18.330 Lecture Notes: Integration of Ordinary Differential Equations

Homer Reid

February 13, 2014

Contents

1 Overview

1.1 ODEs and ODE Integrators

A first-order ordinary differential equation (ODE) is an equation of the form

$$\frac{du}{dt} = f(t, u) \tag{1}$$

for some function $f(t, u)$. (We will always call the independent variable t and the dependent variables u.) More specifically, 1 is an example of an *initial value problem*; one also encounters ODEs posed in the form of *boundary value problems*, to be discussed below.

Given an ODE like (1) with a reasonably well-behaved RHS function f, and given a single point (t_0, u_0), an extensive and well-established theory assures us that there is a unique solution curve $u(t)$ passing through this point (i.e. satisfying $u(t_0) = u_0$). However, the theory doesn't tell us how to write down an analytical expression for this solution curve, and in general no analytical expression exists. Instead, we must resort to numerical methods to compute points that lie approximately on the curve $u(t)$.

An *ODE integrator* is an algorithm that takes an ODE like (1) and a point (t_0, u_0) and produces a new point (t_1, u_1) that (at least approximately) lies on the unique solution curve passing through (t_0, u_0). More generally, we will usually want to compute a whole sequence of points $(t_1, u_1), (t_2, u_2), \cdots, (t_{\max}, u_{\max})$. up to some maximum time $t_{\max}$.

ODE systems

In general we may have two or more ODEs that we need to solve at the same time. For example, denote the populations of the U.S. and Uruguay respectively by $u_1(t)$ and $u_2(t)$. Suppose the U.S. and Uruguayan birth rates are respectively γ_1 and γ_2, and suppose that every year a fraction γ_a of the U.S. population emigrates to Uruguay, while a fraction γ_b of the Uruguayan population emigrates to the U.S. Then the differential equations that model the population dynamics of the two countries are

$$\frac{du_1}{dt} = (\gamma_1 - \gamma_a)u_1 + \gamma_b u_2 \tag{2a}$$

$$\frac{du_2}{dt} = \gamma_a u_1 + (\gamma_2 - \gamma_b)u_2 \tag{2b}$$

These two ODEs are *coupled*; each one depends on the other, so we can't solve separately but must instead solve simultaneously. We generally write systems like (2) in the form

$$\frac{d\mathbf{u}}{dt} = \mathbf{f}(t, \mathbf{u}) \tag{3}$$

where $\mathbf{u}$ is now a time-dependent d-dimensional *vector* and $\mathbf{f}$ is a d-dimensional *vector-valued function*. For the case of (2), the dimension $d = 2$ and the $\mathbf{u}$ vector

is $\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$.

As in the case of the one-dimensional system (1), for reasonably well-behaved functions $\mathbf{f}$ we are guaranteed to have a unique solution curve $\mathbf{u}(t)$ passing through any given point $(t_0, \mathbf{u}_0)$, but again in general we cannot write down an analytical expression for such a curve; we must use numerical methods to generate a sequence of points $(t_0, \mathbf{u}_0), (t_1, \mathbf{u}_1), \cdots (t_{\max}, \mathbf{u}_{\max})$.

Actually, in the case of (2), the function $\mathbf{f}(t, \mathbf{u})$ is *linear*, $\mathbf{f} = \mathbf{A}\mathbf{u}$ for a constant matrix $\mathbf{A}$. In this special case, (3) can be solved analytically in terms of the matrix exponential to yield, for initial conditions $\mathbf{u}(t = 0) = \mathbf{u}_0$,

$$\mathbf{u}(t) = e^{\mathbf{A}t}\mathbf{u}_0.$$

However, this only works for *linear* ODE systems; for nonlinear systems in general no analytical solution is possible.

Higher-order ODEs

Equations (1) and (3) are *first-order* ODEs, i.e. they only involve first derivatives. What if we have a higher-order ODE? Consider, for example, the one-dimensional motion of a particle subject to a position-dependent force:

$$\frac{d^2u_1}{dt^2} = \frac{1}{m}F(u_1). \tag{4}$$

It turns out that we can squeeze high-order ODEs like this into the framework of the above discussion simply by giving clever names to some of our variables. More specifically, let's assign the name u_2 to the first derivative of u_1 in (4).

$$\frac{du_1}{dt} \equiv u_2.$$

Now we can reinterpret the second-order equation (4) as a *first-order* equation for u_2:

$$\frac{du_2}{dt} = \frac{1}{m}F(u_1).$$

Equations (5) constitute a 2-dimensional system of first-order ODEs:

$$\frac{d\mathbf{u}}{dt} = \mathbf{f}(t, \mathbf{u}), \qquad \mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \qquad \mathbf{f} = \begin{pmatrix} u_2 \\ F(u_1)/m. \end{pmatrix}$$

Thus we can use all the same tricks we use to solve first-order ODE systems; there is no need to develop special methods for higher-order ODEs. This is a remarkable example of the efficacy of using the right notation.

We can play this trick to convert any system of ODEs, of any degree, to a one-dimensional system. In general, a d-dimensional system of p-th order ODEs can always be rewritten as a pdth-dimensional system of first-order ODEs.

Here's another example of this reduction process: The third-order nonlinear ODE

$$\dddot{x} + A\ddot{x} - (\dot{x})^2 + x = 0$$

is equivalent, upon defining $\{u_1, u_2, u_3\} \equiv \{x, \dot{x}, \ddot{x}\}$, to the following system:

$$\frac{d}{dt}\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} u_2 \\ u_3 \\ -u_1 + u_2^2 - Au_3 \end{pmatrix}.$$

This system has been called the "simplest dissipative chaotic flow."[1]

1.2 Comparison to numerical quadrature

Consider the problem of evaluating the integral

$$\int_a^b f(t)\,dt \tag{6}$$

which we studied earlier in our unit on numerical quadrature. This problem may be recast in the language of ODEs as follows: Define $u(t)$ to be the function

$$u(t) \equiv \int_a^t f(t')\,dt'.$$

Then $u(t)$ satifies the first-order initial-value ODE problem

$$\frac{du}{dt} = f(t), \qquad u(a) = 0 \tag{7}$$

and the integral (6) we want to compute is the value of the solution curve $u(t)$ at $t = b$.

