NSDE 1: LECTURE 5

TYRONE REES[*]

Recap

$$u' = f(t,u), \qquad u(t_0) = u_0.$$

Runge-Kutta R-stage method

$$\begin{aligned}
U_{n+1} &= U_n + h\Phi(t_n, U_n; h)\\
\Phi(t,u;h) &= \sum_{r=1}^{R} c_r k_r\\
k_1 &= f(t_n, U_n)\\
k_r &= f\left(t_n + a_r\, h, U_n + h\sum_{s=1}^{r-1} b_{rs}k_s, \quad r = 2,\dots,R\right)\\
a_r &= \sum_{s=1}^{r-1} b_{rs}, \quad r = 2,\dots,R
\end{aligned}$$

This is usually written in the form of a Butcher table:

$$\begin{array}{c|c} a & B \\ \hline & c^T \end{array}$$

Last time we saw that, for $R = 2$, we require that the coefficient satisfy

$$1/2 = c_2 a_2 = c_2 b_{21},$$

which tells us we should take $b_{21} = a_2$, $c_2 = 1/2a_2$, and $c_1 = 1 - 1/(2a_2)$. We still have a free parameter, a_2, which can take any value and still give a second order method. (Note that no choice of parameters will, in general, give a third order method).

Popular choices are:

$a_2 = 1/2$**:**

$$\begin{array}{c|cc} 0 & 1 & \\ 1/2 & 1/2 & \\ \hline & 0 & 1 \end{array}$$

This gives:

[*]Rutherford Appleton Laboratory, Chilton, Didcot, UK, tyrone.rees@stfc.ac.uk

$$U_{n+1} = U_n + hf(t_n + 1/2h, U_n + 1/2hf(t_n, U_n))$$

Which is a method called **Modified Euler**.

$a_2 = 1$**:**

$$\begin{array}{c|cc} 0 & 1 & \\ 1 & 1 & \\ \hline & 1/2 & 1/2 \end{array}$$

This gives:

$$U_{n+1} = U_n + \frac{h}{2}\left(f(t_n, U_n) + f(t_n + h, U_n + hf(t_n, U_n))\right)$$

Which is, of course, the method we started with, **improved Euler**.

R=3 The same trick can be done (with messier algebra) to obtain three stage Runge-Kutta method. Again, the consistency condition is that

$$c_1 + c_2 + c_3 = 1.$$

Now, we can obtain $T_n = O(h^3)$ if we choose the parameters to satisfy

$$\begin{aligned} c_2 b_{21} + c_2(b_{31} + b_{32}) &= \frac{1}{2} \\ c_2 b_{21}^2 + c_3(b_{31} + b_{32})^2 &= \frac{1}{3} \\ c_3 b_{21} b_{32} &= \frac{1}{6} \end{aligned}$$

Including the consistency condtion, we therefore have four equations for six unknowns, leaving two parameters free.

A few examples are important enough to have a name...

The *classical* RK method is:

$$\begin{array}{c|ccc} 0 & 1 & & \\ 1/2 & 1/2 & & \\ 1 & -1 & 2 & \\ \hline & & & \\ & 1/6 & 2/3 & 1/6 \end{array}$$

(which is related to Simpson's rule). The *Nystrom* scheme is:

$$\begin{array}{c|ccc} 0 & 1 & & \\ 2/3 & 2/3 & & \\ 2/3 & 0 & 2/3 & \\ \hline & 1/4 & 3/8 & 3/8 \end{array}$$

R=4

A widely used fourth order method has Butcher table:

$$\begin{array}{c|cccc} 0 & 1 & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array},$$

which corresponds to the method

$$\begin{aligned} U_{n+1} &= U_n + \frac{1}{6}h\left\{k_1 + 2k_2 + 2k_3 + k_4\right\} \\ k_1 &= f(t_n, U_n) \\ k_2 &= f(t_n + \frac{1}{2}h, U_n + \frac{1}{2}hk_1) \\ k_3 &= f(t_n + \frac{1}{2}h, U_n + \frac{1}{2}hk_2) \\ k_4 &= f(t_n + h, U_n + hk_3). \end{aligned}$$

`compare_methods.m`

Adaptive time steps. If the solution changes very slowly, then we may be able to get a pretty good approximation with a large time step. However, if the solution changes rapidly, we won't be able to resolve the details unles we use a sufficiently small time step.

We can use our knowledge of the error to inform us of where we should next approxmimate the solution. Since the step size will change, we'll use the notation $t_{n+1} = t_n + \Delta t_n$. If the error is too large, we can reduce it by taking a smaller step.

We'll see this by way of an example. For a fourth order Runge-Kutta method, our approxiation satisfies

$$u(t_{n+1}) = U^a_{n+1} + K_1(\Delta t_n)^5 u^{(v)}(t_n) + O(\Delta t_n^6)$$

for some constant K_1.

As well as a step size of Δt_n, we could also have taken two steps of size $\Delta t_n/2$ to give us a *different* approximation at the same point. The error here will satisfy:

$$u(t_{n+1}) = U^b_{n+1} + 2K_1(\Delta t_n/2)^5 u^{(v)}(t_n) + O(\Delta t_n^6)$$

(convince yourself this is true – i.e., that the constants are the same in both cases).

Subtracting these gives

$$|U^b_{n+1} - U^a_{n+1}| \approx \frac{15}{16} K_1$$

Now, suppose we wanted this difference to take some value, ϵ. How would we pick a time step $\bar{\Delta t}_n$ to ensure this?

Suppose that

$$\epsilon = K_1 (\bar{\Delta t}_n)^5 u^{(v)}(t_n),$$

and so we can remove the unknowns by dividing these two expressions, giving:

$$\left(\frac{\bar{\Delta t}_n}{\Delta t_n} \right)^5 = \frac{\epsilon}{|U^b_{n+1} - U^a_{n+1}|}$$

Rearranging, we get that we should take

$$\bar{\Delta t}_n = \left(\frac{\epsilon}{|U^b_{n+1} - U^a_{n+1}|} \right)^{1/5} \Delta t_n.$$

This gives us a method for adapting the step length.

- if $\bar{\Delta t}_n < \Delta t_n$, (i.e. $|U^b_{n+1} - U^a_{n+1}| > \epsilon$) repeat the step from t_n with the reduced step length
- if $\bar{\Delta t}_n > \Delta t_n$, (i.e. $|U^b_{n+1} - U^a_{n+1}| \leq \epsilon$), take $U_{n+1} = U^b_{n+1}$ and set $\Delta t_{n+1} = \bar{\Delta t}_n$.

This gives a method of adapting the step length depending on the properties of the equation being solved. However, it is more expensive – at each step we do an extra application of RK4.