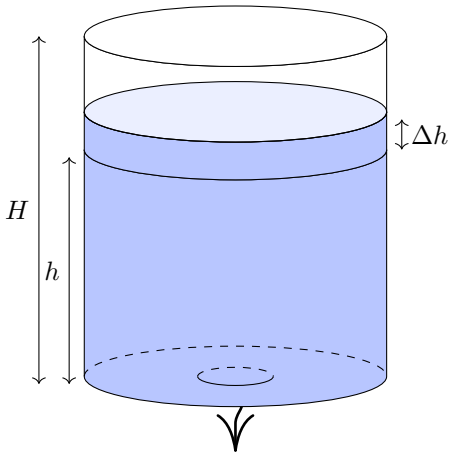
VARIATIONS ON A THEME OF EULER

ROBERT M. CORLESS AND JULIA E. JANKOWSKI

This paper is an exploration of numerical methods for solving initial-value problems for ordinary differential equations. We look in detail at one "simple" problem, namely the leaky bucket, and use it to explore the notions of explicit marching methods *vs.* implicit marching methods, interpolation, backward error, and stiffness. While the leaky bucket example is very well known, and these topics are studied in a great many textbooks, we will here emphasize backward error in a way that might be new and that we hope will be useful for your students. Indeed the paper is intended to be a resource for students themselves. We will also use two techniques not normally seen in a first course, a new one, namely "optimal backward error", and an old one, namely "the method of modified equations".

1. Torricelli's Law and the Leaky Bucket. Quite a few textbooks use Torricelli's Law, often derived using Bernoulli's Law for steady flow, to write down a differential equation modelling the height of fluid in a bucket with a hole in the bottom. For example, the beautiful book [13]—really one of our favourite textbooks of all time—has an extensive treatment in their section 4.2 "Uniqueness: The Leaking Bucket Versus Radioactive Decay". Their treatment even includes backward error, which they call 'slope error'. We (partially) disagree with one thing that they say, however: on p.172 we find the statement (emphasized in the original) *"numerical methods are inherently untrustworthy, and anything that you might guess using them, especially if it concerns long term behaviour, must be checked by other methods."* We believe that statement is only partly true, and this is a good thing too because for many (most?) problems in science nowadays there is *no other way* of obtaining insight (or control or prediction) than by using numerical methods. The purpose of the treatment here is to help learn when to trust numerical methods, and when not to.

Following [13] we use an energy balance to derive an equation for the height of fluid in a leaky bucket.



fluid density ρ, velocity v

The bucket is assumed to be cylindrical with cross-sectional area A, with perfectly vertical walls and a perfectly flat bottom, while the hole in the bottom has an area a. Thus the volume of fluid lost out the hole in a time interval $\Delta\tau$ is $a \cdot v(\tau)\Delta\tau$ (if $\Delta\tau$ is small enough that the velocity $v(\tau)$ can be considered constant). At the same time, the volume lost is $A \cdot h'(\tau)\Delta\tau$ where $h'(\tau) = dh/d\tau$; by conservation these are equal, so

$$av(\tau) = A\frac{dh}{d\tau}. \tag{1.1}$$

The change in potential energy in this time $\Delta\tau$ is $(\rho A\Delta h)gh$ because the mass is the product of ρ—the density of the fluid—with the volume $A\Delta h$, and the change in energy occurs over the height h because the leak is at the bottom. If there is no energy dissipation (we will revisit this later) then this must equal

the kinetic energy of the fluid flowing out, $\frac{1}{2}[\rho A \Delta h]v^2$, plus the increase in kinetic energy of the (slowly) downward flowing fluid in the bucket itself, which like the dissipation we ignore. Thus $\rho A \Delta h g h = \frac{1}{2}\rho A \Delta h v^2$ or

$$2gh = v^2 = \left(\frac{A}{a}\frac{dh}{d\tau}\right)^2 . \tag{1.2}$$

Thus at last we have

$$\frac{dh}{d\tau} = -\frac{a}{A}\sqrt{2gh} \tag{1.3}$$

and we take the negative sign because the height is decreasing over time. Let us now nondimensionalize. Put $y = h/H$ where H is the height of the bucket itself. Then $y = 1$ corresponds to a bucket brim full of liquid while $y = 0$ means the bucket is empty. Dividing (1.3) by H we have

$$\frac{dy}{d\tau} = -\frac{a}{A}\sqrt{\frac{2g}{H}} \cdot \sqrt{y} \, . \tag{1.4}$$

Now we rescale time. Put

$$t = \frac{a}{A}\sqrt{\frac{2g}{H}}\, \tau \tag{1.5}$$

and so by the chain rule $\frac{dy}{d\tau} = \frac{dt}{d\tau}\,\frac{dy}{dt}$ and the scale factors cancel leaving

$$\frac{dy}{dt} = -\sqrt{y} \, . \tag{1.6}$$

The initial condition is thus $y(0) = y_0$, the initial fraction full: $0 \le y_0 \le 1$. We will almost always take $y_0 = 1$.

Hubbard and West [13] use this example to great effect for studying uniqueness of solutions. Given an empty bucket, there's no way to tell when it was full—if it ever was. This fact is well-displayed in the model, as we will see. The reference solution[1] of this separable equation is easy: $y(t) = (\sqrt{y_0} - t/2)^2$.

This solution is valid only so long as $y > 0$. Investigating when $y = 0$ separately we have $\frac{dy}{dt} = 0$ thereafter, so

$$y(t) = \begin{cases} \left(\sqrt{y_0} - t/2\right)^2 & 0 \le t < 2\sqrt{y_0} \\ 0 & 2\sqrt{y_0} \le t \end{cases} \tag{1.7}$$

and by inspection this is continuous, even differentiable, at $T = 2\sqrt{y_0}$. In dimensional terms, the discharge time when $y_0 = 1$ is predicted to be

$$\tau_{\text{T}} = \frac{A}{a}\sqrt{\frac{H}{2g}} \cdot 2 = \frac{A}{a}\sqrt{\frac{2H}{g}} \, . \tag{1.8}$$

Notice that $y(t) \ge 0$ always, and of course this is necessary for the $\sqrt{y(t)}$ in equation (1.6) to be real.

All this is fine, and educational. We don't even need numerical methods for this problem. But, for other equations, we do; and we claim it's worthwhile to have a look at what numerical methods do on this

[1] We use the phrase "reference solution" instead of the more common "exact solution", because starting in section 4 we will see everything is an "exact solution" of some relevant equation.

nonlinear problem. Before we do, though, notice one more fact about this equation: the differential equation is *not* Lipschitz continuous at $y = 0$, because $\frac{\partial}{\partial y} - \sqrt{y} = -1/(2\sqrt{y})$ is infinite there; hence there is no finite L such that $|\sqrt{y_1} - \sqrt{y_2}| \leq L \cdot |y_1 - y_2|$. This means that the classical existence and uniqueness theorems fail to apply, see e.g theorems 5 and 6, p 28 of [1]. Indeed this is the whole point of Section 4.2 of [13]. Moreover, the hypotheses for some theorems, see e.g. theorem 3.4 in [12], or theorem 203A in [3], about the convergence of numerical methods therefore *also* must fail to hold. So let's see what happens, in detail.

