Lecture 7, Sci. Comp. for DPhil Students II

Nick Trefethen, Thursday 13.02.20

Last lecture

- V.3 Dispersion relations and numerical instability
- V.4 Implicit 1D finite differences

Today

- V.5 Order of accuracy
- V.6 Reaction-diffusion equations and other stiff PDEs

Handouts

- Assignment 3
- Outline of lectures
- `m43_BackwardEuler.m` - accuracy of backward Euler for heat equation
- `m44_CrankNicolson.m` - accuracy of Crank-Nicolson heat equation
- `m45_leapfrog.m` - accuracy of leap frog for wave equation
- First pages of Cox-Matthews and Kassam-Trefethen papers on high-order time-stepping for stiff systems
- KdV page from the PDE Coffee Table Book ; Fornberg & Whitham 1978 on reverse
- `m46_kdv.m` - ETDRK4 code for KdV equation
- "Exponential integrators for stiff PDEs"

V.5 Order of accuracy

A couple of weeks ago we discussed order of accuracy for discretizations of ODEs, and in particular, `m27_RK4convergence.m` illustrated the fourth-order accuracy of the RK4 formula.

Now we turn to the same questions for PDEs. For example, here is the implicit "backward Euler" discretisation of $u_t = u_{xx}$. (Last lecture we did an analogous discretisation of $u_t = -u_{xxxx}$.)

$$\frac{v_j^{n+1} - v_j^n}{k} = \frac{v_{j+1}^{n+1} - 2v_j^{n+1} + v_{j-1}^{n+1}}{h^2}, \quad \text{i.e.,} \quad Bv^{n+1} = v^n$$

where $B = \text{tridiag}(-\sigma, 1 + 2\sigma, -\sigma)$, $\sigma = k/h^2$.

Here's an illustration of the first-order behaviour of this scheme.

[`m43_BackwardEuler.m`]

Here is a loose definition. Suppose we are given:

- A PDE with smooth solutions,
- A finite difference approximation with $k, h \to 0$.

It's usually simplest to separate h and k dependences and speak of, e.g.,

$$O(kh^2) \quad \text{or} \quad O(h^2 + k^2) \text{ accuracy.}$$

Algebraic determination of this accuracy goes just as with ODE:
The **local truncation error** is defined as $v_j^{n+1} - u(x_j, t_{n+1})$, where u is a smooth solution and v^{n+1} is computed from exact values v_j^n, v_j^{n-1}, etc.

Mechanically, we proceed much as for ODEs:

(1) Replace v_j^{n-1} by Taylor series for $u(x_j, t_{n-1})$, etc.,

(2) Cancel terms to find local truncation error,

(3) Divide by one power of k to find global accuracy.

Example: forward Euler for $u_t = u_{xx}$. (Backward Euler is similar.)

$$v_j^{n+1} = v_j^n + \frac{k}{h^2}(v_{j+1}^n - 2v_j^n + v_{j-1}^n)$$

Taylor series: (with $u = u(jh, nk)$ for short)

$$u((j \pm 1)h, nk) = u \pm hu_x + \frac{h^2}{2}u_{xx} \pm \dots$$

$$\implies v_{j+1}^n - 2v_j^n + v_{j-1}^n = h^2 u_{xx} + \frac{h^4}{12}u_{xxxx} + \dots.$$

From this we compute

$$v_j^{n+1} = u + ku_{xx} + \frac{kh^2}{12}u_{xxxx} + \cdots = u + ku_t + \frac{kh^2}{12}u_{tt} + \dots,$$

whereas the true value would be

$$u(x_j, t_{n+1}) = u + ku_t + \frac{k^2}{2}u_{tt} + \cdots.$$

Thus

$$v_j^{n+1} - u(x_j, t_{n+1}) = \frac{kh^2}{12}u_{tt} - \frac{k^2}{2}u_{tt} + \cdots = O(k^2 + kh^2).$$

Thus the local truncation error is $O(k^2 + kh^2)$.

