www.simiode.org
SIMIODE
A SYSTEMIC INITIATIVE FOR MODELING INVESTIGATIONS
& OPPORTUNITIES WITH DIFFERENTIAL EQUATIONS

STUDENT VERSION
Stiff Differential Equations

Kurt Bryan
Department of Mathematics
Rose-Hulman Institute of Technology
Terre Haute IN U.S.

1 INTRODUCTION

Most real-world systems of ordinary differential equations (ODE's) must be solved numerically, either because no closed-form solution exists, or the solution is too unwieldy to manipulate symbolically. "Stiffness" is a particular property that some systems of ODE's have that makes them challenging to solve numerically. Stiffness often reflects an essential disparity of scale in a physical system—one facet of a physical system may evolve in time on a relatively slow scale, e.g., seconds, while another facet of the same system evolves much more rapidly, e.g., microseconds. It might be the slow scale/large time evolution of the system that interests us, but the rapidly changing aspect of the system's behavior demands the attention of the numerical method and makes long-term simulations computationally expensive. Stiff systems are surprisingly common, but rarely talked about in introductory ODE or numerical analysis courses.

We consider here a variety of straightforward examples that illustrate the essential idea behind stiffness. The goal here is not a treatise on the theory of numerical methods for ODE's and stiff systems, but rather an intuitive and practical guide to what stiff ODE's are and some ideas on how to handle them. There are also some physical examples and exercises to illustrate how stiff systems come about.

2 NUMERICAL METHODS FOR ODE'S REVIEW

Consider a first order initial value problem

$$y'(t) = f(t, y(t)) \tag{1}$$

for an unknown scalar function $y(t)$, with initial condition $y(t_0) = a$. You can consider t as time. It is frequently the case that we can't solve (1) explicitly and so must resort to numerical approximation. Recall that Euler's method is one of the simplest schemes for this computation, and is really just repeated tangent line extrapolation. In the examples that follow we'll often use Euler's method to illustrate, but the conclusions generally hold for any "explicit one-step" numerical method (defined below).

In Euler's method we choose a step size "h" with $h > 0$ for the independent variable t and then construct a sequence of approximations $y_k \approx y(t_k)$ where $t_k = t_0 + kh$ with $k = 1, 2, \ldots$, as follows. If h is sufficiently small the approximation

$$y'(t) \approx \frac{y(t+h) - y(t)}{h} \tag{2}$$

should be accurate (in the limit $h \to 0$ the right side converges to $y'(t)$, assuming y is differentiable). The approximation (2) can be rearranged to

$$y(t+h) \approx y(t) + hy'(t)$$

and if $y(t)$ satisfies (1) we then have

$$y(t+h) \approx y(t) + hf(t, y(t)). \tag{3}$$

Equation (3) tells us how to extrapolate the solution forward from time t to time $t + h$, at least approximately.

We can use (3) to approximate $y(t)$ for $t_0 \le t \le T$ for some final time $T = t_0 + Mh$ with the following algorithm, Euler's Method:

1. Set $y_0 = a$ and counter $k = 0$. Set $t_k = t_0 + kh$.
2. Use (3) to generate y_{k+1}, an approximation to $y(t_{k+1})$, as
$$y_{k+1} = y_k + hf(t_k, y_k). \tag{4}$$
3. If $t_k = T$ (equivalently, $k = M$) stop and return $y_0, y_1, \ldots, y_M$. Otherwise, increment $k \leftarrow k+1$ and return to step 2.

The smaller the step size h, the better this algorithm "tracks" the true solution.

Example 2.1. Consider the classic "exponential decay" ODE

$$y' = \lambda y \tag{5}$$

where $\lambda < 0$, with initial condition $y(0) = 1$. The true solution is $y(t) = e^{\lambda t}$.

For $\lambda = -1$ we have $y(t) = e^{-t}$. If we approximate $y(t)$ with Euler's method on the interval $0 \le t \le 5$ with step size $h = 0.5$, the situation is as shown in the left panel in Figure 1. Also shown is the direction field for the ODE (5). When h is smaller, the Euler procedure tracks the direction field or solution more accurately, as shown in the right panel of Figure 1.

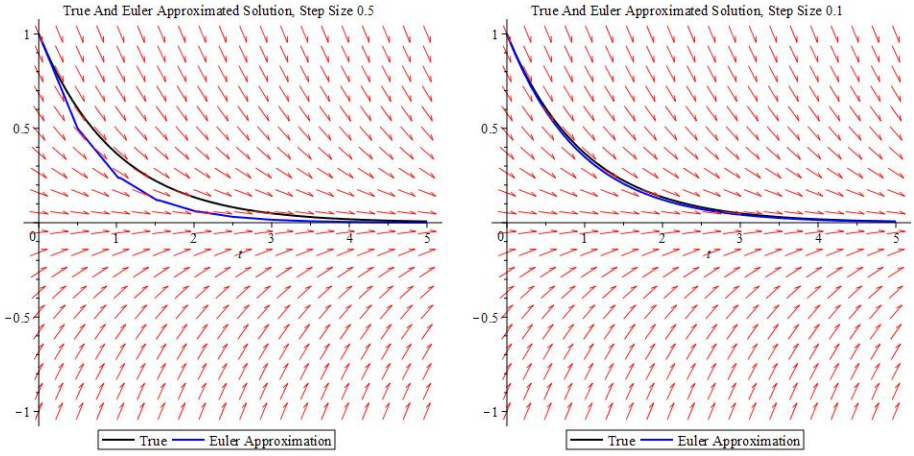

Figure 1. Euler approximation to $y' = -y$ solution, step sizes 0.5 (left) and 0.1 (right).