2. Forward Euler for the Leaky Bucket. Euler's method for $y'(t) = f(t, y)$ marches forward with the iteration $y_{n+1} = y_n + \Delta t_n f(t_n, y_n)$ where $t_{n+1} = t_n + \Delta t_n$. This is intended to give the *skeleton*[2] $\{(t_n, y_n)\}_{n=0}^N$ of an approximate solution on some time interval $t_0 \leq t \leq t_N$. The sequence of time steps Δt_n is chosen either for convenience (all $\Delta t_n =$ some fixed Δt), for reliability (take each Δt_n small enough to ensure accuracy), or for efficiency (take each Δt_n as large as possible given constraints on accuracy). For now, take the convenient route and fix $\Delta t_n = \Delta t$. Our iteration, for equation (1.6), becomes

$$y_{n+1} = y_n - \Delta t \sqrt{y_n} \,. \tag{2.1}$$

Taking $\Delta t = 1$, to start with, we find that a full bucket $y_0 = 1$ empties in one time step: $y_i = 1 - 1 \cdot \sqrt{1} = 0$. It remains empty thereafter. Possibly this time step is too large. Taking $\Delta t = 1/2$ instead, $y_1 = 1 - \frac{1}{2}\sqrt{1} = \frac{1}{2}$, and $y_2 = \frac{1}{2} - \frac{1}{2} \cdot \sqrt{\frac{1}{2}} \doteq 0.146446, y_3 \doteq -0.04489$ is negative, and y_4 is complex! This doesn't seem right.

Taking a more artful time step, we choose $\Delta t = \frac{2}{1+\sqrt{5}} \doteq 0.618...$ We will see the reason for this choice in a moment. Then $y_0 = 1$ gives

$$y_1 = 1 - \frac{2}{1 + \sqrt{5} \cdot 1} \;=\; \frac{-1 + \sqrt{5}}{1 + \sqrt{5}} = \frac{4}{(1 + \sqrt{5})^2} \tag{2.2}$$

and

$$y_2 = \frac{4}{(1 + \sqrt{5})^2} - \frac{2}{(1 + \sqrt{5})} \cdot \frac{2}{(1 + \sqrt{5})} = 0 \,. \tag{2.3}$$

Thus with this Δt the bucket empties in two steps, at a time $\Delta t + \Delta t = 4/1 + \sqrt{5} = 1.236$, larger than with $\Delta t = 1$.

Another artful choice is $\Delta t = \frac{2}{1+\sqrt{7+2\sqrt{5}}} \doteq 0.45588...$ You, the reader, can verify that $y_0 = 1$ leads in *three* steps to $y_3 = 0$ and an empty bucket, in a time $3\Delta t \doteq 1.26764$, still larger.

It turns out that taking Δt smaller doesn't generally help, except temporarily, unless we're artful and choose just those steps such that $y_N = \Delta t^2$ exactly for some lucky N, when $y_{N+1} = 0$ exactly and the bucket empties. For all other Δt, which means almost all Δt, the iteration terminates when some $y_N < (\Delta t)^2$ and then $y_{N+1} = y_N - \Delta t \sqrt{y_N} < 0$. Once any $y_{N+1} < 0$ the next iterate must be complex.

Something important has happened. The forward, explicit, Euler scheme seems to have generically failed for this problem. Not, it's true, till the size of the solution is $\mathcal{O}(\Delta t^2)$, and a reasonable human, computing the solution by hand, would stop at this point and declare the bucket empty; especially since taking Δt smaller and doing it again would get us to an even more nearly empty bucket. But dumb computers, on the other hand... We emphasize that it is quite unusual for the explicit Euler method to fail to be able to take a step[3]. Even for problems with singular solutions its tendency is to keep going!

[2] In older books, numerical methods *only* produce approximate solutions at discrete points: $\{(t_n, y_n)_{n \geq 0}\}$. We call this just the bare bones of the solution now, hence "skeleton". Numerical solutions of ODE include interpolants, nowadays.

[3] Apparently, Euler knew this could happen.

We will revisit this, but first note that we can remove one parameter from the iteration by putting $y_n = f_n(\Delta t)^2$, whence equation (2.1) becomes

$$f_{n+1} = f_n - \sqrt{f_n}\,. \tag{2.4}$$

Of course now $f_0 = \frac{y_0}{(\Delta t)^2}$ can be quite large but this further "nondimensionalization" will be helpful. We now see that the source of the 'artful' choices for Δt: We look at the preimages of 0 by inverting $f_{n+1} = f_n - \sqrt{f_n}$:

$$f_n = \left(\frac{1+\sqrt{1+4f_{n+1}}}{2}\right)^2 \tag{2.5}$$

giving us 1, $\left(\frac{1+\sqrt{5}}{2}\right)^2$ $\left(\text{whence } \Delta t = \frac{2}{1+\sqrt{5}}\right)$, and $\left(\frac{1+\sqrt{1+(1+\sqrt{5})^2}}{2}\right)^2 = \left(\frac{1+\sqrt{7+2\sqrt{5}}}{2}\right)^2$, and so on. For $y_0 = f_0/(\Delta t)^2 = 1$ we see that taking $\Delta t = 1/\sqrt{f_{-N}}$ for any preimage f_{-N} of 0, the bucket empties in a finite number of steps. But for almost all Δt, and certainly if there's rounding errors, we will miss these 'sweet spots' and wind up with complex y. Let's try another method.

Remark:

One reviewer pointed out that we have some freedom in coding the differential equation. We don't have to limit ourselves to just the equation as it was derived, and we can try to head off any difficulties that we might foresee. Since $\sqrt{y}$ occurs in the equation, we might anticipate trouble if ever y becomes less than 0. It shouldn't, we know, and we shouldn't have to worry, but prudent programmers put in safeguards. One way to do this (suggested by the reviewer) is to code the equation as

$$\frac{dy}{dt} = \begin{cases} -\sqrt{y} & \text{if} \quad y > 0 \\ 0 & \text{if} \quad y \le 0 \end{cases} \tag{2.6}$$

This way at least if y overshoots the equilibrium when the bucket empties only by a small amount, then thereafter the solution will stay constant at that small negative value, which a reasonable human might accept with a wink. However, this did not come from the modelling, but rather from some geometric analysis of the model. Indeed, some works studying Torricelli's law work from the equation $(\dot{y})^2 + y = 0$ and give a decidedly different analysis [14].

Even if an implicit method is used it needs to avoid negative y in the solution to the implicit equation. If a nonlinear root finder is used in the process of taking a step it is quite possible that negative y might be sampled in the course of solving the implicit expression. The reformulation discussed would here avoid any possible problems with a nonlinear solver without changing the behaviour of backward Euler.

Further, nowadays most software actually used for solving ODEs can handle events such as y reaching zero. It is important for students to know that this sort of issue arises in many practical applications, and has been successfully recognized and dealt with by software developers. A better "precautionary coding" then might be to use this feature. That is, instead of integrating over a fixed time interval, one integrates until the event $y = 0$ is detected. But we now return to the more primitive situation, and consider another method.

3. Euler Backward for the Leaky Bucket. Implicit Euler, also known as Backward Euler or Euler Backward, is of course $y_{n+1} = y_n + \Delta t f(t_{n+1}, y_{n+1})$. Here we have

$$y_{n+1} = y_n - \Delta t\sqrt{y_{n+1}}. \tag{3.1}$$

Normally, *ie* for general nonlinear f, this equation must be solved numerically at each step, making the process expensive and worth the bother only when there is something to gain: typically an implicit method is only better than an explicit method when the problem is "stiff". Indeed that's the pragmatic *definition* of a stiff problem: a stiff problem is one for which an implicit method is more effective. For a more mathematical discussion of stiffness see [16].