Comparing (7) to (1), we notice an important distinction: The RHS function $f(t)$ in (7) depends only on t, not on u. This means that integrating functions is *easier* than integrating ODEs. In particular, suppose we use a numerical quadrature rule of the form

$$\int_a^b f(t)\,dt \approx \sum w_i f(t_i)$$

to estimate our integral. Because the integrand only depends on t, we could (for example) evaluate all the function samples $f(t_i)$ in parallel, as they are independent of each other. No such approach is possible for general ODEs like (1) or (3); we must instead proceed incrementally, computing one point on our solution curve at a time and then using this point as the springboard to compute the next point.

[1] Reference: J. C. Sprott, *Physics Letters A* **228** 271 (1997).

2 Examples of ODE systems

2.1 Motion of particles in force fields

Historically, techniques for numerically integrating ODEs were first developed to study the motion of celestial bodies (planets, comets, etc.) in the solar system. Consider a planet of mass m at a distance r from the sun, which has mass $M_\odot$. The planet experiences a gravitational force directed radially inward with magnitude $F = GM_\odot m/r^2$ (where G is Newton's gravitational constant). Then Newton's second law reads

$$m\ddot{\mathbf{r}} = -\frac{GM_\odot m}{r^2}\hat{\mathbf{r}} \tag{8}$$

This is a three-dimensional second-order system of ODEs, which we may express as a six-dimensional first-order system as follows. First, define the u variables to be

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \qquad \begin{pmatrix} u_4 \\ u_5 \\ u_6 \end{pmatrix} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} \tag{9}$$

where x, y, z are the components of $\mathbf{r}$. Then equation (8) is equivalent to the following system:

$$\frac{d}{dt}\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{pmatrix} = \begin{pmatrix} u_4 \\ u_5 \\ u_6 \\ -\gamma u_1/(u_1^2+u_2^2+u_3^2)^{3/2} \\ -\gamma u_2/(u_1^2+u_2^2+u_3^2)^{3/2} \\ -\gamma u_3/(u_1^2+u_2^2+u_3^2)^{3/2} \end{pmatrix} \tag{10}$$

where $\gamma = GM_\odot$.[2]

Of course, in the real solar system we have more than simply one planet, and planets experience gravitational attractions *to each other* in addition to their attraction to the sun. You will work out some implications of this fact in your problem set.

[2] To derive e.g. the 4th component of this equation, we write the x component of equation (8) as follows:

$$\ddot{x} = \frac{d}{dt}\dot{x} = \frac{GM_\odot}{x^2+y^2+z^2}\hat{r}_x \tag{11}$$

where $\hat{r}_x$ is the x component of the radially-directed unit vector $\hat{\mathbf{r}}$, which we may write in the form

$$\hat{r}_x = \frac{x}{(x^2+y^2+z^2)^{1/2}}.$$

Plugging this into (11) and renaming the variables according to (9) yields the fourth component of equation (10).

2.2 Molecular dynamics

2.3 Electric circuits

2.4 Chemical reactions

Ozone (O_3) in the Earth's atmosphere disintegrates through a process of reacting with oxygen monomers and dimers. The primary reactions are

$$O_3 + O_2 \underset{k_2}{\overset{k_1}{\rightleftharpoons}} O + 2O_2, \qquad O_3 + O \xrightarrow{k_3} 2O_2$$

Label the concentrations of O, O_2, O_3 respectively by $u_1(t), u_2(t), u_3(t)$. Then the above reaction system implies the following nonlinear ODE system:

$$\frac{d}{dt}\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} k_1 u_3 u_2 - k_2 u_1 u_2^2 - k_3 u_3 u_1 \\ -k_1 u_3 u_2 + k_2 u_1 u_2^2 + k_3 u_3 u_1 \\ -k_1 u_3 u_2 + k_2 u_1 u_2^2 - k_3 u_3 u_1. \end{pmatrix}.$$

This system is not even *close* to being linear.

2.5 Meteorology and chaos

2.6 Charge renormalization in quantum electrodynamics

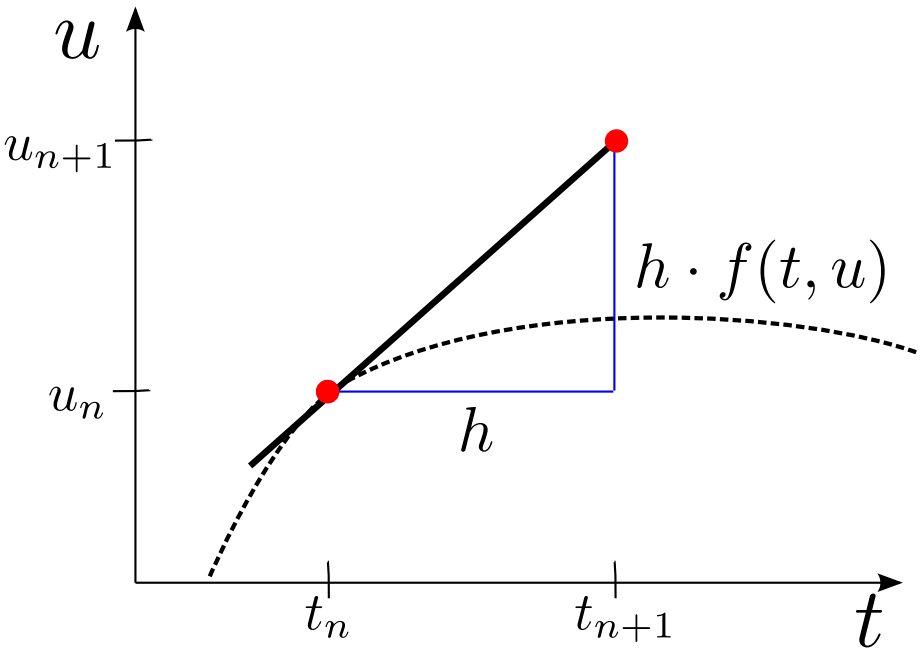

Figure 1: Euler's method illustrated for the 1D case. We are given an ODE $du/dt = f(t, \mathbf{u})$ and a single point (t_n, u_n). The dashed line denotes the unique solution curve through this point; we know it exists, but we don't have an analytical expression for it. What we *do* know is its slope at the given point [this slope is just $s = f(t_n, u_n)$], so we move along this line until we have traveled a horizontal distance $\Delta t = h$ on the t axis.