But now consider what happens if we solve (5) with $\lambda = -3$ and the same step size $h = 0.5$, as shown in the left panel in Figure 2. The Euler approximation is very poor, and qualitatively incorrect—the true solution is always positive, but the approximation is actually negative at times. When $\lambda = 5$ the situation is even worse, as illustrated in the right panel of Figure 2, for although the true solution decays smoothly to zero, Euler's method is unstable—the iterates grow without bound!

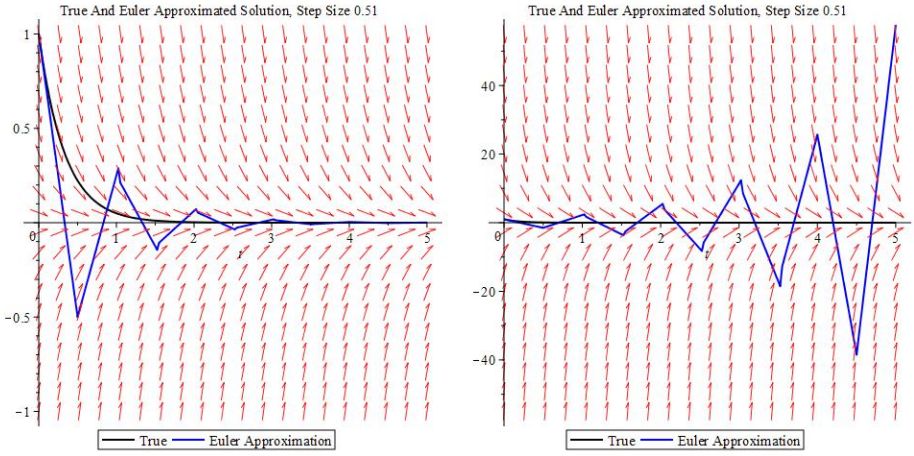

Figure 2. Euler approximation to $y' = -\lambda y$ solution, step size 0.5, $\lambda = 3$ (left) and $\lambda = 5$ (right).

If you examine Euler's method geometrically in relation to the direction field, it's easy to see what goes wrong. Although the true solution decays smoothly to zero for any $\lambda < 0$, when λ is "very" negative the solution changes rapidly, with a rather steep rate of decay. Even if the current iterate y_k is exact, Euler steps that are too large end up extrapolating to a wildly incorrect value

(very negative). For this ODE, a large step size in Euler's method yields an iteration that isn't simply inaccurate, but actually unstable.

2.1 Exercises

1. For the ODE (5) with $y(0) = 1$, show that Euler's method produces iterates that satisfy $y_{k+1} = (1 + \lambda h)y_k$, and that (given that $y_0 = 1$)

$$y_k = (1 + \lambda h)^k. \tag{6}$$

2. Based on (6), for a fixed $\lambda < 0$ show that the condition $|1 + \lambda h| < 1$ requires that $h < 2/|\lambda|$ if we want $y_k \to 0$ as $k \to \infty$ (so the iterates decay, just like the solution). What happens if this condition is not met (h is too large?) What condition on h assures that the y_k iterates actually remain positive?
3. The Improved Euler Method replaces the formula (4) in Euler's Method with the two-step procedure

$$\begin{aligned} u &= y_k + hf(t_k, y_k) \\ y_{k+1} &= y_k + \frac{h}{2}\left(f(t_k, y_k) + f(t_{k+1}, u)\right) \end{aligned} \tag{7}$$

   and is generally more accurate than Euler's Method.

   Solve the ODE (5) with $\lambda = -1$ and $y(0) = 1$ using the Improved Euler method, with step size $h = 0.5$ out to time $t = 5$. Look at the iterates $y_0, \ldots, y_{10}$. Do they remain positive? Decay to zero? Are they accurate? Repeat with $\lambda = -3$ and $\lambda = -5$ (step size $h = 0.5$). Does the Improved Euler method suffer from the same stability problems as Euler's method?

The moral here is that for many numerical methods and ODE's, step sizes that are too large result in more than just inaccurate iterates. The numerical solution actually becomes unstable and "blows up", even if the true solution decays. This is starting to get at what stiffness is about, but it's not the whole story. Stiffness is really a property possessed by a system of differential equations. Let's look at some more examples, now involving systems of ODE's.

3 STIFF SYSTEMS OF ODE'S

A system of ODE's can be written in a form similar to (1), namely

$$\mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y}(t)) \tag{8}$$

where $\mathbf{y}(t) = \langle y_1(t), \ldots, y_n(t)\rangle$ is a vector-valued function taking values in $\mathbb{R}^n$, whose components are the functions of interest. The function $\mathbf{f}(t, \mathbf{y}(t))$ is a vector-valued function taking values in $\mathbb{R}^n$ that indicates how to compute the derivative of each function $y_k(t)$ in terms of t and $\mathbf{y}(t)$. Euler's Method for such a system takes precisely the same form as the scalar version and is simply repeated extrapolation on each component of $\mathbf{y}(t)$.