Here, though, we can usefully solve the nonlinear equation once and for all and get an explicit iteration again. Put $p = \sqrt{y_{n+1}}$. Notice $p > 0$. Then $p^2 + \Delta t p - y_n = 0$ is just a quadratic equation and the positive root can be written

$$p = \frac{-\Delta t + \sqrt{(\Delta t^2) + 4y_n}}{2} \tag{3.2}$$

$$= \frac{-\Delta t + \sqrt{(\Delta t^2) + 4y_n}}{2} \cdot \frac{\Delta t + \sqrt{(\Delta t^2) + 4y_n}}{\Delta t + \sqrt{(\Delta t^2) + 4y_n}} \tag{3.3}$$

$$= \frac{-(\Delta t^2) + (\Delta t^2) + 4y_n}{2(\Delta t + \sqrt{(\Delta t^2) + 4y_n})} \tag{3.4}$$

$$= \frac{2y_n}{\Delta t + \sqrt{(\Delta t^2) + 4y_n}}\,, \tag{3.5}$$

using a standard trick to avoid catastrophic cancellation when y_n is very small [4], so the formula

$$y_{n+1} = p^2 = \frac{4y_n^2}{(\Delta t + \sqrt{(\Delta t^2) + 4y_n})^2} \tag{3.6}$$

lets us iterate implicit Euler in an explicit manner. Of course this is unusual and only possible because the example is so "simple". Again we can scale by (Δt^2) to remove a parameter: put $y_n = b_n(\Delta t^2)$ so

$$b_{n+1} = \frac{4}{(1 + \sqrt{1 + 4b_n})^2} b_n^2\,. \tag{3.7}$$

Taking a few possible initial values, say $b_0 = 10$ or $b_0 = 1$, gives quite different behaviour to forward Euler. Taking $b_0 = 10$ is typical: after only a few iterations[4], five in this case, $b_5 \doteq 0.9193$ is less than 1, and thereafter convergence to 0 is quadratically fast—reminiscent of Newton's method—and indeed $b_{15} \doteq 10^{-302}$ and b_{16} underflows to 0. This behaviour is universal so long as $b_0 < \mathcal{O}(1/\varepsilon_m)$, where ε_m is the machine epsilon. No matter what other initial condition $b_0 > 0$ is given, eventually some $b_N < 1$ and thereafter the bucket empties very rapidly. In perfect arbitrary precision arithmetic it would never *actually* empty, but getting values of $y_n = b_n \Delta t^2$ smaller than the reciprocal of Avogadro's number[5] seems like overkill for a continuum fluid model[6].

It seems Euler backward is better for the leaky bucket. This indicates that the DE $\dot{y} = -\sqrt{y}$ is "stiff" near $y = 0$, and indeed it is. But there's more to say.

4. Optimal Backward Error. Since the 1980's, at least, people have known that one way to assess the error of a numerical method is to interpolate the skeleton (somehow) and compute the residual (also called the defect, the deviation, or the slope error). Specifically and without loss of generality, suppose that the interpolant on $t_n \le t \le t_{n+1}$ is called $z_n(t)$; the residual is then

$$r_n(t) := z_n'(t) - f(t, z_n(t)) \tag{4.1}$$

on that same interval. Then the piecewise continuous function $z(t)$ defined by

$$z(t) = \begin{cases} z_0(t) & t_0 \le t \le t_1 \\ z_1(t) & t_1 \le t \le t_2 \\ \vdots & \\ z_{N-1}(t) & t_{N-1} \le t \le t_N \end{cases}$$

[4] If $b_n > 1$, $4b_n/(1 + \sqrt{1 + 4b_n})^2 < 1$ and $b_{n+1} < b_n$; if $b_n \le 1$, $b_{n+1} < b_n^2$.

[5] Avogadro's number, 6.0221413×10^{23}, counts the number of molecules in a mole. A 10-litre bucket of water would have about 55.5 moles in it, when full.

[6] This discussion echoes a point made by L.N. Trefethen in [17]: the continuum model breaks down before floating point does!

satisfies the differential equation

$$z'(t) = f(t, z(t)) + r(t) \tag{4.2}$$

where the piecewise function $r(t)$ is defined similarly as

$$r(t) = \begin{cases} r_0(t) & t_0 \le t < t_1 \\ r_1(t) & t_1 \le t < t_2 \\ \vdots & \\ r_{N-1}(t) & t_{N-1} \le t \le t_N \end{cases}$$

where now $r(t)$ can be discontinuous at the skeleton. This remark bears emphasis. The computed solution $z(t)$ is the exact solution of a *different* differential equation. If $r(t)$ is very small, but the equation $\dot{y} = f(y)$ is somehow perfect and must not be changed, then this observation is only indirectly helpful: one must appeal e.g to the Gröbner-Alekseev nonlinear variation of constants formula to say

$$y(t) - z(t) = \int_0^t G(t, \tau) r(\tau) d\tau \tag{4.3}$$

and then bound G in order to bound the forward error [12]. Here, $G(t, \tau)$ is a function that depends on the DE and the reference solution y and can in principle be computed *a priori*, and is a kind of nonlinear Green's function. See [12] for details.

But if, as is generally true in science, the differential equation is only one model among many, then we may already be finished[7]. Imagine, for instance, the bucket being shaken gently with time: the dynamic effects on the fluid would be quite complicated but one would be tempted to model them with a small time-dependent forcing term. And anyone who's ever emptied a bottle knows that the flow regime at the end is quite different to what's gone before. Constant velocity v is not very realistic when the bucket is nearly empty! One has to worry about the impact of such forcing and of the details of the fluid, e.g. its viscosity—but then, one has to do that anyway in a responsible solution. 'Everybody knows this', but not every textbook or course or software package emphasizes it.

For numerically generated $r(t)$ it's well known that the value of $r(t)$ depends on the quality of the interpolant. One needs high-order accuracy throughout the time step to have the residual be small everywhere in the interval. For general solvers, considerable attention is paid to this issue [8]. Indeed, one can wonder if some other interpolant could produce a smaller $r(t)$, as Hubbard and West do: "It is unclear from this derivation whether a wigglier curve joining curve points might even give a higher order dependence on $r(t)$; however, looking at the actual evidence in section 3.3 shows that it won't." [13, p. 183]

We're going to answer these doubts analytically. We will think about an *optimal* interpolant for this problem, and compute the smallest possible backward error explicitly. The scope and utility of this technique for general systems $f \in \mathbb{C}^n$ is explored in [6]. Its use for the Dahlquist test problem—which generates some surprises—is explored in [7]. For the present example we do not need a general method of optimization and we can just use the triangle inequality. It's better here to think about *relative* backward error, also. That is, instead of

$$\frac{dz}{dt} = -\sqrt{z(t)} + r(t) \tag{4.4}$$

we use $\delta(t) = -r(t)/\sqrt{z(t)}$, so $\delta(t)$ is the residual $r(t)$ divided by what the derivative should be, to get

$$\begin{aligned} \frac{dz}{dt} &= -\sqrt{z(t)} + r(t) \\ &= -\sqrt{z(t)} - \sqrt{z(t)}\delta(t) \\ &= -\sqrt{z(t)}(1 + \delta(t)) \,. \end{aligned} \tag{4.5}$$

[7] Well, we still have to study conditioning or sensitivity, perhaps by using Gröbner-Alekseev. But now the point is robustness, not forward error.

so $r(t) = -\sqrt{z(t)}\delta(t)$ and

$$-\frac{1}{\sqrt{z}}\frac{dz}{dt} - 1 = \delta(t)\,. \tag{4.6}$$

Integrating both sides over the time step,

$$\int_{s=t_n}^{t_{n+1}} -\frac{1}{\sqrt{z(s)}}\dot{z}(s)\ \ ds - \int_{s=t_n}^{t_{n+1}} 1\ \ ds = \int_{s=t_n}^{t_{n+1}} \delta(s)\ \ ds \tag{4.7}$$

so

$$2\sqrt{y_n} - 2\sqrt{y_{n+1}} - \Delta t = \int_{s=t_n}^{t_{n+1}} \delta(s)\ \ ds \tag{4.8}$$

because $z(t_n) = y_n$ and $z(t_{n+1}) = y_{n+1}$ and $t_{n+1} = t_n + \Delta t$. This constrains how small $\delta(t)$ can be: in other words it has to be large enough to push the solution to y_{n+1}. One can show from this that if y_{n+1} was generated from y_n by a p^{th} order method, then $\|\delta\| = \mathcal{O}(\Delta t^p)$. For Euler's method, $p = 1$.