This implies that the global error is $O(k + h^2)$
(assuming k is small enough for stability)

In other words, the Euler formula for $u_t = u_{xx}$ is of just first-order accuracy in time.

Backward Euler gives the same 1st-order behaviour. Not so good.

To improve this to $O(k^2+h^2)$ accuracy, we can use a formula symmetric wrt t, the **trapezoidal rule** for ODEs (which Brits call the **trapezium rule**).

In the particular application to the heat equation, this goes by the name of the **Crank-Nicolson** formula, dating to 1947:

$$v_j^{n+1} = v_j^n + \frac{k}{2h^2}(v_{j+1}^n - 2v_j^n + v_{j-1}^n) + \frac{k}{2h^2}(v_{j+1}^{n+1} - 2v_j^{n+1} + v_{j-1}^{n+1}).$$

In matrix form this becomes

$$Bv^{n+1} = Av^n$$

where

$$B = \text{tridiag}(-\sigma/2, 1+\sigma, -\sigma/2), \qquad A = \text{tridiag}(\sigma/2, 1-\sigma, \sigma/2)$$

with $\sigma = k/h^2$.

Sure enough, the accuracy is now second-order in k.

[`m44_CrankNicolson.m`]

As another example let's consider the wave equation

$$u_{tt} = u_{xx}.$$

A Crank-Nicolson-type formula would give 2nd-order accuracy. However, this PDE is not stiff, and explicit formulas are fine too. A famous one is the **leap frog** formula

$$\frac{v_j^{n+1} - 2v_j^n + v_j^{n-1}}{k^2} = \frac{v_{j+1}^n - 2v_j^n + v_{j-1}^n}{h^2},$$

that is,

$$v^{n+1} = v^{n-1} + Av^n$$

where $A = \text{tridiag}(\sigma, 2-2\sigma, \sigma)$, $\sigma = k^2/h^2$.

The symmetry in t suggests $O(k^2+h^2)$ global accuracy, and this can be confirmed by Taylor series.

Here's a demonstration:

`m45_leapfrog.m`
`m45per` - comment two lines in for periodic BCs
`m45u` - change to $k = 1.02$ for instability

V.6 Reaction-diffusion equations and other stiff PDEs

Many problems take the form

$$u_t = Lu + N(u).$$

where L is a linear differential operator and N is a nonlinear operator that may be a differential operator of lower order or not a differential operator at all. Examples include the Korteweg-de Vries (KdV), Kuramoto-Sivashinksy, FitzHugh-Nagumo, Hodgkin-Huxley, Allen-Cahn, Cahn-Hilliard, Fisher-KPP, Gray-Scott, and Navier-Stokes equations. There's a lot of science here!

Handout: "Exponential integrators for stiff PDEs"

For stability reasons, we want the L term to be discretized implicitly. On the other hand to avoid having to solve nonlinear equations at each step, we want the N term to be discretized explicitly. There has been much attention to how to achieve this balance, which, roughly speaking, is easy if you are content with 2nd-order accuracy in time but much trickier if you want higher-order accuracy.

One strategy is **ETD** or **exponential integrator** methods, which can achieve this with arbitrary order of accuracy. We won't give details.

My favourite method is called ETDRK4, by Cox and Matthews.

KdV equation: $u_t + uu_x + u_{xxx} = 0$.

Soliton solutions:

$$u(x,t) = \alpha \operatorname{sech}^2(\beta(x - ct)),$$

where $\alpha = 12\beta^2$, $c = 4\beta^2$ for any β.

Note that the speed c is proportional to the height α.

Solitons pass through one another with no lasting effect. Numerical computations were crucial in figuring this out (see e.g. Fornberg & Whitham paper, 1978).

[*PDE Coffee Table book* on KdV equation / page from Fornberg and Whitham 1978]

[Cox-Matthews / Kassam-Trefethen handout]

`m46_kdv.m` and `m46b.m` for other initial conditions

Also `spin('kdv'), spin('ac'), spin('ch'), spin2('gl'), spin2('gs'), spin2('gsspots'), spin3('gl'), spinsphere('gl')`