Math 128A, Summer 2019
Lecture 19, Wednesday 7/24/2019

1 Runge-Kutta method

We open up class answering questions about integration rules.
Last time, we proved that Euler's method converges to the exact solution:

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

in that $(u_n - y_n) \to 0$ as $n \to \infty, h \to 0$, with
The idea behind Runge-Kutta methods is to be like Euler but get a higher order of accuracy.
Euler's method says that for u_n, if this is a solution to the diff eq (which it has no hope being), then the slope throughthe point u_n would be $f(t_n, u_n)$.
Euler's method is to extend that slope in a line, and extrapolate to the point at t_{n+1}, then repeat for t_{n+1} to t_{n+2}.
Alternatively, consider taking $v := u_n + hf(u_n)$, the approximate solution at the end of some interval. Then we suddenly want to use the trapezoidal rule, $u_{n+1} := u_n + \frac{h}{2}\left(f(u_n) + f(v)\right)$. We call this the 'explicit trapezoidal rule (ETR)', where now we know v and $f(v)$.
Another way is to use Euler's to get halfway to get an approximate solution, then to set:

$$\begin{aligned} v &= u_n + \frac{h}{2}\underbrace{f(u_n)}_{k_1} \\ u_{n+1} &= u_n + h\underbrace{f(v)}_{k_2} \end{aligned}$$

These all follow the **Runge-Kutta framework**. That is, we define k_1, k_2 as:

$$\begin{aligned} k_1 &= f(u_n + h\left[a_1 k_1 + \cdots + a_s k_s\right]) \\ k_2 &= f(u_n + h\left[a_1' k_1 + \cdots a_s' k_s\right]) \end{aligned}$$

We take a linear combination of the slopes to get to the next step. Essentially, we take two steps forward to preview how it looks, then we actually take one step.
Everything depends on the A matrix, $A = [a_{ij}]$, and the vector $b = [b_i]$. Our hope is for linear algebra to save the day.
The standard way of describing Runge-Kutta methods is called the 'Butcher' array, where we put A above b^T.
For example, for Euler's method:

$$\begin{aligned} k_1 &= f(u_n) \\ u_{n+1} &= u_n + hk_1. \end{aligned}$$

Here, we have $A = 0$ (the zero matrix), and b is just 1.
Last time, we also talked about **backwards Euler**, which is an implicit method, where we look at the solution at the end of an interval. Because

$f(u_{n+1})$ is not in our desired form of h times something, we deduce that $f(u_{n+1})$ MUST be k_1.

$$u_{n+1} = u_n + h\underbrace{f(u_{n+1})}_{k_1},$$

so we write k_1 in terms of itself:

$$k_1 = f(u_n + hk_1)$$

As an aside, we can try to solve this via bisection or fixed-point iteration. We take k_1 and shrink it down dramatrically by h, so there is high probability that this will be a contractive iteration. Then, we calculate:

$$u_{n+1} = u_n + hk_1.$$

We can read off the Butcher array, where our 1×1 matrix $A = [1]$, and our array is $b = [1]$.
Trapezoidal Rule:

$$u_{n+1} = u_n + \frac{h}{2}\left(\underbrace{f(u_n)}_{k_1} + \underbrace{f(u_{n+1})}_{k_2}\right)$$

Hence we have:

$$\begin{aligned} k_1 &= f(u_n) \\ k_2 &= f\left(u_n + \frac{h}{2}[k_1 + k_2]\right) \end{aligned}$$

Then, our Butcher array is simply: $[0, 0; 1/2, 1/2; 1/2, 1/2]$.
For **ETR**, we have:

$$\begin{aligned} k_1 &= f(u_n) \\ k_2 &= f(u_n + hk_1), \end{aligned}$$

which nets us a Butcher array of $[0, 0; 1, 0; 1/2, 1/2]$.
For **Midpoint**, we have:

$$\begin{aligned} u_{n+1} &= u_n + hf\left(\frac{u_n + u_{n+1}}{2}\right) \\ k_1 &= f\left(\frac{u_n + u_{n+1}}{2}\right) \\ u_{n+1} &= u_n + hk_1, \end{aligned}$$

where

$$\begin{aligned} \frac{u_n + u_{n+1}}{2} &= \frac{u_n + u_n + hk_1}{2} \\ &= u_n + \frac{h}{2}k_1 \end{aligned}$$

which gives

$$k_1 = f\left(u_n + \frac{h}{2}k_1\right),$$

which is k_1 expressed in terms of k_1 as desired. For this 1-stage Runge-Kutta method, we write the Butcher array $[1/2;1]$.
Break time.
So we have a general Runge-Kutta method with :

$$k_i = f\left(u_n h \sum_{j=1}^{s} a_{ij} k_j\right), \qquad 1 \le i \le s$$
$$u_{n+1} = u_n + h \sum_{i=1}^{s} b_i k_i.$$

How can we make this method accurate?
Recall before we looked at the local truncation error,

$$\begin{aligned}\tau &= \frac{y_{n+1}-y_n}{h} - f(y_n) \qquad \text{Euler}\\ &= \frac{y_{n+1}-y_n}{h} - \frac{1}{2}\left(f(y_n)+f(y_{n+1})\right)\end{aligned}$$

Hence for Runge-Kutta methods, everything depends on the local truncation errors.