Example 3.1. Let's start by considering a system of ODE's of the form

$$y_1'(t) = \lambda_1 y_1(t) \tag{9}$$

$$y_2'(t) = \lambda_2 y_2(t) \tag{10}$$

for some constants $\lambda_1, \lambda_2 < 0$. In the framework of (8) we have $\mathbf{f}(t, \mathbf{y}) = \langle \lambda_1 y_1, \lambda_2 y_2 \rangle$. Of course, this "system" is in fact decoupled—each equation can be solved independently of the other. Nonetheless, it will serve to illustrate the essential issue with stiff systems. The exact solution is $y_1(t) = y_1(0)e^{\lambda_1 t}$ and $y_2(t) = y_2(0)e^{\lambda_2 t}$.

Let's use Euler's Method to solve the system starting at time $t_0 = 0$ with $y_1(0) = 0.5$ and $y_2(0) = 0.5$. Let h be the step size, $t_k = kh$, and let's use y_1^k and y_2^k to denote the iterates produced by Euler's Method, as approximations to $y_1(t_k)$ and $y_2(t_k)$. Euler's Method (8) for the system (9)-(10) takes the form

$$y_1^{k+1} = (1 + \lambda_1 h) y_1^k \tag{11}$$

$$y_2^{k+1} = (1 + \lambda_2 h) y_2^k. \tag{12}$$

It's easy to see then that $y_1^k = y_1(0)(1 + \lambda_1 h)^k$ and $y_1^k = y_2(0)(1 + \lambda_2 h)^k$ from Exercise 1 above.

Suppose that λ_1 and λ_2 are both negative, and recall Exercise 2 above. Both true solutions decay to zero, but $|1 + \lambda_j h| > 1$ if $h > 2/|\lambda_j|$, and so y_j^k (here $j = 1$ or $j = 2$) will grow without bound as the iteration counter k increases, which does not reflect a qualitatively (or quantitatively!) accurate solution; Euler's method will be unstable. This means the maximum step size we can take is dictated by the smaller of the quantities $2/|\lambda_1|, 2/|\lambda_2|$, that is, the larger of $|\lambda_1|, |\lambda_2|$. If, for example, $\lambda_2 < \lambda_1 < 0$, it's λ_2 that dictates the largest permissible step size. But it may be the case that we're interested in the evolution of the system related to λ_1, which occurs on a longer time scale, long after the $e^{\lambda_2 t}$ part of the solution has decayed. Yet we're forced to take tiny steps with size less than $2/|\lambda_2|$ to track this solution.

To illustrate, suppose $\lambda_1 = -1$ and $\lambda_2 = -100$. Then $y_1(t) = y_1(0)e^{-t}$, while $y_2(t) = y_2(0)e^{-100t}$. Here $y_2(t)$ decays much more rapidly than $y_1(t)$ and very quickly will be insignificant in magnitude. If we are interested in tracking the decay of $y_1(t)$, however, we have to continue to iterate (11) and (12) in a stable manner, which requires $h < 2/100$, even though (11) itself would only require $h < 2$. This means taking unnecessarily many more iterations to get out to a large enough time to see the evolution of $\mathbf{y}(t)$. This system is (moderately) "stiff."

Example 3.2. Of course in the above example it would make sense to solve (9) and (10) independently, each with an appropriate step size, but most systems are not decoupled like this one. Consider instead a slightly more complicated (but still linear) system of the form

$$\begin{bmatrix} y_1'(t) \\ y_2'(t) \end{bmatrix} = \begin{bmatrix} -56 & 55 \\ 44 & -45 \end{bmatrix} \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}. \tag{13}$$

The eigenvalues and eigenvectors of the matrix on the right in (13) are $\lambda_1 = -1, \lambda_2 = -100$ with

corresponding eigenvectors $\mathbf{v}_1 =< \langle 1, 1\rangle$ and $\mathbf{v}_2 = \langle 1, -0.8\rangle$. As such, the general solution is

$$\mathbf{y}(t) = c_1 e^{-t} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + c_2 e^{-100t} \begin{bmatrix} 1 \\ -0.8 \end{bmatrix}. \tag{14}$$

Take initial conditions $y_1(0) = 0.5$ and $y_2(0) = 0.5$; these are slightly "rigged" initial conditions, to illustrate the vital point, but we'll consider something more typical below. We find that $c_1 = 0.5$ and $c_2 = 0.0$. The solution is

$$\mathbf{y}(t) = 0.5 e^{-t} \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

The e^{-100t} portion isn't even present in the true solution (and if it were, it would decay toward zero long before the solution above would).

It seems that a step size of $h = 0.1$ or smaller in Euler's method would do a pretty good job of tracking the solution, but this is not the case! A plot of $y_1(t)$ (which equals $y_2(t)$) is shown in Figure 3 in the left panel for $0 \le t \le 5$, along with the Euler iterates. However, this step size results