Explicitly, for Euler's method, we have $y_{n+1} = y_n - \Delta t\sqrt{y_n}$ and so

$$\begin{aligned}
2\sqrt{y_n} - 2\sqrt{y_{n+1}} - \Delta t &= 2\sqrt{y_n} - 2\sqrt{y_n - \Delta t\sqrt{y_n}} - \Delta t \\
&= 2\sqrt{y_n} - 2\sqrt{y_n}(1 - \frac{\Delta t}{\sqrt{y_n}})^{\frac{1}{2}} - \Delta t \\
&= 2\sqrt{y_n} - 2\sqrt{y_n}(1 - \frac{\Delta t}{2\sqrt{y_n}} - \frac{1}{8}\frac{\Delta t^2}{y_n} + \mathcal{O}(\Delta t^3)) - \Delta t \\
&= 2\sqrt{y_n} - 2\sqrt{y_n} + \Delta t - \Delta t + \frac{1}{4\sqrt{y_n}}\Delta t^2 + \mathcal{O}(\Delta t^3) \\
&= \frac{1}{4\sqrt{y_n}}\Delta t^2 + \mathcal{O}(\Delta t^3)
\end{aligned} \tag{4.9}$$

Since

$$\int_0^{\Delta t} \delta(s)\ ds = \bar{\delta}\Delta t \tag{4.10}$$

where $\bar{\delta}$ is the average value of $\delta(s)$, we have

$$\bar{\delta}\Delta t = \frac{1}{4\sqrt{y_n}}\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{4.11}$$

or

$$\bar{\delta} = \frac{1}{4\sqrt{y_n}}\Delta t + \mathcal{O}(\Delta t^2)\,. \tag{4.12}$$

That is, on average, $\delta(s)$ must be $\mathcal{O}(\Delta t^p)$, with $p = 1$. This means that the maximum δ must be at least $\mathcal{O}(\Delta t)$; and we will now see that we can find an interpolant so that δ is exactly this size.

From equation (4.8), the triangle inequality gives

$$\left|2\sqrt{y_n} - 2\sqrt{y_{n+1}} - \Delta t\right| = \left|\int_{t_n}^{t_{n+1}} \delta(s)ds\right| \le \Delta t \cdot \|\delta(t)\|_\infty \tag{4.13}$$

and equality is attained when $\delta(t)$ is constant, namely $\delta(t) \equiv \delta_n$, where $2\sqrt{y_n} - 2\sqrt{y_{n+1}} - \Delta t = \Delta t\delta_n$ or

$$\delta_n = \frac{2}{\Delta t}(\sqrt{y_n} - \sqrt{y_{n+1}}) - 1\,. \tag{4.14}$$

For clarity and emphasis we're going to repeat ourselves. The triangle inequality proves that

$$\|\delta\|_\infty \geq \left| \frac{2}{\Delta t}(\sqrt{y_n} - \sqrt{y_{n+1}}) - 1 \right|, \tag{4.15}$$

giving a lower bound on how big $\delta(t)$ must be to have $dz/dt = -\sqrt{z}(1+\delta(t))$ starting with $z(t_n) = y_n$ and making it all the way to $z(t_{n+1}) = y_{n+1}$. Remember y_{n+1} is determined by the numerical method.

Moreover this lower bound is achieved if $\delta(t) = 2(\sqrt{y_n} - \sqrt{y_{n+1}})/\Delta t - 1$ is constant: That is, this $\delta(t)$ achieves transit from y_n to y_{n+1} with the least possible relative deviation from the original model. Thinking about backward error in this way is likely new to you. We hope to convince you it will be valuable for your students.

Remark: This solution works *only* for this problem so *long* as $z(t)$ is never zero in the interval; in particular it fails when $y_{n+1} < 0$. For that, we must use $\Delta(t)$ instead. Clearly if $\Delta(t)$ is nonzero but $z(t)$ is zero the relative error $\delta(t)$ is infinite. That's an artifact of this example, and tells us something about the model.

Equation (4.14) is quite interesting. It gives the minimum possible backward error for the piece of the skeleton (t_n, t_y) to (t_{n+1}, y_{n+1}). If we solve $\dot{z} = -\sqrt{z}(1+\delta_n)$ on this interval, we have the "best possible" interpolant to this piece of the skeleton. The solution here is just as easy as the reference solution: $z(t) = (\sqrt{y_n} - (1+\delta_n)(t-t_n)/2)^2$. Most importantly this analysis is independent of the method used to generate the skeleton. It applies to Forward Euler, Backward Euler, or indeed to any other method.

For both forward Euler and Euler backward one can use equation (4.14) to show that $\|\delta(t)\| = \mathcal{O}(\Delta t)$ as $\Delta t \to 0$. Together with the Gröbner-Alekseev nonlinear variation of constants formula this establishes convergence of $z(t)$ to $y(t)$ with forward error (on $0 \leq t \leq 2\sqrt{y_0} - \varepsilon$ for every $\varepsilon > 0$ since we need G to be bounded) that goes to 0 as $\mathcal{O}(\Delta t)$, also. But in practice one never takes $\Delta t \to 0$. One always uses a finite number of steps, and indeed one wants to take steps Δt as large as possible, for efficiency. This often means that we are outside the asymptotic ($\Delta t \to 0$) region.

A very important feature of equation (4.14) (and the similar equations that can be developed for other problems, even just using suboptimal interpolants) is that δ_n is computable. In this simple example, so is $z(t)$; in general that's harder, though. This gives a good retrospective diagnostic: after doing the computation, we can assess δ_n to see if our results are good enough. How does it work for these examples?

Consider figures 4.1 and 4.2. We see that for *both* methods the relative error rises as $y_n \to 0$. This is not really a surprise. What's nice is that for, say, $\Delta t = 1/30$, we see that the equation we've solved is, on $0 \leq t \leq 1.9$, no more than 5% different to the one we wanted to solve. This simple tool can be used for many equations.

Still, somehow, this "optimal backward error" may not satisfy someone who thinks of the differential equation as being "smooth" — here we have obvious pulses, changing with every time step. There's another, older, technique that can help, which we take up in section 6. But first, we look at optimal residuals for Matlab's solution.