3 ODE Integration Algorithms

3.1 Forward (Explicit) Euler

The simplest possible ODE integration algorithm is Euler's method (sometimes known as the *forward Euler* or *explicit Euler* method to contrast it with an alternative version we will discuss below). The idea behind this method is pictured in Figure 3.1. Given a point $(t_n, \mathbf{u}_n)$ on a solution curve, the RHS of equation (3) tells us the slope of the tangent line to the solution curve at that point. Euler's method is simply to move along this line until we have traveled a horizontal distance $\Delta t \equiv h$. (h is known as the *step size*). In equations, the Euler step transitions from one point to the next according to the rule

$$(t_n, \mathbf{u}_n) \quad \rightarrow \quad (t_{n+1}, \mathbf{u}_{n+1})$$

$$t_{n+1} = t_n + h, \tag{12a}$$
$$\mathbf{u}_{n+1} = \mathbf{u}_n + h\mathbf{f}(t_n, \mathbf{u}_n). \tag{12b}$$

For the special case of a linear ODE system

$$\dot{\mathbf{u}} = \mathbf{A}\mathbf{u},$$

equation (12b) takes the form

$$\mathbf{u}_{n+1} = \mathbf{u}_n + h\mathbf{A} \cdot \mathbf{u}_{n+1}$$

or

$$\mathbf{u}_{n+1} = \big(\mathbf{I} + h\mathbf{A}\big)\mathbf{u}_n \tag{13}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix. So each step of the forward Euler method requires us to do a single matrix-vector multiplication. If $\mathbf{A}$ is a *sparse* matrix, this can be done in $O(n)$ operations. (We haven't discussed sparse matrices or operation counts yet, so this observation is made for future reference.)

Error analysis

How accurate is Euler's method? Consider the simplest case of a one-dimensional ODE system (the extension to a general n-dimensional system is immediate). Given a point (t_0, u_0), we know there is a unique solution curve $u(t)$ passing through this point. The Taylor expansion of this function around the point t_0 takes the form

$$u(t) = \underbrace{u(t_0)}_{u_0} + (t - t_0)\underbrace{u'(t_0)}_{f(t_0,u_0)} + \frac{1}{2}(t - t_0)^2 u''(t_0) + \cdots \tag{14}$$

Note that, in this expansion, $u(t_0)$ is just u_0, and $u'(t_0)$ is just $f(t_0, u_0)$, i.e. the value of the RHS function in our ODE at the initial point.

If we use (14) to compute the actual value of u at the point $t_0 + h$, we find

$$u(t_0 + h) = u_0 + hf(t_0, u_0) + \frac{1}{2}h^2 u''(t_0) + \cdots \tag{15}$$

On the other hand, the Euler-method approximation to $u(t_0 + h)$ is precisely just the first two terms in this expansion:

$$u^{\text{Euler}}(t_0 + h) = u_0 + hf(t_0, u_0). \tag{16}$$

Thus the error between the Euler-method approximation and the actual value is

$$u(t_0 + h) - u^{\text{Euler}}(t_0 + h) = \frac{1}{2}h^2 u''(t_0, u_0) + \cdots$$

This result depends on $u''(t_0, u_0)$, which we don't know. However, what's important is that it tells us the error is proportional to h^2. If we try again with one-half the step size h, everything on the RHS stays the same except the factor h^2, which now decreases by a factor of 4. To summarize,

$$\text{error in each step of the Euler method} \propto h^2. \tag{17}$$

On the other hand, in general we will not be taking just a single step of the Euler method, but will instead want to use it to integrate over some interval $[t_a, t_b]$. If we break up this interval into $N = \frac{t_b - t_a}{h}$ steps of width h, then (18) tells us that the error *in each step* is proportional to h^2, but there are N steps, so the total error is proportional to $Nh^2 \propto h$. In other words,

$$\text{overall error in the Euler method} \propto h. \tag{18}$$

The Euler method is a method of order 1.

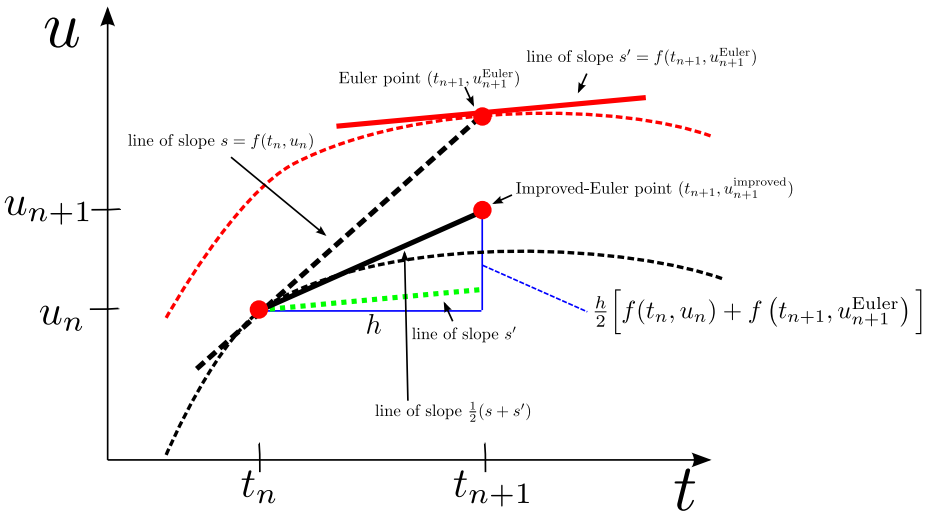

Figure 2: The improved Euler method. **(a)** Starting at a point (t_n, u_n), we take the usual forward Euler step by moving a horizontal distance h along a line of slope s (dashed black line in the figure), where $s = f(t_n, u_n)$ is the value of f at the starting point. This takes us to the Euler point, $(t_{n+1}, u_{n+1}^{\text{Euler}})$. **(b)** When we get to the Euler point, we sample the value of f there. Call this value $s' = f(t_{n+1}, u_{n+1}^{\text{Euler}})$. s' is the slope of a tangent line (solid red line) to the ODE solution curve through the Euler point (dashed red curve). **(c)** Now we go back to the starting point and draw a line of slope $\frac{1}{2}(s + s')$ (solid black line). (The slope of this line is intermediate between the slope of the dashed black and dashed green lines in the figure.) Moving a horizontal distance h along this line takes us to the improved Euler point.