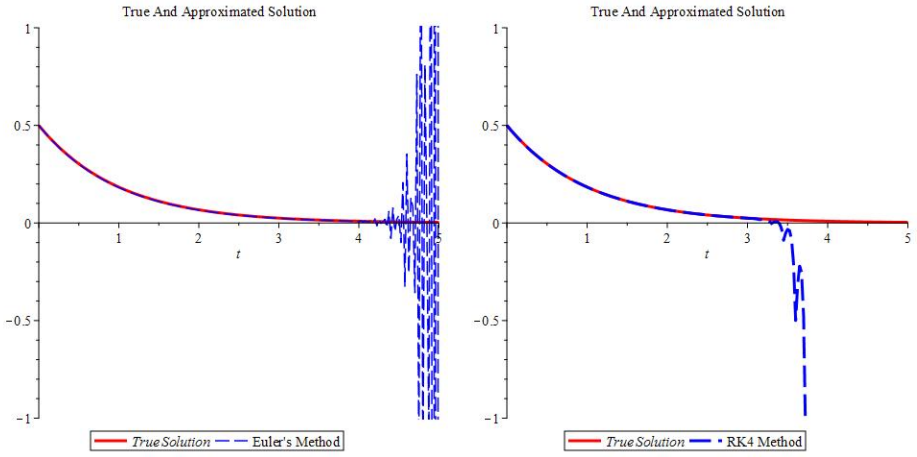

Figure 3. True solution and Euler's method with $h = 0.022$ (left) and RK4 method with $h = 0.03$ (right).

in disaster—the iterates quickly overflow floating point arithmetic. A bit of experimentation shows that no step size larger than $h = 0.02$ is stable in the long run. Switching to a more sophisticated numerical solver doesn't help much. In the right panel a Runge-Kutta 4th order method still falls apart when $h = 0.03$. Again, as with the decoupled system (9)-(10), it's the fastest decaying exponentials in the solution that dictate the largest stable step size. These fast decaying exponentials correspond to the largest magnitude eigenvalues in a linear system.

We can get a little more geometric insight into what's going wrong in this example by looking at the direction field for the system and a more typical solution. Suppose that $y_1(0) = 0.55$ and $y_2(0) = 0.57$. The solution to (13) is, to good a approximation,

$$\mathbf{y}(t) \approx 0.5611 e^{-t} \begin{bmatrix} 1 \\ 1 \end{bmatrix} - 0.0111 e^{-t/100} \begin{bmatrix} 1 \\ -0.8 \end{bmatrix}. \tag{15}$$

The direction field for the system (13) is shown in Figure 4, along with a parametric plot of the true solution $\mathbf{y}(t)$ (the black curve). Bear in mind that for aesthetic reasons, the arrows that delineate the direction field are not drawn to scale—they are much longer than indicated in Figure 4. In particular, the arrows that lie very far off the diagonal line $y_2 = y_1$ are rather long and point almost orthogonally toward this line, a geometric manifestation of the fact that solutions like (14) decay rapidly to a multiple of the vector $\langle 1, 1\rangle$, and then slowly toward the origin.

Of course the true solution in this case does exactly this, starting at $\langle 0.55, 0.57\rangle$ and quickly decaying to the line $y_2 = y_1$. At all points the true solution curve follows the direction field (that's what it means to be a solution!) However, the Euler iterates (shown in blue/dashed) with step size $h = 0.022$ do not behave like this. The first Euler step starts at $\langle 0.55, 0.57\rangle$; at this point the vector that dictates the direction the solution should go is $\langle 0.55, -1.45\rangle$, but a step of $h = 0.022$ puts the next iterate at $\langle 0.55, 0.57\rangle + 0.022\langle 0.55, 1.45\rangle \approx \langle 0.5621, 0.5381\rangle$. This overshoots the line $y_2 = y_1$ and actually ends up farther off this line than the starting point $\langle 0.55, 0.57\rangle$! This makes matters even worse, for the direction field has an even larger magnitude here, and the next iterate overshoots the other way. The process is repeated, with obvious effect. In every case, however, Euler's method is following the direction field toward the true solution, but due to the large magnitude of the direction field (that stems from the large eigenvalue of -100) Euler overshoots and oscillates in an unbounded manner.

The heart of the problem in this example is the large disparity in the magnitude of the eigenvalues. The portion of the solution (14) corresponding to the -1 eigenvalue or e^{-t} term dominates in most cases, at least after a very short time, and the solution decays smoothly and gradually to zero. But the portion of the solution corresponding to the eigenvalue of -100 is what limits the size of the steps we can take, no matter how small that portion of the solution might be. In many stiff systems one might encounter eigenvalues that vary over a much larger range, e.g., five or six orders of magnitude, or more! We are then forced to take time steps perhaps one million times smaller than the actual rate of change of the solution dictates. See Exercises 2 and 3 below.

This is the essence of stiffness, which does not have a precise or universally accepted definition. Nonetheless, the general idea is

> A system of ODE's is *stiff* if the step size required to maintain stability in a numerical ODE solver is small in relation to the scale on which the solution changes with respect to the independent variable.

This can occur in a linear system $\mathbf{x}' = \mathbf{A}\mathbf{x}$ if the eigenvalues of $\mathbf{A}$ (real or complex) are of widely varying magnitude. See Exercise 3 at the end for an example with complex eigenvalues. In a nonlinear system the phenomena can be even more complicated; the solution may be stiff in some intervals and not in others.

