq 0 \quad \text{for } x \geqslant 0.$$

$$(1+h|\lambda|)^{M} \leq e^{h|\lambda|M} = \underbrace{e^{|\lambda|T}}_{K}.$$

Heuristic: At each step we make a new error of the size of the local truncation error, $O(h^p)$, times the time step $O(h^{p+1})$.

We make this error on $\frac{T}{h}$ timesteps, to get $\frac{T}{h} O(h^{p+1}) = T\, O(h^p) = O(h^p)$ error.

So the size of the LTE "should" be roughly the size of the global error.

p'th order accurate methods arise when LTE is $O(h^p)$.

This assumes the method maintains control on the growth of the errors (our constant K independent of h). If there is no analog of K, the method can fail to be convergent even when it is consistent.

Other methods

1) Euler's method: $u'(t_i) = \dfrac{u(t_{i+1}) - u(t_i)}{h} - \dfrac{u''(\eta_i)}{2} h$

2) Backward euler $u'(t_i) = \dfrac{u(t_i) - u(t_{i-1})}{h} + \dfrac{u''(\eta_i)}{2} h$

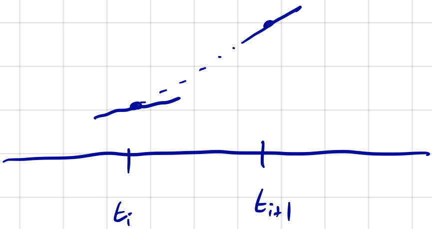
$$u(t_{i-1}) = u(t_i - h) = u(t_i) + u'(t_i)(-h) + \frac{u''(\eta_i)h^2}{2}$$

$$u'(t_i) = \frac{u(t_i) - u(t_{i-1})}{h} + \frac{u''(\eta_i)h}{2}$$

$$\frac{u(t_i) - u(t_{i-1})}{h} = f(t_i, u_i)$$

$$u(t_i) = u(t_{i-1}) + h f(\cancel{t_i}\, t_i, u_i)$$

$$u(t_{i+1}) = u(t_i) + h f(t_{i+1}, u_{i+1})$$



Now there's hard work to do to solve for u_{i+1}

because of the nonlinear equation.

Euler's method is called explicit because it gives us a formula u_{i+1} in terms of prior.

Backwards Euler is implicit, because it is not explicit.

In implementation: Matlab: fzero

python: scipy. optimize. fsolve

But this adds computation time. (Why bother?

Higher order? Nope: $O(h)$ truncation error). Stay tuned.

3) Midpoint (AKA leapfrog)

$$u'(t_i) = \frac{u(t_{i+1}) - u(t_{i-1})}{2h} + \tau_i \qquad \tau = O(h^2)$$

$$\hookrightarrow -\frac{u'''(\eta_i)}{6}h^2$$

omit

$$u(t_i) + u'(t_i)h + \frac{u''(t_i)h^2}{2} + \frac{u'''(\eta_i)h^3}{6}$$

$$- u(t_i) - u'(t_i)(-h) - \frac{u''(t_i)h^2}{2}$$

$$- \frac{u'''(\mu_i)(-h)^3}{6}$$

$$= u'(t_i)2h + \frac{1}{6}\left[u'''(\eta_i) + u'''(\mu_i)\right]h^3$$

$$u'(t_i) + \frac{1}{6}(\text{avg of } u''')h^2$$

This suggests an $O(h^2)$ method.

$O(h)$: to gain a digit of accuracy, need 10x as many steps

$O(h^2)$: to gain two digits . . . . . — — -

I

0.	0.
0.0	0.00
0.00	0.0000
0.000	0.000000
0.0000	0.00000000

in some number of time steps

Minor subtlety:

$$u(t_{i+1}) = u(t_{i-1}) + 2h f(t_i, u_i)$$

"multistep method". Information from two prior steps is used. It's still explicit, which is nice.

Need to bootstrap u_0, given and u_1, some other.

Need to pick u_1 to not spoil $O(h^2)$, and Euler's method will work: $h \cdot \tau$ error is $O(h^2)$.

We'll shortly see some undesired behavior, though.