3.2 Improved Euler

Another possibility is the *improved Euler* method, pictured for the 1D case in Figure 3.1. Like the Euler method, it computes a successor point to $(t_n, \mathbf{u}_n)$ by moving a horizontal distance h along a straight line. The difference is that, whereas in the original Euler method this line has slope s, in the improved Euler method the line has slope $\frac{1}{2}(s + s')$. Here s and s' are the values of the ODE function $f(t, \mathbf{u})$ at the starting point $(t_n, \mathbf{u}_n)$ and at the Euler point $(t_{n+1}, \mathbf{u}_{n+1}^{\text{Euler}})$. (The "Euler point" is just the point to which the usual Euler method takes us.)

The idea is that by averaging the slopes of the solution curves at the starting point and at the Euler point, we get a better approximation to what is happening in between those points. Thus, if we draw a line whose slope is the average of the two slopes, we expect that moving along this line should be better than just moving along the line whose slope is s, as we do in the original Euler method.

In equations, the improved Euler method takes the step

$$(t_n, \mathbf{u}_n) \quad \to \quad (t_{n+1}, \mathbf{u}_{n+1})$$

where

$$t_{n+1} = t_n + h \tag{19a}$$

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \frac{h}{2}\left[\mathbf{f}(t_n, \mathbf{u}_n) + \mathbf{f}\big(t_{n+1}, \mathbf{u}_{n+1}^{\text{Euler}}\big)\right] \tag{19b}$$

where $\mathbf{u}_{n+1}^{\text{Euler}} = \mathbf{u}_n + h\mathbf{f}(t_n, u_n)$.

Error analysis

To analyze the error in the improved Euler method, consider again the 1D case: we are at a point (t_0, u_0), which we know lies on a solution curve $u(t)$, and we want to get to the point $u(t_0 + h)$. We will compare an exact expansion for this quantity with the approximate version computed by the improved Euler method, and this will allow us to estimate the error in the latter.

Exact expansion for $u(t_0 + h)$ As above, we can write down an expression for the exact value of $u(t_0 + h)$ by Taylor-expanding $u(t)$ about t_0:

$$u^{\text{exact}}(t_0 + h) = u(t_0) + hu'(t_0) + \frac{h^2}{2}u''(t_0) + \frac{h^3}{6}u'''(t_0) + \cdots \tag{20}$$

In our error analysis of the Taylor method above, we observed that the first two terms in this expansion were simply

$$\begin{aligned} u(t_0) &= u_0 \\ u'(t_0) &= f(t_0, u_0) \end{aligned}$$

To get at u'', we now go like this:

$$\begin{aligned} u''(t_0) &= \frac{d}{dt}u'(t_0) \\ &= \frac{d}{dt}f(t_0, u_0) \end{aligned}$$

We now evaluate this *total* derivative by making use of the *partial* derivatives of f:

$$\begin{aligned} &= \left|\frac{\partial f}{\partial t}\right|_{t_0,u_0} + \left|\frac{\partial f}{\partial u}\right|_{t_0,u_0}\frac{du}{dt} \\ &= f_t(t_0, u_0) + f_u(t_0, u_0)f(t_0, u_0). \end{aligned}$$

where we are using the shorthand

$$\frac{\partial f}{\partial t} \equiv f_t, \qquad \frac{\partial f}{\partial u} \equiv f_u$$

Inserting into (20), we have

$$u^{\text{exact}}(t_0+h) = u_0+hf(t_0,u_0)+\frac{h^2}{2}\Big[f_t(t_0,u_0)+f_u(t_0,u_0)f(t_0,u_0)\Big]+O(h^3) \quad (21)$$

Improved Euler approximation to $u(t_0+h)$ On the other hand, consider the approximation to $u(t_0+h)$ that we get from the improved Euler method

$$u^{\text{improved}}(t_0+h) = u_0 + \frac{h}{2}\left[f\Big(t_0,u_0\Big) + f\Big(t_0+h, u_0+hf(t_0,u_0)\Big)\right] \quad (22)$$

Let's expand the second term in square brackets here[3]:

$$f\Big(t_0+h, u_0+hf(t_0,u_0)\Big) = f\big(t_0,u_0\big)+f_t\big(t_0,u_0\big)\cdot h+f_u\big(t_0,u_0\big)\cdot hf(t_0,u_0)+\cdots \quad (23)$$

Inserting (23) into (22) and collecting terms, we have

$$\begin{aligned} u^{\text{improved}}(t_0+h) &= u_0+\frac{h}{2}\left[2f\big(t_0,u_0\big)f_t\big(t_0,u_0\big)\cdot h + f_u\big(t_0,u_0\big)\cdot hf\big(t_0,u_0\big)+\cdots\right] \\ &= u_0+hf\big(t_0,u_0\big)+\frac{h^2}{2}\Big[f_t\big(t_0,u_0\big)+f_u\big(t_0,u_0\big)f\big(t_0,u_0\big)\Big]+O(h^3) \end{aligned}$$

Comparison Comparing this against expression (21), we see that the improved Euler method has succeeded in replicating the first *three* terms in the Taylor-series expansion of $u(t_0+h)$ (whereas the usual Euler method only replicates the first *two* terms), so the error decays one order more rapidly than in the ordinary Euler method:

$$u^{\text{exact}}(t_0+h) - u^{\text{improved}}(t_0+h) = O(h^3)$$

Overall error Of course, as before, this only gives us the error *per step*, i.e.

$$\text{error in each step of the improved Euler method} \quad \propto \quad h^3$$

so if we use improved Euler to integrate an ODE from a to b using steps of width h, then the number of steps grows linearly as we shrink h, so as before the global error decays one order less rapidly than the local error:

$$\text{overall error in the improved Euler method} \quad \propto \quad h^2.$$

The improved Euler method is a method of order 2.