$$\tau = \frac{y_{n+1}-y_n}{h} - \sum_{i=1}^{s} b_i k_i =: O(h) \qquad \text{when?},$$

when

$$k_i = f\left(y_n + h \sum_{j=1}^{s} a_{ij} k_j\right)$$

Because τ is a function of h, we can Taylor expand it :

$$\tau = \tau(h=0) + \tau'(0)h + \frac{\tau''(0)}{2!}h^2 + \cdots + \frac{\tau^{(p)}(0)}{p!}h^p + \cdots,$$

where $\tau(0) = \tau'(0) = \tau^{(p-1)}(0) = 0$.
What is $\tau(0)$?

$$\begin{aligned}\tau(h) &= \frac{y_n + hy_n' + \frac{1}{2}h^2 y''(\xi_n) - y_n}{h} - \sum_{i=1}^{s} b_i k_i(h)\\ &= y_n' + \frac{1}{2}hy''(\xi_n) - \sum_{i=1}^{s} b_i k_i(h),\end{aligned}$$

as $h \to 0$. Hence we need to know what k_i approaches as $h \to 0$.

$$k_i = f(y_n + h\sum_{j=1}^{s} a_{ij} k_j) \to f(y_n),$$

where $f(y_n) \equiv f(y_n')$. Hence we write, as $h \to 0$,

$$\tau(h) \to f(y_n) - \left(\sum_{i=1}^{s} b_i\right) f(y_n) = \left(1 - \sum_{i=1}^{s} b_i\right) f(y_n),$$

and we conclude that to get first order of accuracy, we require:

$$\sum_{i=1}^{s} b_i := 1.$$

Now for **second order** of accuracy, we find:

$$\tau(h) = y_n' + \frac{1}{2}hy_n'' + \frac{1}{3!}h^2 y_n''' + \cdots - \left(\sum_{i=1}^{s} b_i\right)\left[k_i(0) + k_i'(0)h + \frac{1}{2!}k_i''(0)h^2 + \cdots\right].$$

Hence we conclude our 2nd order condition is:

$$\frac{1}{2}y_n'' - \sum_{i=1}^{s} b_i k_i'(0) = 0$$

and our 3rd order condition is:

$$\frac{1}{3!}y_n''' - \sum_{i=1}^{s} b_i k_i''(0) = 0.$$

So we write:

$$\begin{aligned} y' &= f(y) \\ y'' &= f'(y)y' = f'(y)f(y) \\ y''' &= f''(y)(y')^2 + f'(y)y'' \\ &= f''(y)f(y)^2 + f'(y)f'(y) + f(y) \\ y''' &= f''f^2 + (f')^2 f. \end{aligned}$$

This is one source of the combinatorial explosion that makes it hard, say, to find the 10th order Runge Kutta method.
Key conclusions from above are:

$$f'(y)f(y) = y''$$

and

$$y''' = f''(y)[f(y)]^2 + [f'(y)]^2 f(y).$$

We have two terms, and when we differentiate, we will have two separate conditions.
Now we calculate k_i:

$$\begin{aligned} k_i &= f\left(y_n + h\sum_j a_{ij}k_j\right) \\ k_i'(h) &= f'\left(\underbrace{y_n + h\sum_j a_{ij}k_j}\right)\left(\sum_j a_{ij}k_j + \sum_j a_{ij}k_j'\right), \end{aligned}$$

where by chain rule, we have to remember that we have k_j is a function on h, and hence we have the right factor. As $h \to 0$, we have:

$$\begin{aligned} k_i' &= f'(y_n)\sum_j [a_{ij}k_j(0)] \\ &= \left(\sum_j a_{ij}\right) f'(y_n)f(y_n) \end{aligned}$$

Notice $\sum_j a_{ij}k_j$ is bounded as $k_i(0) = f(y_n)$ as $h \to 0$, which lets us use the above convergences.
So to satisfy the 2nd order condition,

$$\frac{1}{2}f'(y_n)f(y_n) - \sum_{i=1}^{s} b_i \sum_{j=1}^{s} a_{ij}f'(y_n)f(y_n) = 0,$$

so we have:

$$\begin{aligned} \frac{1}{2} &= \sum_{i=1}^{s} b_i \left(\sum_{j=1}^{s} a_{ij}\right) \\ \implies b^T Ae &= \frac{1}{2}, \end{aligned}$$

where $e :=[1;1;1;\cdots 1]$ and

$$(Ae)_i := \sum_{j=1}^{s} a_{ij}1$$

Differentiating further and going straight $h \to 0$, we have:

$$\begin{aligned} k_i'(h) &= f'\left(\underbrace{y_n + h\sum_j a_{ij}k_j}\right)\left(\sum_j a_{ij}k_j + \sum_j a_{ij}k_j'\right) \\ k_i''(0) &= f''(y_n)\left(\sum_j a_{ij}f(y_n)\right)^2 + f'(y_n)\left[\sum_j a_{ij}f'(y_n)\sum_l a_{jl}f(y_n)\right] \end{aligned}$$

So we have two conditions:

$$\begin{aligned} \frac{1}{3!} &= \sum_i b_i \left(\sum_j a_{ij}\right)^2 \\ \frac{1}{3!} &= 2\sum_i b_i \sum_j a_{ij} \sum_l a_{jl} \end{aligned}$$

The first equation is with matrices, but we don't necessarily use linear algebra; however, if we wish, we can set the second equation equal to $b^T A^2 e$, as defined before.

Lecture ends here.

Tomorrow we'll build Runge-Kutta methods with p^2 stages in order p.