Numerical solution of ordinary differential equations

Differential equations—equations that relate one or more functions and their derivatives—frequently arise as models of physical processes (because derivatives measure rates of change). However, most problems originating from the study of real-world phenomena cannot be solved exactly, with the solution expressed in terms of elementary functions. Therefore, obtaining approximate solutions to differential equations is a very important area of scientific computing.

If a differential equation involves only one independent variable, it is an *ordinary* differential equation (ODE); if there are two or more independent variables, it is a *partial* differential equation (PDE).
We will begin by considering the problem of approximating the solution $u(t)$ to an ODE

$$\frac{du}{dt} = f(t,u), \quad \text{given that } u(t_0) = u_0$$

where t_0 and u_0 are fixed values. The techniques developed will also be useful for obtaining numerical solutions to PDEs.

Time-stepping

We will call the independent variable t and think of it as "time". The basic idea is to choose a sequence of t-values

$$t_0,\ t_1,\ t_2,\ \ldots,\ t_N$$

and "step" from one to the next, computing the approximate value of u at each successive t_n.
We will use u_n to denote the approximate value of $u(t_n)$, for $n = 1, 2, \ldots, N$.
Supposing that we have already computed $u_n \approx u(t_n)$, the aim is to derive formulas/algorithms for obtaining $u_{n+1} \approx u(t_{n+1})$.

The formulas to be discussed will be interpreted in one or both of the following ways:

- The ODE $\dfrac{du}{dt} = f\big(t, u(t)\big)$ says that the *slope* of the solution curve $u(t)$ at any point t is given by $f(t,u)$; values of f are *slopes* in the (t,u) plane.
- By the fundamental theorem of calculus, $\displaystyle\int_{t_n}^{t_{n+1}} u'(t)\,dt = u(t_{n+1}) - u(t_n)$, so integrating both sides of the ODE $u'(t) = f\big(t, u(t)\big)$ from $t = t_n$ to $t = t_{n+1}$ gives

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

  and hence

  $$u_{n+1} \approx u_n + \int_{t_n}^{t_{n+1}} f\big(t, u(t)\big)\,dt$$

  The *integral* on the right-hand side is the area bounded by the $f\big(t, u(t)\big)$ curve and the t-axis between $t = t_n$ and $t = t_{n+1}$.

Systems

Although we shall derive the numerical methods by thinking of scalar first-order ODEs, i.e. ODEs having just one dependent variable and containing only first-order derivatives, the algorithms will also be applicable to ODE *systems* with more than one dependent variable. In particular, higher-order ODEs can always be rewritten as systems and then solved numerically using the same algorithms.

EXAMPLES

The simple harmonic oscillator (spring–mass system with no damping) is described by the second-order ODE $mx'' + kx = 0$ where $'$ denotes differentiation with respect to time t.

The two-body problem describes the orbit of one body under the gravitational attraction of another, much heavier, body. The scaled equations of motion are

$$x'' + \frac{x}{r^3} = 0, \qquad y'' + \frac{y}{r^3} = 0, \qquad \text{where } r = \sqrt{x^2 + y^2}$$

Euler's method

If we approximate the integral $\displaystyle\int_{t_n}^{t_{n+1}} f\big(t, u(t)\big)\, dt$ by the area of the *left-hand rectangle*, we get Euler's method. The height of the left-hand rectangle is $f\big(t_n, u(t_n)\big)$, which we approximate by $f\big(t_n, u_n\big)$, and its width is $t_{n+1} - t_n$, usually denoted by h. Therefore the formula for Euler's method is

$$u_{n+1} = u_n + h\, f(t_n, u_n)$$

This is the simplest example of an *explicit* method, where the "next" approximation u_{n+1} is given explicitly in terms of values that are already known, namely t_n and the previously computed u_n.

From the "slope" viewpoint, Euler's method can be interpreted as approximating the solution curve $u(t)$ between $t = t_n$ and $t = t_{n+1}$ by the tangent line at t_n, i.e. we advance from t_n to t_{n+1} by following the slope at t_n, which is given by $f\big(t_n, u(t_n)\big) \approx f(t_n, u_n)$.

Backward Euler method

If we approximate the integral $\displaystyle\int_{t_n}^{t_{n+1}} f\big(t, u(t)\big)\, dt$ by the area of the *right-hand rectangle*, we get the backward Euler method. The height of the right-hand rectangle is $f\big(t_{n+1}, u(t_{n+1})\big)$, which we approximate by $f\big(t_n + h, u_{n+1}\big)$, and its width is h. Therefore the formula for the backward Euler method is

$$u_{n+1} = u_n + h\, f(t_n + h,\ u_{n+1})$$

This is the simplest example of an *implicit* method, because the quantity that we want to find, namely u_{n+1}, appears on both sides of the formula. To obtain the value of u_{n+1}, we need to *solve* the above equation. Because of this extra equation-solving step, implicit methods generally require more computational effort and are much slower.

Crank–Nicolson method

If we approximate the integral $\displaystyle\int_{t_n}^{t_{n+1}} f\big(t, u(t)\big)\, dt$ by the *trapezoidal* rule, and then substitute the approximations $u(t_n) \approx u_n$ and $u(t_{n+1}) \approx u_{n+1}$, we get the Crank–Nicolson formula

$$u_{n+1} = u_n + \frac{h}{2}\Big[f(t_n, u_n) + f(t_n + h,\ u_{n+1})\Big]$$

This is also an implicit method that requires us to solve the equation for u_{n+1}.