[3] All we are doing in this step is expanding $f(t,u)$ in a two-variable Taylor series in t and u around the points t_0 and u_0 and keeping only the linear terms in the expansion:

$$f(t_0+\Delta t, u_0+\Delta u) = f(t_0,u_0)+f_t(t_0,u_0)\Delta t + f_u(t_0,u_0)\Delta u+\cdots$$

3.3 Runge-Kutta Methods

Although the error analysis for improved Euler is a little tricky, the idea of the method is straightforward: Instead of simply sampling $f(t, u)$ at the left endpoint of the interval we are traversing, we sample it at both the left and right endpoints and take the average between the two. This gives us a better representation of the behavior of the function over the interval than just sampling at one endpoint.

It's also pretty clear how we might improve further on the method: Just sample $f(t, u)$ at even more points, now including points inside the interval, and do some kind of averaging to get a better sense of the behavior of f throughout the interval.

This is the motivation for *Runge-Kutta* methods.[4] There are a family of these methods, indexed by the number of function samples they take on each step and the order of convergence they achieve. For example, the simplest Runge-Kutta method is known as the *midpoint method* and is defined by the following algorithm: Given an ODE $\frac{du}{dt} = f(t, u)$ and a point (t_n, u_n), we compute the successor point as follows:

$$\begin{aligned} s_1 &= f(t_n, u_n) \\ s_2 &= f\left(t_n + \frac{h}{2}, u_n + \frac{h}{2}s_1\right) \\ (t_{n+1}, u_{n+1}) &= \left(t_n + h, u_n + hs_2\right) \end{aligned}$$

What this algorithm does is the following: It first takes an Euler step with stepsize $h/2$ and samples the function f at the resulting point, yielding the value s_2. This is an estimate of the slope of ODE solution curves near the midpoint of the interval we are traversing. Then we simply proceed from the starting point to the successor point by moving a horizontal distance h along a line of slope s_2. Thus the midpoint method is almost identical to the original Euler method, in the sense that it travels to the successor point by moving a distance h along a straight line; the only difference is that we use a more refined technique to esimate the slope of that straight line.

The most popular Runge-Kutta method is the fourth-order method, known colloquially as "RK4." This method again travels a horizontal distance h along a straight line, but now the slope of the line is obtained as a weighted average of *four* function samples throughout the interval of interest. More specifically,

[4] Note: "Runge" rhymes with "cowabunga."

RK4 is the following refinement of the midpoint method:

$$\begin{aligned} s_1 &= f(t_n, u_n) \\ s_2 &= f\left(t_n + \frac{h}{2}, u_n + \frac{h}{2}s_1\right) \\ s_3 &= f\left(t_n + \frac{h}{2}, u_n + \frac{h}{2}s_2\right) \\ s_4 &= f(t_n + h, u_n + hs_3) \\ (t_{n+1}, u_{n+1}) &= \left(t_n + h, u_n + \frac{h}{6}\Big(s_1 + 2s_2 + 2s_3 + s_4\Big)\right) \end{aligned}$$

Here s_1 is the ODE slope at the left end of the interval, s_2 and s_3 are samples of the ODE slope midway through the interval, and s_4 are samples of the ODE slope at the right end of the interval. We compute a weighted average of all these slopes, $s^{\text{avg}} = (s_1+2s_2+2s_3+s_4)/6$, and then we proceed to our successor point by moving a horizontal distance h along a line of slope s^{avg}.

As with all the methods we have discussed, it is easy to generalize RK4 to ODE systems of arbitrary dimensions. Given an ODE $\frac{d\mathbf{u}}{dt} = \mathbf{f}(t, \mathbf{u})$ and a point $(t_n, \mathbf{u}_n)$, RK4 computes the successor point as follows:

$$\begin{aligned} \mathbf{s}_1 &= \mathbf{f}(t_n, \mathbf{u}_n) \\ \mathbf{s}_2 &= \mathbf{f}\left(t_n + \frac{h}{2}, \mathbf{u}_n + \frac{h}{2}\mathbf{s}_1\right) \\ \mathbf{s}_3 &= \mathbf{f}\left(t_n + \frac{h}{2}, \mathbf{u}_n + \frac{h}{2}\mathbf{s}_2\right) \\ \mathbf{s}_4 &= \mathbf{f}(t_n + h, \mathbf{u}_n + h\mathbf{s}_3) \\ (t_{n+1}, \mathbf{u}_{n+1}) &= \left(t_n + h, \mathbf{u}_n + \frac{h}{6}\Big(\mathbf{s}_1 + 2\mathbf{s}_2 + 2\mathbf{s}_3 + \mathbf{s}_4\Big)\right) \end{aligned}$$

Error analysis in RK4

Although we won't present the full derivation, it is possible to show that RK4 is a *fourth-order method*: with a stepsize h, the error per step decreases like h^5 and the overall error decreases like h^4.

4 Stability

4.1 Stability of the forward Euler method

Consider the following initial-value problem:

$$\frac{du}{dt} = -\lambda u, \qquad u(0) = 1 \tag{24}$$

with solution

$$u(t) = e^{-\lambda t}. \tag{25}$$

Consider applying Euler's method with stepsize h to this problem. The sequence of points we get is the following:

$$\begin{aligned}
(t_0, u_0) &= (0, 1) \\
(t_1, u_1) &= (h, 1 - h\lambda) \\
(t_2, u_2) &= \Big(2h, 1 - h\lambda - h\lambda(1 - h\lambda)\Big) \\
&= \Big(2h, (1 - h\lambda)^2\Big) \\
(t_3, u_3) &= \Big(3h, (1 - h\lambda)^2 - h\lambda(1 - h\lambda)^2\Big) \\
&= \Big(3h, (1 - h\lambda)^3\Big)
\end{aligned}$$

and in general

$$(t_N, u_N) = \Big(Nh, (1 - h\lambda)^N\Big).$$

In other words, the Euler-method estimate of the value of $u(t)$ after N timesteps is

$$u^{\text{Euler}}(Nh) = (1 - h\lambda)^N$$

More generally, if we had started with initial condition $u(0) = u_0$, then after N timesteps we would have

$$u^{\text{Euler}}(Nh) = (1 - h\lambda)^N u_0. \tag{26}$$

Notice something troubling here: If $h > 2/\lambda$, then the quantity $(1 - h\lambda)$ is *negative with magnitude greater than 1*, which means that $u^{\text{Euler}}(Nh)$ grows in magnitude and flips sign each time we compute a new step. This cannot come close to capturing the correct behavior of the exact function $u(t)$, which is always positive and decays monotonically to zero. Figure 4.3 shows the result of applying Euler's method, with $h = 0.42$, to the problem (24) with $\lambda = 5$.

