Lecture 2, Sci. Comp. for DPhil Students II

Nick Trefethen, Thursday 17.01.19

Last lecture

- IV.1 ODE IVPs
- IV.2 Runge-Kutta and multistep formulas
- IV.3 IVP codes in MATLAB and Simulink
- IV.4 IVP solutions in Chebfun

Today

- IV.5 Order of accuracy
- IV.6 Convergence and stability
- IV.7 Adaptive ODE codes

Assignment 1 due Tuesday 11:00.

Handout

- `m27_RK4convergence.m`, `m28_adaptiveRK.m`, `m28chebfun.m`

Numerical ODEs is a beautiful, mature subject. Here are some favorite books to pass around. (Last lecture I circulated *Exploring ODEs*, but although that is very much numerically informed, it is not a book about numerical algorithms.)

Griffiths & Higham, 2010 - introduction to numerical ODE

Iserles, 2009 - includes connection to PDEs

LeVeque, 2007 - likewise

Hairer, Norsett & Wanner I & II - authoritative; full of fun and historical remarks

Ascher & Petzold 1998 - also includes DAEs (differential-algebraic equations, which combine ODEs and nonlinear eqs)

Deuflhard & Bornemann, 2002

IV.5 Order of accuracy

Recall: we have an ODE IVP $u' = f(t,u)$, $u(0) = u_0$, which we discretize by a RK or multistep method on a grid $t_n = nk$, k = time step.

```
  |-------|-------|-------|-------|-------|-------|--
t =0    t =k    t =2k     ...
 0       1       2
```

Loose definition:

A RK or multistep formula for an IVP has ***order of accuracy*** p *if errors are* $O(k^p)$ *but not* $O(k^{p+1})$*, for sufficiently smooth problems.*

(We ignore rounding errors.)

`m27_RK4convergence.m`: illustrating the 4th-order accuracy of RK4 formula for the example

$$u' = \sin(40tu), \quad u(0) = 1$$

as k goes down to 1/128.

How do we determine order of accuracy by examining the discretization formula?

Local truncation error $v_{n+1} - u(t_{n+1})$ if u is smooth soln to ODE and v_{n+1} is computed from exact values $v_n, v_{n-1}, \ldots$.

(1) Replace v_{n-1} by Taylor series* for $u(t_n - k)$, etc.

(2) Cancel terms to find local truncation error.

(3) If L.T.E. is $O(k^{p+1})$, then order of accuracy is p.

Explanation:

```
GLOBAL ERROR  ~  LOCAL ERROR  x  NO. OF STEPS

  O(k^p)      ~   O(k^(p+1))  x   O(k^(-1))
```

- formally, i.e., without worrying about convergence

Example: 2nd-order Adams-Bashforth

$$v_{n+1} = v_n + \frac{k}{2}(3f_n - f_{n-1})$$

Taylor series (writing $u = u(t_n)$ for simplicity) :

$$u(t_{n+1}) = u + ku' + (1/2)k^2u'' + (1/6)k^3u''' + \dots$$

$$f(t_{n-1}) = u' - ku'' + (1/2)k^2u''' - \dots$$

$$\begin{aligned} v_{n+1} &= u + (k/2)[3u' - u' + ku'' - (1/2)k^2u''' + \dots] \\ &= u + ku' + (k^2/2)u'' - (1/4)k^3u''' + \dots \\ &= u(t_{n+1}) - (5/12)k^3u''' + \dots \end{aligned}$$

This implies: local truncation error $O(k^3)$, hence order 2.

IV.6 Convergence and stability

Some plausible formulas don't work at all — in fact they fail catastrophically.

Here's an example of an unstable formula with order 3 — try it and see how the computed solutions blow up as $k \to 0$!

$$v_{n+1} = -4v_n + 5v_{n-1} + k(4f_n + 2f_{n-1})$$

Famous theory of convergence of multistep formulas: Dahlquist 1956. This was one of the first major achievements of modern numerical analysis: a substantial mathematical theory closely addressing the problems that arise in machine computation. Adams, Runge, and Kutta didn't know about instability.

There are analogous results for RK.

Key definitions for a multistep formula:

consistent: order of accuracy ≥ 1

stable: if for $f(t, u) = 0$, all solns are bounded (easily checked)

convergent: $v \to u$ for each fixed t as $k \to 0$ (ignoring rounding errors)

Dahlquist Equivalence Theorem

```
 ------------------------------------------
|                                          |
| Convergence  <==>  consistency + stability |
|                                          |
 ------------------------------------------
```

The Adams formulas are consistent and stable, hence convergent.

(We should talk about the First Dahlquist Stability Barrier here, but for lack of time, we won't. This limits the attainable order of s-step stable linear multistep methods to s if they are explicit, $s+1$ if they are implicit with s odd, and $s+2$ if they are implicit with s even. The s-step Adams-Bashforth formula has order s, so it is optimal in this regard. Conversely the formula given above, being a 2-step explicit formula of order 3, could not possibly have been stable.)

IV.7 Adaptive ODE codes

ODEs, together with the related problem of numerical integration ("quadrature"), are the first great success story of **automatic** and/or **adaptive** numerical computation, dating to the 1960s and 1970s. As in many areas of computing, numerical analysts here pioneered techniques which became popular later in other areas of computer science. Indeed one might argue that the intelligent behaviour of the devices we use every day has its roots in the adaptive numerical codes of the 1960s.

Idea:

(1) User specifies tolerance

(2) Program chooses grid size and/or discretization formula to achieve this. Both may vary with t.

Method: use a local **error estimate**:

(1) Compute value v_{n+1} by two formulas of different orders.

(2) Difference in two results approximates error in low-order formula

If error is too large, refine mesh.

In MATLAB,
`ode23` compares 2nd- and 3rd-order RK formulas, and
`ode45` compares 4th- and 5th-order RK formulas.

See any of the textbooks mentioned earlier or the ODE chapter of Moler's textbook *Numerical Computing with MATLAB*, freely available online.

`m28_adaptiveRK.m` - demonstration of adaptive mesh choice for a simple ODE:

$$u' = u \tanh(\exp(t)\sin(5t))$$

`m28chebfun.m` - the same computation with Chebfun