5. Optimal Backward Error for Matlab's Solutions. The technique of optimal backward error can be used to analyze the skeleton produced by any numerical method. In figures 5.3, 5.6, and 5.9 we see the results from analyzing ode45, ode15s, and ode113, the three most popular Matlab codes [15], each applied to the leaky bucket. We use the default tolerances for each method. In figure 5.1 we plot not the *optimal* backward error but rather the backward error using the supplied piecewise polynomial interpolant. We see that it's fairly small though not so small as the optimal, which is plotted in figure 5.2. Similarly figure 5.4 plots the residual using the supplied interpolant for ode15s, while figure 5.5 shows the optimal

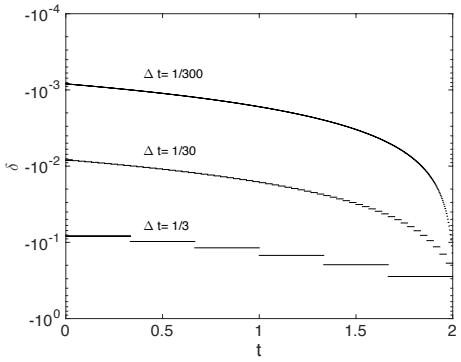


FIG. 4.1. *Optimal Backward Error for the implicit Euler method for Torricelli's Law,* $\dot{y} = -\sqrt{y}$, $y(0) = 1$. *We plot* $\delta_n = 2 \cdot (\sqrt{y_n} - \sqrt{y_{n+1}})/\Delta t - 1$ *for solutions using three different fixed time steps,* $\Delta t = 1/3, \Delta t = 1/30, \Delta t = 1/300$. *We see convergence away from the singularity when the bucket empties at* $t = 2$. *We also see the error get large near* $t = 2$. *Notice that* $\delta_n < 0$ *on this semilog scale.*

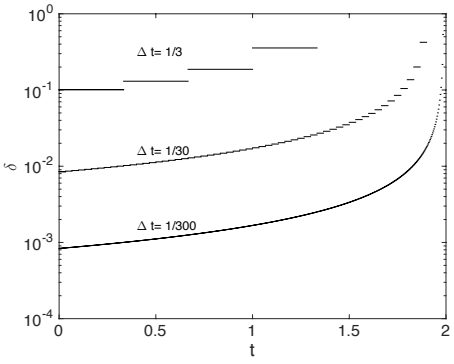


FIG. 4.2. *Optimal Backward Error for the explicit Euler method for Torricelli's Law,* $\dot{y} = -\sqrt{y}$, $y(0) = 1$. *We plot* $\delta_n = 2 \cdot (\sqrt{y_n} - \sqrt{y_{n+1}})/\Delta t - 1$ *for solutions using three different fixed time steps,* $\Delta t = 1/3, \Delta t = 1/30, \Delta t = 1/300$. *We see convergence away from the singularity when the bucket empties at* $t = 2$. *We also see the error get large near* $t = 2$. *Notice that for* $\Delta t = 1/3$ *the last two* δ_n *are blank because* $y_5 < 0$.

residual. Finally, figures 5.7 and 5 show that for this problem ode113 is very accurate indeed; even the supplied interpolant has only a tiny residual.

Remark: To do the computations we find it convenient to use the RefineMesh code from [4]. The code is included in the supplementary material.

In cases where the quality of the supplied interpolant is in doubt, computing the optimal backward error can supply useful reassurance.

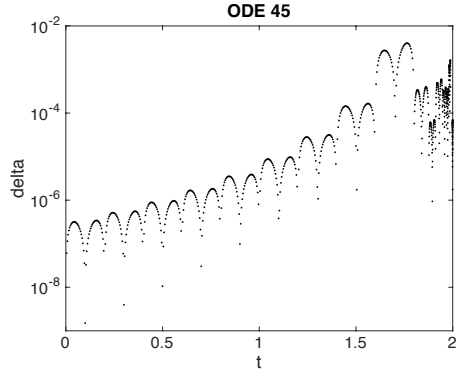


FIG. 5.1. *We used ode45 to solve* $\dot{y} = -\sqrt{y}, y(0) = 1$ *on* $0 \le t \le 2$ *using the default tolerances. Afterwards, we sampled the supplied interpolant and its derivative by uisng deval, and plotted the relative residual* $R(t) = (\dot{z} + \sqrt{z})/\sqrt{z}$ *on* $0 \le t \le 2$. *The code to do this is available in the supplementary material. We see that the residual is intially small, but grows large near the singularity at the end, indicating either a problem with the skeleton (given by a 4-5 Runge-Kutta pair) or with the supplied interpolant, or both.*

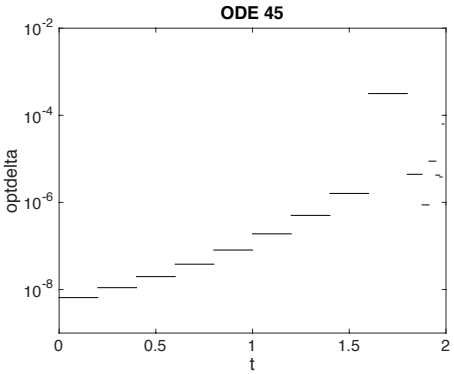


FIG. 5.2. *We took the skeleton from the ode45 solution as in 5.1, and computed the optimal backward error using equation* (4.14) *on each step. Notice that here the optimum is about 10 times smaller than the results in 5.1. This suggests that the supplied interpolant for ode45 is not as accuracte (indeed, it's only fourth order, but it suffices for many purposes and is quite convenient). The relative residual is never larger than* 10^{-3}, *which provides reassurance that the computed solution is valid.*

FIG. 5.3.

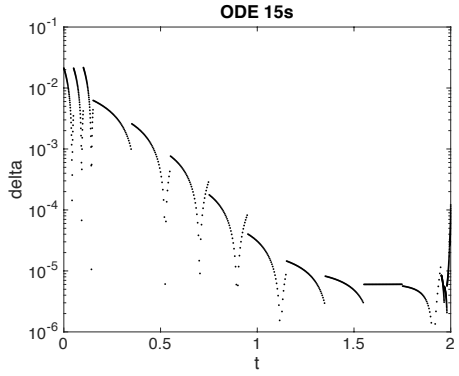


FIG. 5.4. *We used ode15s, which is suited to stiff problems, to solve* $\dot{y} = -\sqrt{y}(0) = 1$, *using its default tolerances. As for figure 5.1 and ode 45, we sampled the supplied interpolant and its derivative and computed the relative residual* $R(t) = (\dot{z} + \sqrt{z})/\sqrt{z}$. *We see here that the residual is initially large-ish, as is suitable for the default tolerances which are quite loose, but improves as* t *gets larger. This is evidence that the problem gets stiff as* z *gets small, because ode15s performs better the stiffer the problem is.*

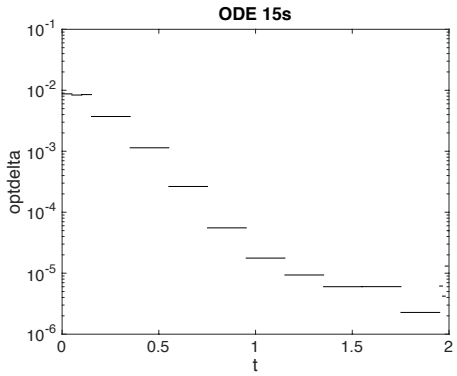


FIG. 5.5. *We took the skeleton for the ode15s solution and computed the optimal backward error using equation* (4.14) *on each step. Notice here that the relative residual of the supplied interpolant plotted in figure 5.4 is not much larger than the optimal, plotted here. This provides evidence that the supplied interpolant is high quality. That the backward error is small provides evidence of the accuracy of the solution.*

FIG. 5.6.