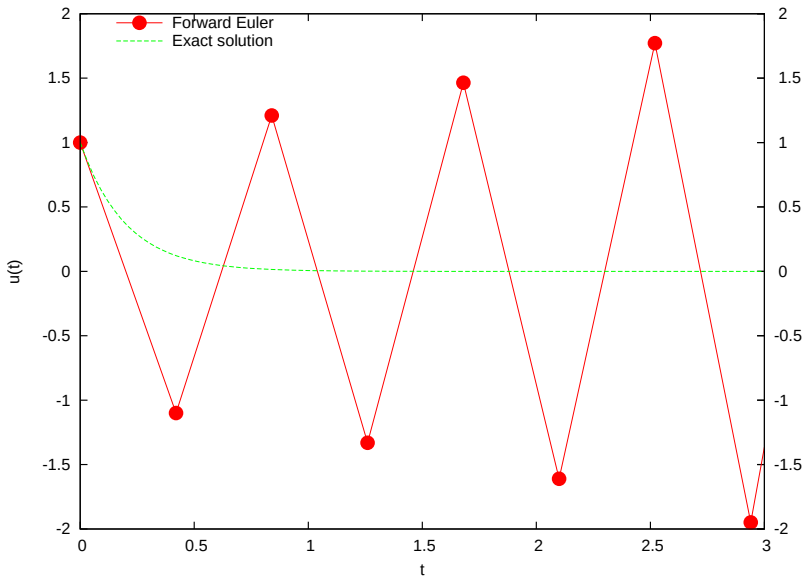

Figure 3: Instability of the forward Euler method with stepsize $h = 0.42$ applied to the ODE $\frac{du}{dt} = -5u$.

We diagnose this problem by saying that Euler's method applied to (24) with stepsize h is *unstable* if $h > \frac{2}{\lambda}$. More broadly, we say that Euler's method for this problem is *conditionally stable*: it is stable for some values of h, and unstable for others.

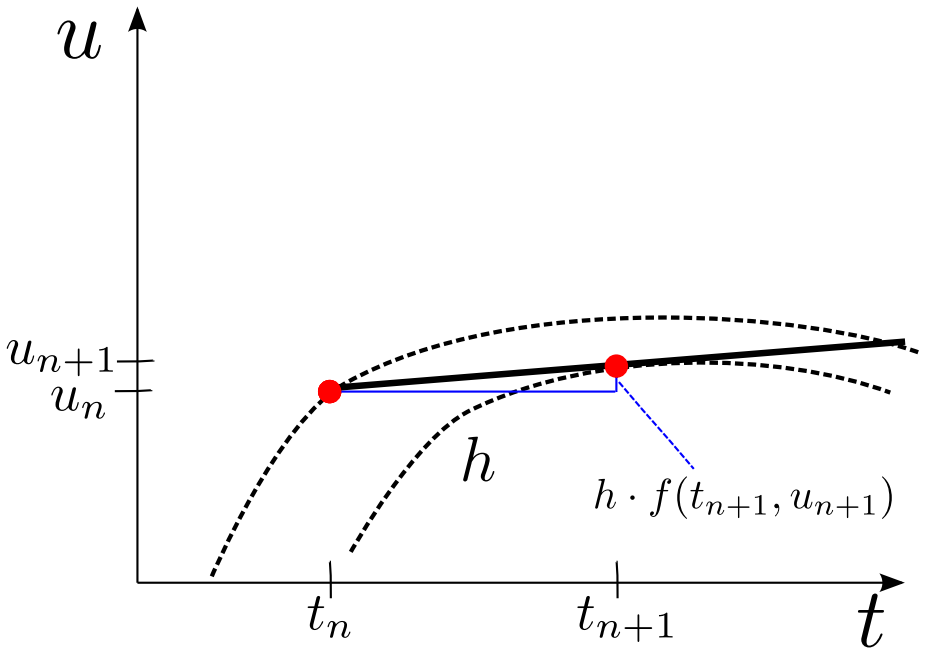

Figure 4: The implicit Euler method (also known as the backward Euler method). As in the forward Euler method, we proceed from (t_n, u_n) to (t_{n+1}, u_{n+1}) by moving on a straight line until we have traveled a horizontal distance h along the t axis. The difference is that now the slope of the line is the slope of the ODE solution curve through the *new* point (t_{n+1}, u_{n+1}). Because we don't know this point *a priori*, we must solve an implicit equation to find it – hence the name of the technique.

4.2 The backward (implicit) Euler method

Instability in ODE integration schemes may be remedied by using *backward* or *implicit* methods, of which the simplest is the implicit version of the usual Euler method, known as the *backward Euler* or *implicit Euler* method and illustrated in Figure 4.2. As in the case of the forward Euler method, we proceed from the old point $(t_n, \mathbf{u}_n)$ to the new point $(t_{n+1}, \mathbf{u}_{n+1})$ by moving along a straight line until we have traveled a horizontal distance h along the t axis. The difference is that now the slope of this line is chosen to be the slope of the ODE solution curve through the *new* point $(t_{n+1}, \mathbf{u}_{n+1})$. But since we don't know where this point is *a priori*, we have to solve implicitly for it. In equations, the implicit Euler method for proceeding from one point to the next is

$$(t_n, \mathbf{u}_n) \quad \rightarrow \quad (t_{n+1}, \mathbf{u}_{n+1})$$

$$t_{n+1} = t_n + h \tag{27a}$$
$$\mathbf{u}_{n+1} = \mathbf{u}_n + h\mathbf{f}(t_{n+1}, \mathbf{u}_{n+1}). \tag{27b}$$

For the typical case of a nonlinear function $\mathbf{f}$, solving the implicit equation (27b) is significantly more costly than simply implementing the explicit equation (12b).

