Popular Explicit One-Step Methods
MATH 447/847 - Numerical Analysis
Dr. Adam Larios

Goal: We want to approximate the solution to the equation:

$$\begin{cases} y' = f(t,y) \\ y(0) = y_0 \end{cases}$$

We look at two methods for doing this. The first has several names, including "Modified Euler," "Runge-Kutta-2 (RK-2)," "Heun's Method" and "Ralston's Method." To understand where it comes from, consider the Euler methods:

$$y_{n+1} = y_n + h \cdot f(t_n, y_n) \qquad \text{(Forward Euler)}$$

$$y_{n+1} = y_n + h \cdot f(t_{n+1}, y_{n+1}) \qquad \text{(Backward Euler)}$$

Backward Euler has better stability properties, but y_{n+1} is only *implicitly* defined, which means we have to solve an algebraic problem every time step to find y_{n+1} (unless f is very nice, e.g., it is linear, so we can solve it by hand). We would like to use Backward Euler, but the y_{n+1} on the right-hand side is not known. Instead, we can approximate it using forward Euler. We then average the results of the two methods. It looks something like this:

$$\begin{cases} y^*_{n+1} = y_n + h \cdot f(t_n, y_n) & \text{(prediction with forward Euler)} \\ y^{**}_{n+1} = y_n + h \cdot f(t_{n+1}, y^*_{n+1}) & \text{(use prediction in backward Euler)} \\ y_{n+1} = \frac{1}{2}(y^*_{n+1} + y^{**}_{n+1}) & \text{(average the predictions)} \end{cases}$$

The final y_{n+1} is what we use as our approximated value. This looks a little messy with all the $*$'s and so on thought. We can make it a little cleaner be first noting that

$$\begin{aligned} \tfrac{1}{2}(y^*_{n+1} + y^{**}_{n+1}) &= \tfrac{1}{2}[(y_n + h \cdot f(t_n, y_n)) + (y_n + h \cdot f(t_{n+1}, y^*_{n+1}))] \\ &= y_n + \tfrac{h}{2}(f(t_n, y_n) + f(t_n + h, y_n + h \cdot f(t_n, y_n))). \end{aligned}$$

since $t_{n+1} = t_n + h$. Next, note that we are being inefficient, since we compute $f(t_n, y_n)$ multiple times. Therefore, we can just save it as a value, say, k_1, and use it when we need it. We can note write the method like this:

$$\text{(RK-2)} \qquad \begin{cases} k_1 = f(t_n, y_n) \\ k_2 = f(t_n + h, y_n + h \cdot k_1) \\ y_{n+1} = y_n + \frac{h}{2}(k_1 + k_2) \end{cases}$$

This is the Modified Euler Method (or Heun, or RK-2, or Ralphson, etc.). It is an **explicit** method of **order 2**, meaning its error behaves like $M \cdot h^2$ when h is small, where M is some fixed number depending on the ODE problem, but not depending on h. For short-hand, we say it is an $\mathcal{O}(h^2)$ method, using the "Big-O" notation.

One can use a similar approach to get **higher-order methods**. Also, instead of just approximating at t and $t+h$, one can introduce approximations at other points, such as the midpoint $t+\frac{h}{2}$. By far the most popular higher-order method is the Runge-Kutta-4 method, or "RK-4". It is a 4^{th}-order method which is so popular, it is often called just, "The Runge-Kutta Method", while all other similar methods are called RK-2, RK-3, RK-5, and so on. It is very messy to derive, but the ideas are similar to those used for RK-2, so we will just give the method here:

$$\text{(RK-4)} \quad \begin{cases} k_1 = f(t_n, y_n) \\ k_2 = f(t_n + \frac{h}{2}, y_n + \frac{h}{2} \cdot k_1) \\ k_3 = f(t_n + \frac{h}{2}, y_n + \frac{h}{2} \cdot k_2) \\ k_4 = f(t_n + h, y_n + h \cdot k_3) \\ y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{cases}$$

Note that this is sometimes written in the following equivalent form:

$$\text{(RK-4)} \quad \begin{cases} k_1 = h \cdot f(t_n, y_n) \\ k_2 = h \cdot f(t_n + \frac{h}{2}, y_n + \frac{1}{2}k_1) \\ k_3 = h \cdot f(t_n + \frac{h}{2}, y_n + \frac{1}{2}k_2) \\ k_4 = h \cdot f(t_n + h, y_n + k_3) \\ y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{cases}$$

A common mistake is to use the k_i's from one form, but to use the averaging from the other form. **This is a good mistake to avoid!**

In general, one can have Runge-Kutta methods of any order. An order p method can be give as

$$\text{(RK-p)} \quad \begin{cases} k_1 = f(t_n, y_n) \\ k_2 = f(t_n + \alpha_2 h, y_n + \beta_{21} h \cdot k_1) \\ k_3 = f(t_n + \alpha_3 h, y_n + \beta_{31} h \cdot k_1 + \beta_{32} h \cdot k_2) \\ \quad \vdots \qquad \vdots \\ k_p = f(t_n + \alpha_p h, y_n + \beta_{p1} h \cdot k_1 + \beta_{p2} h \cdot k_2 + \cdots + \beta_{p,p-1} h \cdot k_{p-1}) \\ y_{n+1} = y_n + h(c_1 k_1 + c_2 k_2 + \cdots + c_p k_p) \end{cases}$$

For any given method the constants α_i, β_i, and c_i are usually looked up in a table (they are determined by working out the local truncation error with Taylor series, and choosing the constants to make all the terms cancel up to a desired order). They are typically given in the form of a "**Butcher-tableau**", named after the New Zealand mathematician John Butcher, who works at the University of Auckland. For the method to be *consistent* (i.e., for the local truncation error $\tau \to 0$ as $h \to 0$), it is sufficient for $\sum_{j=1}^{p} \beta_{i,j} = \alpha_i$ for each $i = 2, 3, \ldots, p$.

For example, (RK-p) is given by the Butcher tableau:

$$\begin{array}{c|ccccc} 0 & & & & & \\ \alpha_2 & \beta_{21} & & & & \\ \alpha_3 & \beta_{31} & \beta_{32} & & & \\ \vdots & \vdots & & \ddots & & \\ \alpha_p & \beta_{p1} & \beta_{p2} & \cdots & \beta_{p,p-1} & \\ \hline & c_1 & c_2 & \cdots & c_{p-1} & c_p \end{array}$$

Forward Euler (RK-1) is given by the Butcher tableau:

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

Modified Euler (RK-2) is given by the Butcher tableau:

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

And RK-4 is given by the Butcher tableau:

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

Matlab's ODE solver `ode45.m` is based on Erwin Fehlberg's method, which is two methods combined into one, allowing for an adaptive step-size. They have the same coefficients α_i, β_i, and only differ in the c_i coefficients, so we can write them in the same table as:

0						
1/4	1/4					
3/8	3/32	9/32				
12/13	1932/2197	-7200/2197	7296/2197			
1	439/216	-8	3680/513	-845/4104		
1/2	-8/27	2	-3544/2565	1859/4104	-11/40	
	16/135	0	6656/12825	28561/56430	-9/50	2/55
	25/216	0	1408/2565	2197/4104	-1/5	0

The first bottom row is used to compute a 4th-order accurate solution. The second bottom row is used to compute a 5th-order accurate solution. If the two methods are significantly different, the step size h is decreased, and the calculation is repeated for that step.