Modern numerical ODE solvers also incorporate "adaptive stepsizing." That is, the step size h is adjusted at each time step in order to control the (estimated) error of the solver at each step. When things are going poorly, h is decreased. This can combat stiffness to some extent, but in practice

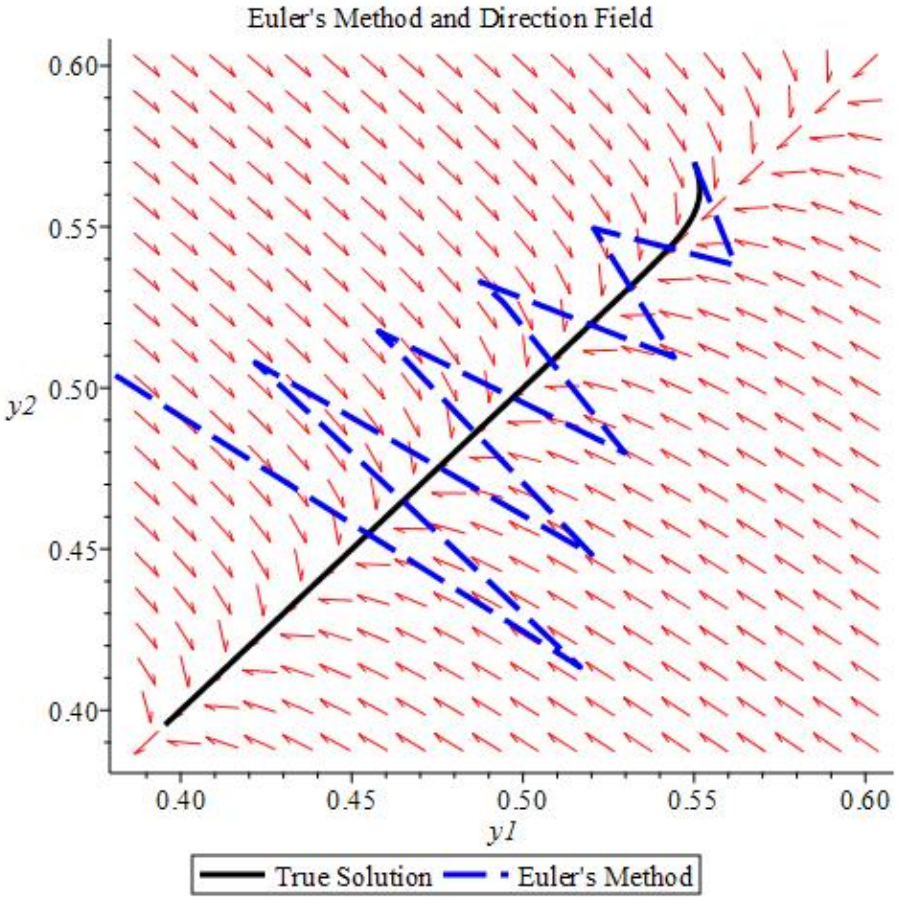

Figure 4. Euler iterates versus true solution and the direction field.

what ultimately happens is that h is reduced almost to 0 and the solver grinds to a halt. Another approach to overcoming stiffness is needed.

4 THE IMPLICIT EULER METHOD

If a system is stiff, what can be done? One common solution is to use an "implicit" ODE solver. Euler's method and most other numerical methods encountered in an introductory ODE course are "explicit, one-step" methods. What this means is that $\mathbf{y}^{k+1}$ is computed from the previous iterate $\mathbf{y}^k$, the time t_k, and the step size h (and the ODE itself, of course), in the form

$$\mathbf{y}^{k+1} = \mathbf{y}^k + h\phi(t_k, \mathbf{y}^k, h) \tag{16}$$

where $\phi(t, \mathbf{y}, h)$ is a function of the indicated variables. For example, for an ODE (1) with Euler's method we have $\phi(t, \mathbf{y}, h) = \mathbf{f}(t, \mathbf{y})$. For the Improved Euler method we have (using (7)) $\phi(t, \mathbf{y}, h) = \frac{1}{2}(\mathbf{f}(t, \mathbf{y}) + \mathbf{f}(t + h, \mathbf{y} + h\mathbf{f}(t, \mathbf{y})))$. The "explicit" refers to the fact that $\mathbf{y}^{k+1}$ is given explicitly in

(16). The "one-step" means that only $\mathbf{y}^k$ (and not previous iterates $\mathbf{y}^{k-1}, \mathbf{y}^{k-2}$, etc.) are involved in the right side of (16).

Explicit methods are convenient because we can "march" the iterates forward in time with a minimum of fuss. Their drawback, for stiff systems, is that the step size h has to be small enough to maintain stability of the method, even if the solution doesn't change rapidly. An alternative is to use an "implicit" method. These frequently remain stable for large (even arbitrarily large) step sizes.

The simplest example of such a method is the *implicit Euler* method. The implicit version of Euler's method for a scalar ODE (1) replaces the approximation (2) with a so-called "backwards difference"

$$y'(t) \approx \frac{y(t) - y(t-h)}{h} \tag{17}$$

(again, $h > 0$ here). The right side of (17) converges to $y'(t)$ as $h \to 0$, so if h is small enough the approximation of (17) should be accurate. With $t = t_k$ (17) becomes $y'(t_k) \approx (y(t_k) - y(t_{k-1}))/h$, which can be rearranged to $y(t_k) \approx y(t_{k-1}) + hy'(t_k)$. Finally, if y satisfies (1) then we have $y(t_k) \approx y(t_{k-1}) + hf(t_k, y(t_k))$. If y_k denotes our approximation to $y(t_k)$ this becomes a recipe for how to compute y_k, as a solution to $y_k = y_{k-1} + hf(t_k, y_k)$. Actually, we can shift indices $k \to k+1$ and write the implicit Euler method as

$$y_{k+1} = y_k + hf(t_{k+1}, y_{k+1}), \tag{18}$$

a prescription for computing y_{k+1} from y_k by solving (18). If f is sufficiently simple we might be able to solve for y_{k+1} in closed-form, but in most cases we actually have to use a numerical method (e.g., Newton's method) just to obtain y_{k+1}. Thus the implicit Euler's method probably requires more work at each iteration. The payback is that the method remains stable for large step sizes h. Equation (18) generalizes to systems in an obvious way.