For the special case of a linear ODE system

$$\dot{\mathbf{u}} = \mathbf{A}\mathbf{u},$$

equation (27b) takes the form

$$\mathbf{u}_{n+1} = \mathbf{u}_n + h\mathbf{A} \cdot \mathbf{u}_{n+1}$$

which we may solve to obtain

$$\mathbf{u}_{n+1} = \big(\mathbf{I} - h\mathbf{A}\big)^{-1}\mathbf{u}_n. \tag{28}$$

Thus each iteration of the implicit Euler algorithm requires us to *invert* an $n \times n$ matrix (or, essentially equivalently, to solve an $n \times n$ linear system). This is *much* more costly than simply evaluating a matrix-vector product, which is all that we need for the explicit Euler method [equation (13)].

Error analysis of the implicit Euler method

It is easy to mimic the analysis we performed of the usual (explicit) Euler method to show that the implicit Euler method is a first-order method, i.e. the overall error decays like h^p with $p = 1$. This is the same convergence rate as the explicit Euler method, so all the extra cost of the implicit Euler method doesn't buy us anything on this front.

4.3 Stability of the backward Euler method

What the implicit method *does* buy us is unconditional stability. Consider applying the backward Euler method with stepsize h to the problem (25). At each timestep, the equation we have to solve, equation (27), reads

$$u_{n+1} = u_n - h\lambda u_{n+1}$$

which we can solve to find

$$u_{n+1} = \frac{1}{(1 + h\lambda)} u_n.$$

Starting from an initial point $(t, u) = (0, u_0)$, the value of u after N timesteps is now

$$u^{\text{backward Euler}}(Nh) = \frac{1}{(1 + h\lambda)^N} u_0. \tag{29}$$

Comparing this result to (26), we see the advantage of the implicit technique: assuming $\lambda > 0$, there is *no value of* h for which (29) grows with N. We say that the implicit Euler method is *unconditionally stable.*

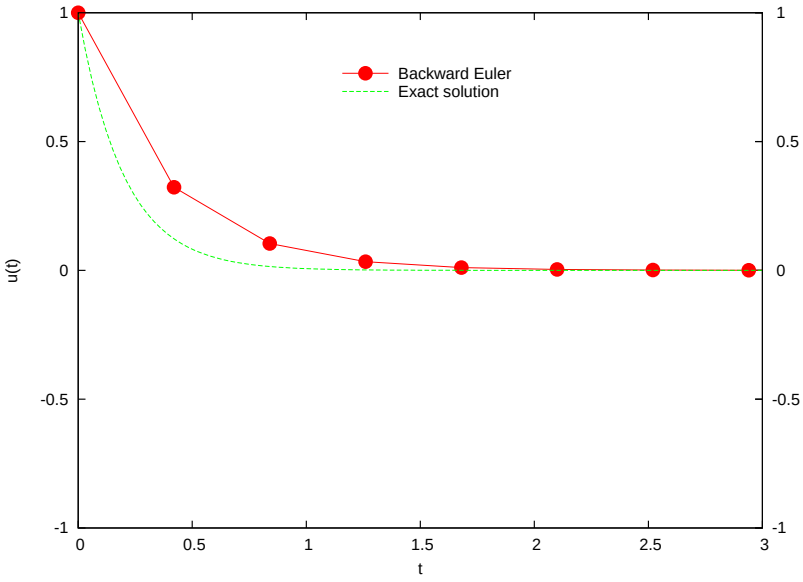

Figure 5: Stability of the backward Euler method with stepsize $h = 0.42$ applied to the ODE $\frac{du}{dt} = -5u$.

4.4 Stability in the multidimensional case

We carried out the analysis above for a one-dimensional linear ODE, but it is easy to extend the conclusions to a higher-dimensional linear ODE. Recall from 18.03 that the N-dimensional linear ODE system

$$\frac{d\mathbf{u}}{dt} = \mathbf{A} \cdot \mathbf{u}, \qquad \mathbf{u}(0) = \mathbf{u}_0$$

(where A is an $N \times N$ matrix with constant coefficients) has the solution

$$\mathbf{u}(t) = C_1 e^{\lambda_1 t}\mathbf{v}_1 + C_2 e^{\lambda_2 t}\mathbf{v}_2 + \cdots + C_N e^{\lambda_N t}\mathbf{v}_N \tag{30}$$

where $(\lambda_i, \mathbf{v}_i)$ are the eigenpairs of $\mathbf{A}$, and where the C_i coefficients are determined by expanding the initial-condition vector in the basis of eigenvectors:

$$\mathbf{u}_0 = C_1\mathbf{v}_1 + C_2\mathbf{v}_2 + \cdots + C_N\mathbf{v}_N.$$

If we were to use Euler's method with stepsize h to integrate this ODE, we would find that the condition for stability would be

$$h < \frac{2}{\lambda_{\max}} \tag{31}$$

where $\lambda_{\max}$ is the eigenvalue lying farthest to the left in the complex plane, i.e. the eigenvalue with the largest negative real part (corresponding to the most rapidly decaying term in 30). On the other hand, the timescale over which we will generally want to investigate the system is determined by the *least* rapidly decaying term in 30, i.e.

$$t_{\max} \approx \frac{2}{\lambda_{\min}} \tag{32}$$

where $\lambda_{\min}$ is the eigenvalue with the smallest negative real part.[5]

Comparing (31) and (32), we see that the number of ODE timesteps we would need to take to integrate our system stably using Euler's method is roughly

$$\frac{t_{\text{Max}}}{h} \sim \frac{\lambda_{\max}}{\lambda_{\min}}.$$

If the dynamic range spanned by the eigenvalues is large, we will need many small timesteps to integrate our ODE. Systems with this property are said to be *stiff*, and stiff ODEs are a good candidate for investigation using implicit integration algorithms.

