1. Review of Numerical Methods

We have talked about methods based on Taylor series expansion (e.g. the forward Euler method). The forward Euler method is easily implemented directly from Taylor series terms. But to obtain higher order methods, we would need symbolic derivatives. We want to avoid that. So, we turned to methods like the explicit trapezoidal method, which involve repeated evaluations of the right-hand-side of the ODE $f$. These methods are called Runge-Kutta methods and we gained intuition about how they work by thinking of them in terms of numerical quadrature (e.g. a trapezoid is a better estimate of the area under the curve than a rectangle is).

Runge-Kutta methods are called 1-step methods because all of the function evaluations are done within one interval. I.e. if the method is computing $y(t_n)$, then all evaluations of $f$ are done within the interval between $t_n$ and $t_{n-1}$. And before we move onto today’s topic, it is important to notice that Runge-Kutta methods are not necessarily linear in $f$. E.g. for the trapezoidal method we have

$$y_n = y_{n-1} + \frac{h}{2} f(t_{n-1}, y_{n-1}) + \frac{h}{2} f(t_n, y_{n-1} + hf(t_{n-1}, y_{n-1}))$$

Today, we will move to linear multi-step methods, which

1. Use information from time steps earlier than $t_{n-1}$
2. Are linear in $f$.

The family of linear multi-step methods which we will consider are the Adams family. In particular, we will explore explicit Adams methods called Adams-Bashforth methods.

To understand the Adams family, we should return to our picture of numerical quadrature and think about our method written like this:

$$y_n = y_{n-1} + \int_{t_{n-1}}^{t_n} f(t, y(t))dt$$

Runge-Kutta methods aimed to find the best approximation of the area under $f$. Adams methods, instead of approximating the area under $f$, find a polynomial function that
approximates $f$, and then compute the exact integral under the polynomial. So an Adams-Bashforth method works like this:

$$y_n = y_{n-1} + \int_{t_{n-1}}^{t_n} \phi(t)dt$$

where $\phi(t)$ is a polynomial estimated with evaluations of $f$ at previous time steps.

So let’s take a moment to talk about polynomial interpolation. If you have a function $f(t)$, you can estimate it with a polynomial. The number of time points at which you know the value of $f$ determines the order of the polynomial you can use. If you know $f$ at one point, then you can use a zeroth order polynomial (i.e. it is a constant function at that value). If you know $f$ at two points, then you can use a first order polynomial (i.e. 2 points define a line). If you know $f$ at three points, then you can use a second order polynomial. And so on and so forth. Higher order polynomials may lead to better approximations of the line altogether.

In our context, higher order polynomials do lead to better approximations. Which brings us back to the original goal - finding $\phi(t)$ that will allow us to solve our ODE. To find $\phi(t)$, we use values of $f$ evaluated at earlier time steps. For the first order method, we use just $f(t_{n-1}, y_{n-1})$, but for the second order method we also use $f(t_{n-2}, y_{n-2})$. For a $k^{th}$ order method we use the steps from $f(t_{n-k}, y_{n-k})$ to $f(t_{n-1}, y_{n-1})$. These time steps must be evenly spaced.

**First order method.** A first order Adams-Bashforth method employs a 0th order polynomial:

$$\phi(t) = f(t_{n-1}, y_{n-1})$$

The method is then

$$y_n = y_{n-1} + \int_{t_{n-1}}^{t_n} \phi(t)dt$$

$$= y_{n-1} + \int_{t_{n-1}}^{t_n} f(t_{n-1}, y_{n-1})dt$$

$$= y_{n-1} + f(t_{n-1}, y_{n-1})t_n|_{t_{n-1}}$$

$$= y_{n-1} + f(t_{n-1}, y_{n-1})t_n - f(t_{n-1}, y_{n-1})t_{n-1}$$

$$= y_{n-1} + f(t_{n-1}, y_{n-1})(t_n - t_{n-1})$$

$$= y_{n-1} + f(t_{n-1}, y_{n-1})h$$

which is our old friend forward Euler.
Second order method. A second order Adams-Bashforth method employs a first order polynomial:

\[ \phi(t) = -\frac{f(t_{n-1}, y_{n-2})}{h} (t - t_{n-1}) + \frac{f(t_{n-1}, y_{n-1})}{h} (t - t_{n-2}) \]

The method is

\[
y_n = y_{n-1} + \int_{t_{n-1}}^{t_n} \phi(t) dt
\]

\[
= y_{n-1} + \int_{t_{n-1}}^{t_n} \left( -\frac{f(t_{n-1}, y_{n-2})}{h} (t - t_{n-1}) + \frac{f(t_{n-1}, y_{n-1})}{h} (t - t_{n-2}) \right) dt
\]

\[
= y_{n-1} + \frac{3h}{2} f(t_{n-1}, y_{n-1}) - \frac{h}{2} f(t_{n-2}, y_{n-2})
\]

which is similar to the Runge-Kutta trapezoidal method in that it uses a trapezoid to estimate the area under the curve. It is different in that it uses a different method to approximate the later point of the trapezoid.

To obtain higher order methods, we can use higher order polynomials. The Lagrange formula for the interpolating polynomial helps us find the coefficients (the scalars that we use to multiply f). Wikipedia and other sources do this, and compute the integrals, and supply the methods for us.

Coding Issues. There are two issues to consider when writing code for Adams Bashforth methods.

1. The theory depends upon having the same step size for all \( k \) steps. So adapting the step size is tricky for these methods. We don’t be adapting step size in this course.
2. We need the first \( k \) points instead of just the first 1 point. The solution is to use another method (e.g. a Runge-Kutta method) to find those values, then transition to the Adams-Bashforth method. It is important to use a Runge-Kutta method of order \( k \), so that we don’t loose precision in the method overall (the method is only as good as its weakest step!).