Example 4.1. To illustrate, let's consider the ODE (5) again, with $y(0) = 1$, this time with the implicit Euler method. In this case equation (18) yields $y_{k+1} = y_k - h\lambda y_{k+1}$. Because f is so simple here, we can actually solve for y_{k+1} in closed form as

$$y_{k+1} = \frac{y_k}{1 + \lambda h}.$$

With $y_0 = 1$ its easy to see that in general

$$y_k = \frac{1}{(1 + \lambda h)^k}. \tag{19}$$

Contrast this to (6). In the case that $\lambda = -5$ and h is reasonably small, the implicit Euler method does a good job of approximating the true solution. But even if h is large—say $h = 0.5$—the implicit Euler method remains stable, and even positive, albeit a bit inaccurate. See Figure 5. In fact (19) makes it clear that any step size $h > 0$ will yield such an approximation.

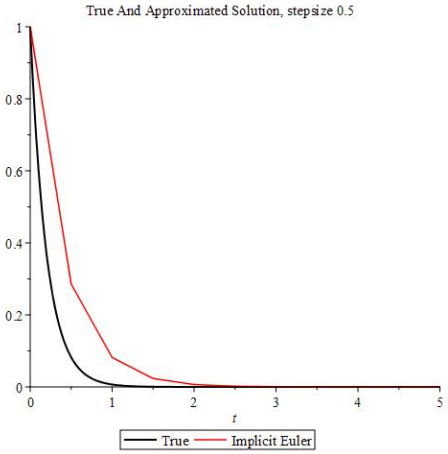

Figure 5. True solution and implicit Euler's method with $h = 0.5$.

Example 4.2. To further illustrate, let's apply the implicit Euler Method to the system (13). As with standard Euler's method, the implicit approach extends to a system of n ODE's (8) in n unknowns and leads to

$$\mathbf{y}^{k+1} = \mathbf{y}^k + h\mathbf{f}(t_{k+1}, \mathbf{y}^{k+1}), \tag{20}$$

a system of n (possibly nonlinear) equations in n unknowns, the components of the vector $\mathbf{y}^{k+1}$. For the system $\mathbf{y}' = \mathbf{A}\mathbf{y}$ of (13) this means we must solve

$$\mathbf{y}^{k+1} = \mathbf{y}^k + h\mathbf{A}\mathbf{y}^{k+1} \tag{21}$$

to compute $\mathbf{y}^{k+1}$ from $\mathbf{y}^k$ at each iteration. This is equivalent to solving $(\mathbf{I} - h\mathbf{A})\mathbf{y}^{k+1} = \mathbf{y}^k$ for $\mathbf{y}^{k+1}$, a linear system of equations that can be handled by Gaussian elimination or any linear solver. This is a fortunate consequence of the fact that our ODE's here are linear. In general the resulting equations for $\mathbf{y}^{k+1}$ could be nonlinear and something like Newton's method might be needed.

In fact in this case since $\mathbf{A}$ has eigenvalues -1 and -100, the matrix $\mathbf{I} - h\mathbf{A}$ has nonzero positive eigenvalues $1 + h$ and $1 + 100h$ if $h > 0$. Thus $\mathbf{I} - h\mathbf{A}$ must be invertible and so (21) will have a unique solution for $\mathbf{y}^{k+1}$.

Figure 6 shows the result of this iteration, a plot of the true solution $y_1(t) = e^{-t}$ and the implicit Euler iterates for y_1 with step size $h = 0.1$. Contrast this with the results of Figure 3. We can get away with a much larger step size here and still maintain stability. An excessively large step size will still result in a poor approximation though, just one that doesn't blow up!

SUMMARY

There are many other implicit methods for systems of ODE's. These methods often take the form

$$\psi(\mathbf{y}^{k+1}, \mathbf{y}^k, t_k, t_{k+1}) = 0, \tag{22}$$

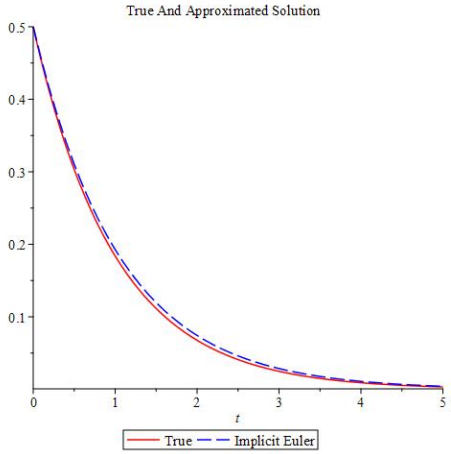

Figure 6. True solution and implicit Euler's method with $h = 0.1$.

for some function ψ, at least in a one-step version. Here $\mathbf{y}^{k+1}$ must be solved for, usually in some non-trivial way. An example is the *Trapezoidal Method*, which takes the form

$$\mathbf{y}^{k+1} = \mathbf{y}^k + \frac{h}{2}(\mathbf{f}(t_k, \mathbf{y}^k) + \mathbf{f}(t_{k+1}, \mathbf{y}^{k+1})). \tag{23}$$

The use of such a method is generally required if the equations are stiff and the solution is to be computed stably over an interval of any significant length. Most software packages for scientific computing (e.g., Maple, Mathematica, Matlab) have built-in solvers specifically designed for stiff systems of ODE's. See [1] for much more information on numerical methods for ODE's, stiff and otherwise.

