# AM205: additional notes on ODE solution methods

This document contains two derivations that were presented in lecture 13, about numerically solving the ordinary differential equations (ODEs) of the form

$$\frac{dy}{dt} = f(t, y). \tag{1}$$

For simplicity, $y(t)$ is interpreted as a scalar function $y : \mathbb{R} \to \mathbb{R}$, but the arguments easily generalize to coupled ODE systems for a vector function, $y : \mathbb{R} \to \mathbb{R}^m$.

## Order conditions for the two-step Runge–Kutta method

Consider taking a single integration step of size $h$ from $t_k$ to $t_{k+1}$, where $y_k = y(t_k)$. A general class of two-step Runge–Kutta methods is based on first computing two intermediate steps[1]

$$k_1 = f(t_k, y_k), \qquad k_2 = f(t_k + \alpha h, y_k + \beta h k_1) \tag{2}$$

after which the value of $y$ at $t_{k+1}$ is given by

$$y_{k+1} = y_k + h(ak_1 + bk_2), \tag{3}$$

where $a$, $b$, $\alpha$, and $\beta$ are arbitrary constants. Consider what conditions must be placed on these constants in order to achieve second-order accuracy. To begin, the Taylor series expansion of $y(t_{k+1})$ about $y(t_k)$ is

$$\begin{aligned}
y(t_{k+1}) &= y(t_k) + hy'(t_k) + \frac{h^2}{2}y''(t_k) + O(h^3) \\
&= y(t_k) + hf(t_k, y_k) + \frac{h^2}{2}\left.\frac{d}{dt}f(t, y(t))\right|_{t=t_k} + O(h^3) \\
&= y(t_k) + hf(t_k, y_k) + \frac{h^2}{2}\left(f_t(t_k, y_k) + y' f_y(t_k, y_k)\right) + O(h^3) \\
&= y(t_k) + hf(t_k, y_k) + \frac{h^2}{2}\left(f_t(t_k, y_k) + f(t_k, y_k) f_y(t_k, y_k)\right) + O(h^3)
\end{aligned} \tag{4}$$

where $f_t = \partial f / \partial t$ and $f_y = \partial f / \partial y$. The Taylor series expansion of $k_2$ is

$$k_2 = f(t_k, y_k) + \alpha h f_t(t_k, y_k) + \beta h f(t_k, y_k) f_y(t_k, y_k) + O(h^2). \tag{5}$$

Substituting this expansion into Eq. 3 gives

$$y_{k+1} = y_k + h(a+b)f(t_k, y_k) + \alpha b h^2 f_t(t_k, y_k) + \beta b h^2 f(t_k, y_k) f_y(t_k, y_k) + O(h^3). \tag{6}$$

[1] Be aware that there is some unfortunate notation here: $k$ is used both to represent an intermediate step, and as the subscript on the timestep. Since these are both standard notations for the course, we make use of them regardless.

This will match the Taylor series in Eq. 4 to second order in $h$ provided that

$$a + b = 1, \qquad \alpha b = \beta b = \frac{1}{2}. \tag{7}$$

This gives three constraints for the four unknowns, meaning that there is still some freedom to choose the constants. As discussed in the lecture, several different methods satisfying these constraints are typically used, such as the improved Euler method, modified Euler method, or Ralston's method. Usually, the additional freedom is used to simplify the constants to be simple numbers; in particular, setting a number to be zero can result in better performance, since terms can be skipped during the calculation of the $k_i$. Furthermore, the additional freedom can be used to minimize the $O(h^3)$ error term.

Satisfying the order conditions becomes increasingly complex to achieve higher-order accuracy. As can be seen in the above derivation, all of the differential partial derivatives of $f$ must match, and this will become more complicated when higher-order, mixed partial derivatives must be accounted for. Indeed, it can be shown that for $p > 5$, there is no explicit Runge–Kutta method of order $p$ with exactly $p$ intermediate stages; it becomes necessary to use more stages than the order required.

## Derivation of Adams–Bashforth multi-step methods

The Adams–Bashforth methods are part of a family of multi-step integration methods that use information from previous timesteps as a way to increase the order of accuracy. Suppose that the ODE in Eq. 1 has been integrated over many timesteps $t_k$ each with step size $h$, and consider taking a step to $t_{k+1}$. Then the integral

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f(t, y) dt \tag{8}$$

is an exact relationship for $y(t_{k+1})$ in terms of $y(t_k)$. The explicit Adams–Bashforth methods are based on approximating $f(t, y)$ over this interval by using a polynomial interpolation function $p(t)$ of $f$ over the some number of previous time points $t_{k-m+1}, t_{k-m+2}, \ldots, t_k$, since at each one of these points $f_l = f(t_l, y(t_l))$ can be computed. Substitution of $p(t)$ into Eq. 8 gives an explicit formula for $y(t_{k+1})$ in terms of $y(t_k)$ and $f_{k-m+1}, f_{k-m+2}, \ldots, f_k$.

To illustrate this consider the case of $m = 2$, so that the polynomial interpolant is a straight line,

$$p(t) = f_k + s(f_k - f_{k-1}), \tag{9}$$

where the variable $s$ is defined as $t = t_k + hs$ for notational convenience. Substituting Eq. 9

into Eq. 8 gives

$$\begin{aligned} y(t_{k+1}) &= y(t_k) + \int_{t_k}^{t_{k+1}} p(t)dt \\ &= y(t_k) + h\int_0^1 (f_k + s(f_k - f_{k-1}))ds \\ &= y(t_k) + h\left[f_k s + \frac{s^2(f_k - f_{k-1})}{2}\right]_0^1 \\ &= y(t_k) + h\left(\frac{3}{2}f_k - \frac{1}{2}f_{k-1}\right), \end{aligned} \tag{10}$$

which is a second-order accurate formula for $y_(t_{k+1})$.

A second family of implicit Adams–Bashforth methods can be derived by considering a different interpolating polynomial $p^*(t)$ which also goes through the point $t_{k+1}$, $f_{k+1}$. In this case, the resulting numerical scheme will also depend on $f_{k+1}$, and in general it must be solved using nonlinear root finding.