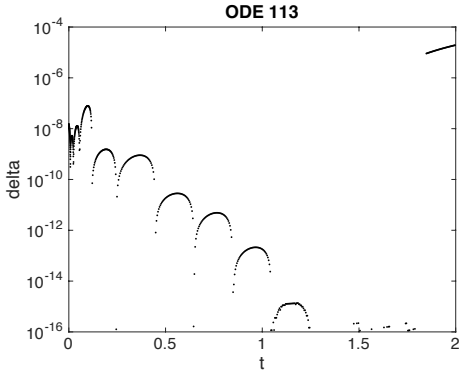


FIG. 5.7. *We used ode113 to solve* $\dot{y} = -\sqrt{y}, y(0) = 1$, *on* $0 \le t \le 2$ *using its default tolerances. Afterwards, we sampled the supplied interpolant and its derivative using* `deval`, *and plotted the relative residual* $R(t) = (\dot{z} + \sqrt{z})/\sqrt{z}$. *We see that ode113 is the most accurate method tried so far. We believe the extreme accuracy shown is an artifact of the simplicity of this problem: indeed a* 2^{nd} *order Taylor series method gets the reference solution exactly. The relative error on the last step is large, but this is only because the solution is so small.*

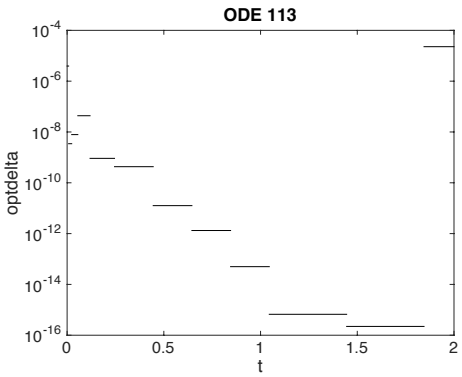


FIG. 5.8. *We took the skeleton from the ode113 solution and computed the optimal backward error using equation* (4.14) *on each step. The optimal backward error is hardly smaller than that from the supplied interpolants, showing that the supplied interpolants are high quality. The optimal error on the final step is again large, relatively, but this is only because the solution is so small.*

FIG. 5.9.

6. The Method of Modified Equations. The Method of Modified Equations has been studied at least since [10] and [18] . It is also discussed in [4], [9], and [11]. The idea is to start with the iteration defining the numerical method, say,

$$y_{n+1} = y_n - \Delta t\sqrt{y_n} \tag{6.1}$$

and conceptually interpolate with $z(t)$ so that $z(t_n) = y_n$ and $z(t_{n+1}) = y_{n+1}$. Now we try to find $z(t)$ so that the recurrence holds for all t with this fixed Δt. This converts the iteration to a functional equation:

$$z(t + \Delta t) = z(t) - \Delta t\sqrt{z(t)}\,. \tag{6.2}$$

We then use Taylor series, on the left hand side, to convert this, first, to a *singularly*-perturbed DE:

$$z(t) + \dot{z}(t)\Delta t + \frac{1}{2}\ddot{z}(t)\Delta t^2 + \cdots \tag{6.3}$$

$$= z(t) - \Delta t\sqrt{z(t)}$$

and then truncate and use repeated differentiation and substitution to convert this to a *regularly*- perturbed differential equation. Just briefly, subtracting $z(t)$ from each side and dividing by Δt, we get

$$\dot{z}(t) + \frac{1}{2}\Delta t\ \ddot{z}(t) + \frac{1}{6}(\Delta t)^2\ \dddot{z}(t) + \mathcal{O}(\Delta t^3) \tag{6.4}$$

$$= -\sqrt{z(t)}\,.$$

Differentiating twice we get expressions for $\ddot{z}$ and $\dddot{z}$ that can be combined and manipulated (see supplementary material) to get

$$\Big(-\frac{1}{\sqrt{z}} + \frac{\Delta t}{4z} + \mathcal{O}(\Delta t^3)\Big)\frac{dz}{dt} = 1 \tag{6.5}$$

which has solution, expressed implicitly as an equation for the unknown $z(t)$, up to $\mathcal{O}(\Delta t^3)$

$$2\sqrt{y_n} - 2\sqrt{z(t)} + \frac{\Delta t}{4}\ln\left(\frac{z(t)}{y_n}\right) = t - t_n\,. \tag{6.6}$$

Because $y_{n+1} = y_n - \Delta t\sqrt{y_{n+1}}$ can be written as $y_n = y_{n+1} + \Delta t\sqrt{y_{n+1}}$ we see that the modified functional equation for Euler Backward is

$$y(t - \Delta t) = y(t) - (-\Delta t)\sqrt{y(t)}\,, \tag{6.7}$$

which is the same equation as for Forward Euler but with reversed sign of Δt. Therefore one modified equation for Euler Backward is:

$$\left(-\frac{1}{\sqrt{z}} - \frac{\Delta t}{4z} + \mathcal{O}(\Delta t^3)\right)\frac{dz}{dt} = 1\,, \tag{6.8}$$

so

$$2\sqrt{y_n} - 2\sqrt{z(t)} - \frac{\Delta t}{4}\ln\left(\frac{z(t)}{y_n}\right) = t - t_n\,. \tag{6.9}$$

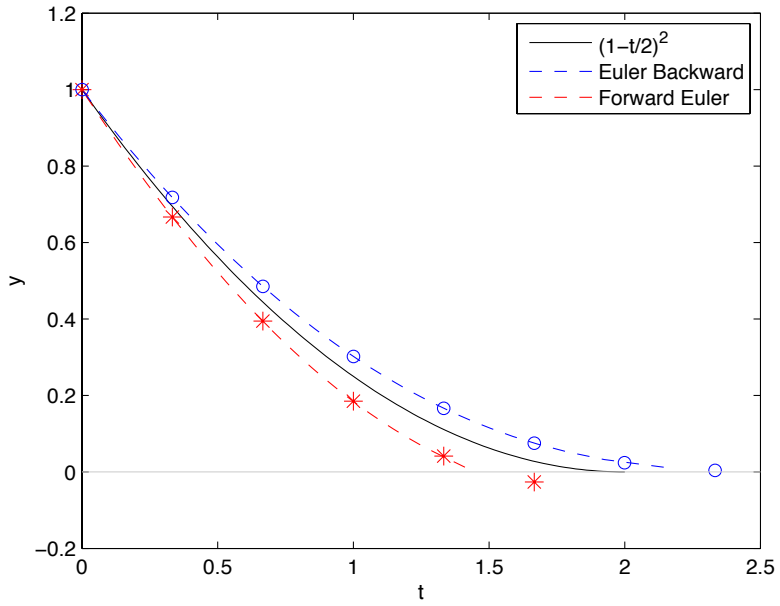


FIG. 6.1. *Modified Equations explaining Forward and Backward Euler for* $\dot{y} = -\sqrt{y}$. *The skeletons of the numerical solutions are shown with symbols o and *. They are nearly interpolated by the solutions of* (6.11) *and* (6.10). $\Delta t = 1/3$.

.

We remark that modified equations are not unique. The difference in sign in the logarithm term has quite profound implications. See figure 6.1. Notice that even for $\Delta t = 1/3$ these modified equations explain the numerical solution very well: it so happens that the $\mathcal{O}(\Delta t^3)$ term is $-\Delta t^3/192z^2$ leading to $(\Delta t^3)/192z$ on integration, which is very small so long as z is not near 0.