5 EXERCISES

5.1 An RC Circuit

Consider the RC circuit in Figure 7, components and current directions as labeled. Let $q_1(t)$ denote the charge on capacitor C_1 and $q_2(t)$ the charge on capacitor C_2.

Suppose the open switch is closed at time $t = 0$ (and the capacitors C_1 and C_2 have some initial charge). From Kirchhoff's Voltage Law applied to the loop involving R_1 and C_1 we have

$$R_1 i_1(t) + q_1(t)/C_1 = 0. \tag{24}$$

For the loop involving C_1, C_2, and R_2 we have

$$R_2 i_3(t) + q_2(t)/C_2 - q_1(t)/C_1 = 0. \tag{25}$$

At the top junction between R_1, R_2, and C_1 we must have

$$i_1 - i_2 - i_3 = 0. \tag{26}$$

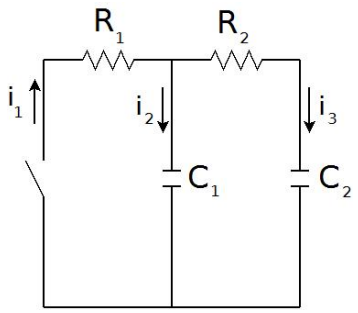

Figure 7. Two loop RC circuit.

Finally, we also have

$$q_1' = i_2 \tag{27}$$

$$q_2' = i_3. \tag{28}$$

Using (26) to eliminate i_1 in (24)-(25), then (27)-(28) to replace i_2, i_3, and finally solving for q_1' and q_2' leads to

$$q_1'(t) = -\left(\frac{1}{R_2C_1} + \frac{1}{R_1C_2}\right) q_1(t) + \frac{1}{R_2C_2} q_2(t) \tag{29}$$

$$q_2'(t) = \frac{1}{R_2C_1} q_1(t) - \frac{1}{R_2C_2} q_2(t) \tag{30}$$

a linear system of ODE's for functions $q_1(t), q_2(t)$.

Suppose $C_1 = C_2 = 0.001$ farad, $R_1 = 1000$ ohms, and $R_2 = 0.1$ ohm. The ODE's (29)-(30) become

$$q_1'(t) = -(1.0001 \times 10^7) q_1(t) + (1.0 \times 10^7) q_2(t) \tag{31}$$

$$q_2'(t) = (1.0 \times 10^7) q_1(t) - (1.0 \times 10^7) q_2(t). \tag{32}$$

With initial condition $q_1(0) = 0.000001$ and $q_2(0) = 0.000001$ the true solution to (31)-(32) is (to good approximation)

$$q_1(t) = (9.99975 \times 10^{-7}) e^{-499.99t} + (2.5 \times 10^{-11}) e^{(-2\times 10^7)t} \tag{33}$$

$$q_2(t) = (1.0 \times 10^{-6}) e^{-499.99t} - (2.5 \times 10^{-11}) e^{(-2\times 10^7)t}. \tag{34}$$

Note that both of $q_1(t)$ and $q_2(t)$ contain a comparatively large amount of $e^{-499.99t}$, a decaying exponential, and a very small multiple of $e^{(-2\times 10^7)t}$, a much faster decaying exponential (this term decays $40,000$ times faster!).

RC Circuit Exercises

1. Plot $q_1(t)$ and $q_2(t)$ on the range $0 \le t \le 0.01$. (They should look pretty similar).

2. It seems that a step size of $h = 0.0001$ would do a good job of tracking both of $q_1(t)$ and $q_2(t)$. Solve the system (31)-(32) numerically with Euler's method and $h = 0.0001$ on the interval $0 \leq t \leq 0.01$ (or as far as the numerics will go). What happens?

   Decrease h until the Euler iteration is stable over $0 \leq t \leq 0.01$. How small does h have to be?

3. Repeat the last problem but with the Implicit Euler method and step size $h = 0.0001$. Compare to the true solution.

4. Try the Trapezoidal Method (23). What step sizes yield stability?

Note the disparity of time scales in the circuit's physical behavior. The $e^{(-2\times10^7)t}$ stems from the capacitors discharging through the small R_2 resistor, and they quickly assume an equal charge; this occurs on a time scale of 10^{-7} seconds. But both capacitors then discharge through the much larger R_1 resistor at a comparatively slow rate—this is the $e^{-499.99t}$ term in the solutions, and this discharges occurs on a scale of 10^{-3} seconds. It may be this slower scale behavior that interests us, while the very rapid equalization of the capacitor charges that occurs when the switch is closed is of no interest. Nonetheless, it is this short time-scale phenomena that dictates the very small step size Euler's method must take to remain stable. We are thus forced to take many very small time steps in order to track the solution out to time 0.01. This problem is not confined to Euler's Method, but shared by any explicit method.