4.5 Stability in the nonlinear case

So far we have discussed stability for linear ODEs. For nonlinear ODEs, we typically consider the system in the vicinity of a fixed point and investigate the stability with respect to small perturbations. More specifically, for an autonomous nonlinear ODE system of the form

$$\frac{d\mathbf{u}}{dt} = \mathbf{f}(\mathbf{u}) \tag{33}$$

(where we suppose the RHS function is independent of t – that's what *autonomous* means) we suppose $\mathbf{u}_0$ is a *fixed point*–that is, a zero of the RHS, i.e.

$$\mathbf{f}(\mathbf{u}_0) = 0. \tag{34}$$

The unique solution of (33) passing through the point $t_0, \mathbf{u}_0$ is $\mathbf{u}(t) \equiv \mathbf{u}_0$, constant independent of time; you can check using (34) that this is indeed a solution of (33). We now consider small perturbations around this solution, i.e. we put

$$\mathbf{u}(t) = \mathbf{u}_0 + \Delta\mathbf{u}(t)$$

[5] For simplicity in this discussion we are supposing that none of the eigenvalues have positive real parts. If any eigenvalues have positive real parts, then the underlying ODE *itself* is unstable (small perturbations in initial conditions lead to exponentially diverging outcomes).

and consider the Taylor-series expansion of the RHS of (33) about the fixed point:

$$\begin{aligned}\frac{d\Delta\mathbf{u}}{dt} &= \mathbf{f}(\mathbf{u}_0 + \Delta\mathbf{u}) \\ &= \mathbf{f}(\mathbf{u}_0) + \boldsymbol{\mathcal{J}} \cdot \Delta\mathbf{u} + O(\Delta\mathbf{u}^2)\end{aligned}$$

where $\boldsymbol{\mathcal{J}}$ is the Jacobian of $\mathbf{f}$ at $\mathbf{u}_0$. Neglecting terms of quadratic and higher order in $\Delta\mathbf{u}$, this is now a linear system that we can investigate using standard linear stability analysis techniques.

5 Pathological cases

Above we noted that for an ODE $\frac{du}{dt} = f(t,u)$ and an initial point (t_0, u_0), we are guaranteed the existence of a unique solution curve as long as the function $f(t,u)$ is "reasonably well-behaved." In this section we will first look at a couple of illustrations of what can go wrong if f is badly behaved; having seen what can go wrong, we will then be in a position to quantify more rigorously the conditions that must be satisfied to guarantee existence and uniqueness.

5.1 Non-uniqueness

One example of what can go wrong if f is badly behaved is furnished by the following initial-value problem:

$$\frac{du}{dt} = \sqrt{u}, \qquad u(0) = 0. \tag{35}$$

Solving this equation by separation of variables, we find

$$\frac{du}{\sqrt{u}} = dt \qquad \Longrightarrow \qquad 2\sqrt{u} = t + C$$

and applying the initial condition yields the solution

$$u(t) = \frac{t^2}{4}. \tag{36}$$

But now consider using one of the ODE integration schemes discussed above to integrate this equation. Using Euler's method with stepsize h, for example, we find

$$\begin{aligned}(t_0, u_0) &= (0, 0)\\ (t_1, u_1) &= (h, 0 + h \cdot \sqrt{0}) = (h, 0)\\ (t_2, u_2) &= (2h, 0 + h \cdot \sqrt{0}) = (2h, 0)\end{aligned}$$

and indeed after N timesteps we find

$$(t_N, u_N) = (Nh, 0 + h \cdot \sqrt{0}) = (Nh, 0).$$

This procedure appears to be tracing out the solution curve

$$u(t) = 0 \tag{37}$$

But we already figured out that the solution is given by equation (36)! What went wrong?!

What went wrong is that the RHS function $f(t,u) = \sqrt{u}$, though seemingly innocuous enough, is not sufficiently well-behaved to guarantee the uniqueness of solutions to (35). Indeed, you can readily verify that (37) is a solution to

(35) that is every bit as valid as (36). But aren't ODE solutions supposed to be unique?

The behavior of f that disqualifies it in this case is that it is *not differentiable at* $u = 0$. Nonexistence of any derivative of f violates the conditions required for existence and uniqueness and can give rise to nonunique solutions such as (36) and (37).

5.2 Blowup in finite time

A different kind of pathological behavior is exhibited by the ODE

$$\frac{du}{dt} = u^2, \qquad u(0) = 1. \tag{38}$$

Again using separation of variables to solve, we find

$$\frac{du}{u^2} = dt \qquad \Longrightarrow \qquad -\frac{1}{u} = t + C$$

and applying the initial conditions yields

$$u(t) = \frac{1}{1-t}.$$

In this case, uniqueness is not a problem, but existence is problematic at the point $t = 1$. The solution blows up in finite time, and doesn't exist for $t \geq 1$.

The behavior of $f(t, u)$ that causes trouble here is that it grows superlinearly in u. The function $f(t, u) = u$ grows linearly in u, and $f(t, u) = \sqrt{u}$ grows sublinearly in u (although this function leads to different pathologies, as noted above), but $f(t, u) = u^2$ grows superlinearly in u. This violates the conditions required for existence of ODE solution and can give rise to situations in which a solution exists only for a finite range of the t variable.

In fact, you can easily repeat the above analysis for the more general differential equation

$$\frac{du}{dt} = u^p, \qquad u(t_0) = 1 \tag{39}$$

to find the solution

$$u(t) = \begin{cases} e^t, & p = 1 \\ \dfrac{1}{\left[t - \frac{1}{p-1}\right]^{p-1}} & p > 1. \end{cases}$$

For $p = 1$ we have existence and uniqueness for all time.[6] But for any $p > 1$ the function $u(t)$ blows up (i.e. ceases to exist) at the finite time $t = \frac{1}{p-1}$.

[6] The function e^t does grow without bound, and for large values of t it assumes values that in *practice* are ridiculously large, but it never becomes infinite for finite t.

5.3 Conditions for existence and uniqueness

The above two cases illustrate the two basic ways in which the solution to an ODE $\frac{du}{dt} = f(t, u)$ can fail to exist or be unique: **(a)** either f or some derivative of f can blow up (fail to exist) at some point in our domain of interest, or **(b)** f can grow superlinearly in y.

To exclude these pathological cases, mathematicians invented a name for functions that do not exhibit either of them. Functions $f(t, u)$ that are free of both pathologies **(a)** and **(b)** are said to be *Lipschitz*, and the basic existence and uniqueness theorem states that *a solution to* $\frac{du}{dt} = f(t, u)$ *exists and is unique iff* f *is Lipschitz.*

We have intentionally avoided excessive rigor in this discussion in order to get the main points across somewhat informally; the topic of Lipschitz functions and existence and uniqueness of ODEs is discussed in detail in every ODE textbook and many numerical analysis textbooks.