In other words, Forward Euler more nearly solves

$$\left(-\frac{1}{\sqrt{z}} + \frac{\Delta t}{4z} \right) \frac{dz}{dt} = 1 \tag{6.10}$$

while Euler Backward more nearly solves

$$\left(-\frac{1}{\sqrt{z}} - \frac{\Delta t}{4z} \right) \frac{dz}{dt} = 1 \, . \tag{6.11}$$

7. Returning to the Physics. Equation (1.2) set the loss of potential energy equal to the gain in kinetic energy, with no dissipation. What if we added a crude model of dissipation, say, replacing $\frac{1}{2}(\rho A \Delta h)v^2$ by $\frac{1}{2}(\rho A \Delta h)v^2 - \varepsilon(\rho A \Delta h)v$ where ε is a small parameter, where the mass is present, times velocity[8] as is common in friction terms? This model is hopelessly crude—friction and energy dissipation are not mere linear functions of momentum—but at least this model is simple and is something like models that have been used before. Then $\rho A \Delta h \cdot gh$ equal to this gives

$$gh = \frac{1}{2}v^2 - \varepsilon v \, . \tag{7.1}$$

This has roots

$$v = +\varepsilon \pm \sqrt{\varepsilon^2 + 2gh} \tag{7.2}$$

[8]Remember, $v < 0$. Thus the total change in potential energy $= 1/2mv^2 - \varepsilon mv$ means smaller magnitude velocity for a given potential difference, as is correct with dissipation present.

and again we discard the positive root because h is decreasing: thus

$$\frac{A}{a}\frac{dh}{d\tau} = +\frac{\varepsilon}{H} - \sqrt{\frac{2g}{H}}\sqrt{\left(\frac{\varepsilon^2}{2gh}\right) + y} \tag{7.3}$$

To clean things up put $\mu^2 = \frac{4\varepsilon^2}{2gH}$ or $\varepsilon = \sqrt{2gH}\mu$, so

$$\therefore \qquad \frac{A}{a}\frac{dy}{d\tau} = +\sqrt{\frac{2g}{H}} - \sqrt{\frac{2g}{H}}\sqrt{\mu^2 + y}\,. \tag{7.4}$$

Putting $t = \frac{a}{A}\sqrt{\frac{2g}{H}}\tau$ as before makes

$$\frac{dy}{dt} = -(-\mu + \sqrt{\mu^2 + y}) = -\frac{y}{\mu + \sqrt{\mu^2 + y}}\,, \tag{7.5}$$

or

$$-\left(\frac{\mu}{y} + \frac{\sqrt{\mu^2 + y}}{y}\right)\frac{dy}{dt} = 1\,. \tag{7.6}$$

Comparing this to the Euler Backward modified equation (6.11) we see these will agree to $\mathcal{O}(\Delta t^2)$ if the dissipation $\mu \doteq \Delta t/4$. This is, Euler Backward has introduced dissipation into the problem of about $\mu = \Delta t/4$. This corresponds to $\varepsilon = \sqrt{2gH}\ \cdot\ \Delta t/4$ in the original equation.

What of Forward Euler? Quite clearly its effect is qualitatively like negative dissipation—a kind of energy injection, if you will. The bucket might well empty faster if you shake it a bit!

This is the principal benefit of modified equations: they help to explain what numerical methods do, in terms that the modeller finds easier to understand.

8. Combining Optimal Backward Error and Modified Equations. One difficulty with modified equations is that they are truncated; another is that the series is not usually convergent, and is valid only in the asymptotic limit as $\Delta t \to 0$. Indeed taking the next few terms here gives

$$\left(-\frac{1}{\sqrt{z}} - \frac{\Delta t}{4z} + \frac{(\Delta t^3)}{192z^2} - \frac{(\Delta t^4)}{384z^{5/2}} + \frac{(\Delta t^5)}{1920z^3} + \mathcal{O}(\Delta t^6)\right)\frac{dz}{dt} = 1 \tag{8.1}$$

[One might be tempted to conjecture that all the subsequent terms are simple reciprocal integers, but computing one more term shows that they're not.] We can see that if $z = \mathcal{O}(\Delta t^\beta)$ for $\beta < 1$ then these terms decrease, but at $z = \mathcal{O}(\Delta t^2)$ the terms actually increase; so we need to "close the loop" somehow. So we see that once z gets so small as $\mathcal{O}(\Delta t^2)$, the modified equation is likely to be divergent if we take an infinite number of terms. We can argue that we should choose some fixed number of terms in a modified equation, e.g. (6.11),

$$\left(-\frac{1}{\sqrt{z}} - \frac{\Delta t}{4z}\right)\frac{dz}{dt} = 1 \tag{8.2}$$

and then ask for the optimal backward error $\hat{\delta}_n$ in *this* equation:

$$\left(-\frac{1}{\sqrt{z}} - \frac{\Delta t}{4z}\right)\frac{dz}{dt} = 1 + \hat{\delta}_n(t) \tag{8.3}$$

where $z(t_n) = y_n$ and $z(t_{n+1})$. Then this equation will completely explain the numerics. One can show that $\|\hat{\delta}_n\| = \mathcal{O}(\Delta t^3)$ as $\Delta t \to 0$. (By lucky chance the $\mathcal{O}(\Delta t^2)$ term is zero). But even if Δt is "large", we will have the best possible backward error. Integrating again,

$$\int_{\tau=t_n}^{t_{n+1}} \left(-\frac{1}{\sqrt{z(\tau)}} - \frac{\Delta t}{4z(\tau)} \right) \dot{z}(\tau) d\tau - \int_{\tau=t_n}^{t_{n+1}} d\tau = \int_{\tau=t_n}^{t_{n+1}} \hat{\delta}_n(\tau) d\tau \tag{8.4}$$

which constrains $\delta(\tau)$ as before, because by the triangle inequality

$$\left| 2\sqrt{y_n} - 2\sqrt{y_{n+1}} - \frac{\Delta t}{4} \ln\left(\frac{y_n}{y_{n+1}}\right) - \Delta t \right| = \left| \int_{\tau=t_n}^{t_{n+1}} \delta(\tau) d\tau \right| \le \int_{\tau=t_n}^{t_{n+1}} 1 d\tau \cdot \|\hat{\delta}_n\|_\infty \tag{8.5}$$

$$= \Delta t \|\hat{\delta}_n\|_\infty \tag{8.6}$$

so

$$\|\hat{\delta}_n\|_\infty \ge \left| \frac{2}{\Delta t}(\sqrt{y_n} - \sqrt{y_{n+1}}) - \frac{1}{4} \ln\left(\frac{y_n}{y_{n+1}}\right) - 1 \right| \tag{8.7}$$

(we always take $\Delta t > 0$ here). This lower bound is attained if $\hat{\delta}_n(\tau)$ is constant, so

$$\begin{aligned} \hat{\delta}_n &= \left[\int_{t_n}^{t_{n+1}} \left(-\frac{1}{\sqrt{z}} - \frac{\Delta t}{4z} \right) \frac{dz}{dt} d\tau - \Delta t \right] \cdot \frac{1}{\Delta t} \\ &= \frac{2\sqrt{y_n} - 2\sqrt{y_{n+1}}}{\Delta t} - \frac{1}{4} \ln\left(\frac{y_{n+1}}{y_n}\right) - 1 \end{aligned} \tag{8.8}$$

which is very similar to equation (2.1). We see that $z(t)$ *exactly* solves

$$\left(-\frac{1}{\sqrt{z}} - \frac{\Delta t}{4z} \right) \frac{dz}{dt} = 1 + \hat{\delta}_n \tag{8.9}$$

on $t_n \le t \le t_{n+1}$. We also see that the optimal backward error here should be considerably smaller than if the term $\Delta t \ln\left(y_{n+1}/y_n\right)/4$ were not included. We also see that we need not worry about convergence: we have a complete model of the error that Euler Backward introduces. We see finally in figures 8.1 and 8.2 that it also gets (relatively) large as $z \to 0$, which is still not surprising.