5.2 A Spring-Mass Problem

Consider a simple spring-mass system consisting of two masses m_1 and m_2, as illustrated in Figure 8. Let's assume the masses are constrained to move horizontally and that the only forces acting on either mass are the springs themselves and friction (no gravity). The spring attaching m_1 to the wall has spring constant k_1, while the spring connecting m_1 to m_2 has constant k_2. Suppose both springs have natural length 1.

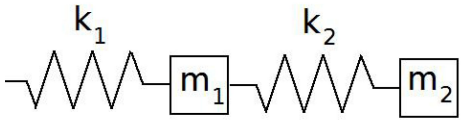

Figure 8. Double spring-mass system.

Let $x_1(t)$ denote the displacement of the spring attaching m_1 to the wall from its natural length (e.g., $x_1 = 0$ means the spring is at its natural length, neither stretched nor compressed) and $x_2(t)$ the same for the second spring. Assume both springs obey Hooke's Law. The position of the first mass would then be $1 + x_1(t)$ and the position of the second mass is $2 + x_1(t) + x_2(t)$. The velocity of the first mass is then $x_1'(t)$ and the velocity of the second mass is $x_1'(t) + x_2'(t)$.

The spring with constant k_1 exerts a force $-k_1x_1$ on m_1, while the spring with constant k_2 exerts a force $k_2(x_2 - x_1)$. Suppose also that m_1 experiences a frictional force proportional to its velocity,

say $-c_1 x_1'$ for some constant $c_1 > 0$. The net force on m_1 is thus $F_1 = -k_1 x_1 + k_2(x_2 - x_1) - c_1 x_1' = -(k_1 + k_2)x_1 + k_2 x_2 - c_1 x_1'$. From Newton's Second Law, $F = ma$, we have

$$m_1 x_1'' = -(k_1 + k_2)x_1 + k_2 x_2 - c_1 x_1'. \tag{35}$$

The force exerted by the spring with constant k_2 on m_2 is $-k_2(x_2 - x_1) = k_2 x_1 - k_2 x_2$. If the frictional force on m_2 is proportional to the velocity of m_2 then this force is of the form $-c_2(x_1' + x_2')$ for some $c_2 > 0$, and so

$$m_2 x_2'' = k_2 x_1 - k_2 x_2 - c_2(x_1' + x_2'). \tag{36}$$

With initial conditions $x_1(0) = a, x_1'(0) = v_1, x_2(0) = b, x_2'(0) = v_2$, equations (35)-(36) form a couple pair of linear second order ODE's. There is a unique solution.

If we let $y_1 = x_1, y_2 = x_1', y_3 = x_2$, and $y_4 = x_2'$, the system (35)-(36) can be formulated as a first order system of four differential equations in four unknowns

$$\begin{aligned}
y_1' &= y_2 \\
y_2' &= -\frac{k_1 + k_2}{m_1} y_1 - \frac{c_1}{m_1} y_2 + \frac{k_2}{m_1} y_3 \\
y_3' &= y_4 \\
y_4' &= \frac{k_2}{m_2} y_1 - \frac{c_2}{m_2} y_2 - \frac{k_2}{m_2} y_3 - \frac{c_2}{m_2} y_4
\end{aligned}$$

or, more compactly, as $\mathbf{y}' = \mathbf{A}\mathbf{y}$ where

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1+k_2}{m_1} & -\frac{c_1}{m_1} & \frac{k_2}{m_1} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_2}{m_2} & -\frac{c_2}{m_2} & -\frac{k_2}{m_2} & -\frac{c_2}{m_2} \end{bmatrix}. \tag{37}$$

Spring-Mass Exercises

1. Take $k_1 = 1, k_2 = 10^4, m_1 = 1, m_2 = 0.001, c_1 = c_2 = 0.1$. Compute $\mathbf{A}$ and its eigenvalues (they are complex). What exponential functions govern the decay rates in this problem? What natural frequencies are present?
2. With the parameters as in (1) above and initial conditions $y_1(0) = 0.1$ and $y_2(0) = y_3(0) = y_4(0) = 0$. it turns out that the true solution for $y_1(t)$ that governs the motion of the first mass is (to a few significant figures)

$$\begin{aligned}
y_1(t) \approx\ & e^{-49.9t}((9.98 \times 10^{-5})\cos(3163.5t) + (4.72 \times 10^{-6})\sin(3163.5t)) \\
& + e^{-0.1499t}(0.0999\cos(0.9882t) + 0.0051\sin(0.9882t)).
\end{aligned}$$

   Note the fast oscillating term is much smaller and decays much faster than the more slowly oscillating term. Plot this for $0 \le t \le 50$. Repeat on a smaller interval, e.g., $0 \le t \le 5$. The graphs of $y_2(t), y_3(t)$, and $y_4(t)$ look fairly similar. It seems a step size of $h = 0.01$ would track the solution well.

3. Solve the system with Euler's method and $h = 0.01$ out to time $t = 5$, or however far it will go. Is it stable?
4. Repeat with the Implicit Euler method. Is the numerical solution stable? Accurate?
5. Repeat the last question with the Trapezoidal Method (23).

REFERENCES

[1] Butcher, J.C., 2016. *Numerical Methods for Ordinary Differential Equations, Third Edition.* New York; John Wiley & Sons, Inc.