We personally find this combined method of assessment of validity of numerical solutions very satisfactory. Modified equations by themselves make some people uneasy because of the asymptotics. Using optimal backward error as well closes off that uneasiness.

9. Concluding Remarks. The main purpose of this paper was to use an understandable physical setting to explore the interpretation of numerical errors as modelling errors. The idea is surprisingly easy to get across to students, and it has happened more than once that a smiling student has asked at an exam "Is it ok if I give the exact answer to a slightly different question?"

Of course things get more complicated with larger models, and an especially serious issue is whether or not the $r(t)$ or $\delta(t)$ are physically reasonable perturbations; in other words, does the numerical method respect the structure of the model equations? This leads naturally to symplectic integration and other geometric integration methods [9], [11].

A relatively minor issue is the use of adaptive stepsize control. We originally had a section on this for this paper but cut it for space reasons. Using optimal backward error is straightforward in that context, and modified equations can be made to work, too (see [10]).

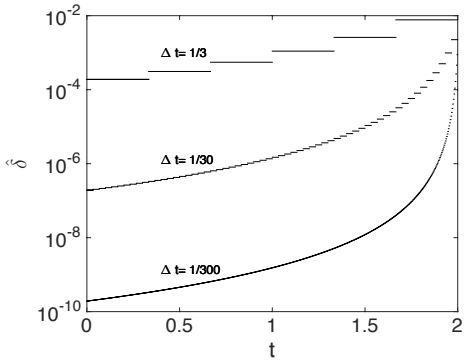


FIG. 8.1. *Optimal backward error for the modified equation* (6.10) *that explains the bulk of the numerical effects of the implicit Euler method applied to Torricelli's Law (3.4). Even for* $\Delta t = 1/3$ *we see that the residual numerical error is between* 10^{-4} *and* 10^{-2} *, showing that the bulk of the backward error shown in 4.1 is explained by the term* $-\frac{\Delta t}{4z} \cdot \frac{dz}{dt}$ *, incidentally illustrating that the implicit Euler method is first order.*

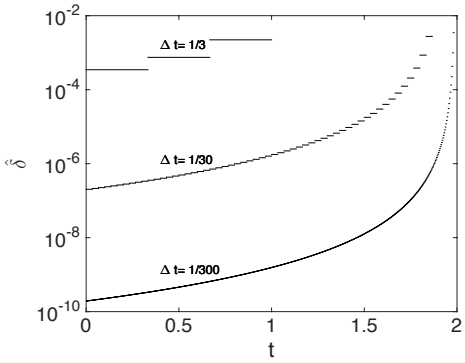


FIG. 8.2. *Optimal backward error for the modified equation* (6.7) *that explains the bulk of the numerical effects of the forward Euler method applied to Torricelli's law (2.2). Even for* $\Delta t = 1/3$ *we see that the residual numerical error is much smaller than that in 4.2, showing that the bulk of the backward error there is explained by the term* $\frac{\Delta t}{4z} \cdot \frac{dz}{dt}$ *. Incidentally this illustrates that Euler's method has first order.*

We have here shown two ways to interpret what Euler's method does to this equation. The ways work for other numerical methods, too: the optimal backward error is still given by equation (4.14), and modified equations can be constructed for any one-step method. We remark that some, but not all, second order methods (second order Taylor series, implicit trapezoidal rule and implicit midpoint rule but not RK2) actually give $r(t) \equiv 0$. That is, the numerical method gives the exact reference solution for Torricelli's Law.

The Torricelli's law model, $\dot{y} = -\sqrt{y}$, is only one of many possible models of a leaky bucket. Another more complicated model is introduced in [2]. We critique that paper in [5].

In addition to the archive of supplementary material, our scripts and Maple worksheets can be found at `www.apmaths.uwo.ca/~rcorless/Bucket`.

Acknowledgements. J.M. Sanz-Serna pointed out that Euler knew that the explicit Euler method could fail and that Euler had a method for dealing with the kinds of singularity encountered here. His method amounts to using the reference solution for this example. Nic Fillion helped with the preparation of the figures and some of the text. This work was supported by NSERC and the Faculty of Science, UWO. Samantha Brennan's puppy, Cheddar, thought an early draft of the paper was chewy, palatable, and fun.

REFERENCES

[1] GARRETT BIRKHOFF AND GIAN-CARLO ROTA, *Ordinary Differential Equations*, John Wiley & Sons, 1989.
[2] M. BLASONE, F. DELL'ANNO, R. DE LUCA, O. FAELLA, O. FIORE, AND A. SAGGESE, *Discharge time of a cylindrical leaking bucket*, Eur. J. Phys., 36 (2015), p. 035017.
[3] J.C. BUTCHER, *Numerical Methods for Ordinary Differential Equations*, Wiley, 1987.
[4] ROBERT CORLESS AND NICOLAS FILLION, *A Graduate Introduction to Numerical Methods: From the Viewpoint of Backward Error Analysis*, Springer, 2013.
[5] ROBERT CORLESS AND JULIA JANKOWSKI, *Revisiting the discharge time of a cylindrical leaking bucket*, In Preparation, (2016).
[6] ROBERT M. CORLESS AND YALÇIN KAYA, *Minimizing residuals in ode integration using optimal control theory*, In Preparation.
[7] ROBERT M. CORLESS, YALÇIN KAYA, AND ROBERT H.C. MOIR, *Optimal backward error and the Dahlquist test problem*, In Preparation.
[8] WAYNE H. ENRIGHT AND WAYNE B. HAYES, *Robust and reliable defect control for Runge-Kutta methods*, ACM Transactions on Mathematical Software (TOMS), 33 (2007), p. 1.
[9] DAVID GRIFFITHS AND DESMOND HIGHAM, *Numerical Methods for Ordinary Differential Equations*, Springer, 2010.
[10] D.F. GRIFFITHS AND J.M. SANZ-SERNA, *On the scope of the method of modified equations*, SIAM Journal on Scientific and Statistical Computing, 7 (1986), pp. 994–1008.

[11] Ernst Hairer, Christian Lubich, and Gerhard Wanner, *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations (Springer Series in Computational Mathematics)*, Springer, 2006.
[12] E. Hairer, S.P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems*, Springer, 1993.
[13] John H. Hubbard and Beverly H. West, *Differential Equations: A Dynamical Systems Approach: Ordinary Differential Equations (Texts in Applied Mathematics) (Pt 1)*, Springer, 1997.
[14] Evelyne Hubert, *Etude algébrique et algorithmique des singularités des équations différentielles implicites.*, PhD thesis, Institut National Polytechnique de Grenoble, 4 1997.
[15] Lawrence F. Shampine and Mark W. Reichelt, *The matlab ode suite*, SIAM J. Sci. Comput., 18 (1997), pp. 1–22.
[16] Gustaf Söderlind, Laurent Jay, and Manuel Calvo, *Stiffness 1952–2012: Sixty years in search of a definition*, BIT Numerical Mathematics, 55 (2014), pp. 531–558.
[17] Lloyd N. Trefethen, *Iv. 21 Numerical analysis*, The Princeton Companion to Mathematics, (2010), pp. 604 – 615.
[18] R.F. Warming and B.J. Hyett, *The modified equation approach to the stability and accuracy analysis of finite-difference methods*.