Supplementary Note 9.3

What Makes a Differential Equation Stiff?

When stability, not accuracy, determines the time step

Chapter 9 of the textbook

---

1 Why can a smooth solution be difficult to calculate?

Suppose we solve a differential equation numerically and plot the result. The curve is smooth, and it changes only slowly. It seems reasonable to use a fairly large time step. Yet an explicit method such as Euler may become unstable unless we take thousands of very small steps. Why?

This is the puzzle behind *stiff differential equations*. Chapter 9 introduces stiffness and explains that an explicit method can be limited by stability even when accuracy would permit a much larger time step. Here we will work through one example carefully. We will know its exact solution, so we can see what happens without guessing whether a numerical result is correct.

The central point is worth keeping in mind: *a small time step may be needed not because the solution changes rapidly, but because an explicit numerical method reacts badly to a rapidly decaying part of the equation.*

2 An equation with two very different time scales

Consider the initial value problem

$$\boxed{\frac{dx}{dt} = -1000(x - \cos t) - \sin t, \qquad x(0) = 2.} \tag{1}$$

At first sight the right-hand side looks rather artificial. We chose it for a reason: it has a simple exact solution that separates fast and slow motion.

Let us check that

$$\boxed{x(t) = \cos t + e^{-1000t}} \tag{2}$$

solves the equation. Its derivative is

$$x'(t) = -\sin t - 1000e^{-1000t}.$$

Also, $x(t) - \cos t = e^{-1000t}$, so the right-hand side of Eq. (1) is precisely $-1000e^{-1000t} - \sin t$. At $t = 0$ the proposed solution gives $x(0) = 1 + 1 = 2$, as required.

There are two parts to the solution:

- The term e^{-1000t} changes on a short time scale $1/1000 = 0.001$. By $t = 0.01$ its value is only $e^{-10} \approx 4.5 \times 10^{-5}$.
- The term $\cos t$ varies much more slowly, with period $2\pi \approx 6.28$.

The rapidly decaying term is called a *transient*. After a brief initial interval, the numerical solution should stay close to $\cos t$. One might expect a step such as $h = 0.05$ to be perfectly reasonable for following that slow motion.

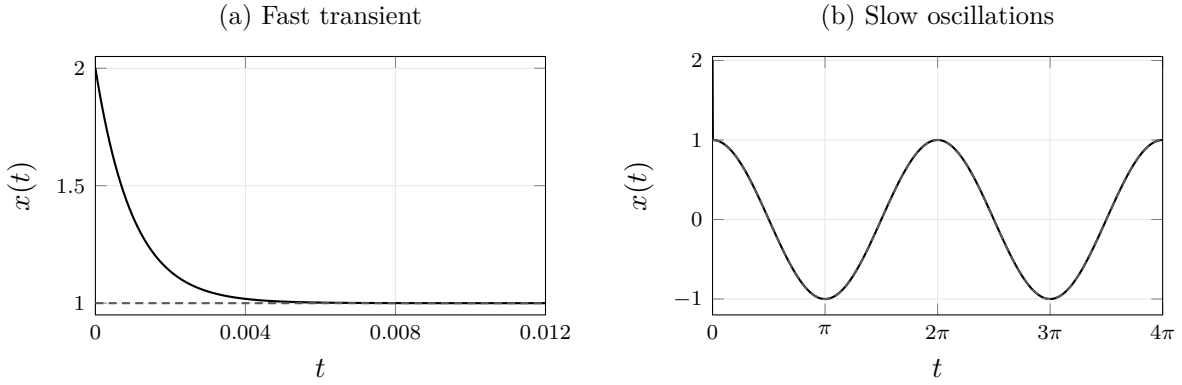


Figure 1: Two views of the exact solution $x(t) = \cos t + e^{-1000t}$ (solid), compared with its slowly varying part $\cos t$ (dashed). On the left, the fast transient disappears in a few thousandths of a time unit. On the right, the solution follows the slow oscillation over two full periods. Notice the very different time scales.

3 What happens with explicit Euler?

For an equation $x' = f(t, x)$, forward Euler uses

$$x_{n+1} = x_n + h f(t_n, x_n), \qquad t_n = nh.$$

Applying it to Eq. (1) gives

$$x_{n+1} = x_n + h\left[-1000(x_n - \cos t_n) - \sin t_n\right]. \tag{3}$$

Before calculating a long table of values, let us ask a more useful question: if two Euler calculations start with slightly different values, does the difference become smaller or larger after a step?

Suppose the two values at step n differ by a small amount δ_n. Since the terms involving $\cos t_n$ and $\sin t_n$ are the same in both calculations, subtracting their Euler updates gives

$$\delta_{n+1} = (1 - 1000h)\delta_n.$$

Repeating this relation, we obtain

$$\boxed{\delta_n = (1 - 1000h)^n \delta_0.} \tag{4}$$

The number $1 - 1000h$ tells us what happens to a small error or disturbance at each step. If its absolute value is less than one, that disturbance decreases. If it is greater than one, the disturbance grows.

For stability we therefore require

$$|1 - 1000h| < 1,$$
$$-1 < 1 - 1000h < 1.$$

For a positive time step this becomes

$$\boxed{0 < h < 0.002.} \tag{5}$$

This is the surprising restriction. The slow part of the solution varies on a time scale of order one, but explicit Euler cannot take a step as large as 0.002 without losing stability. A step of $h = 0.05$—quite modest for $\cos t$—is far beyond the limit.

3.1 What do the stability factors tell us?

We can learn a great deal by trying a few time steps:

Time step h	$1 - 1000h$	Effect on a small disturbance
0.0005	0.5	shrinks at every step
0.001	0	eliminated in one step
0.0015	-0.5	alternates sign, but shrinks
0.002	-1	alternates without shrinking
0.003	-2	alternates and doubles in magnitude

A negative multiplier is not automatically a problem; it simply reverses the sign of the disturbance. The real issue is whether its *magnitude* exceeds one.

There is an important distinction here. Equation (4) tracks how disturbances grow or decay under the numerical method. A numerical calculation also produces truncation errors at successive steps. When the multiplier has magnitude greater than one, those small errors are amplified, even if we started with an exact initial condition.

Stability versus accuracy. A method may be unstable with a time step that would otherwise approximate the slowly varying solution accurately. Stiffness is precisely the situation in which this stability restriction becomes the more severe one.

4 Why does the fast term matter after it has disappeared?

This is perhaps the most puzzling part. The transient e^{-1000t} is essentially gone after $t \approx 0.01$. Why must we continue using a tiny time step long afterward?

Look again at Eq. (1). If the numerical value differs slightly from the slowly varying solution $\cos t$, the term $-1000(x - \cos t)$ reacts strongly to that difference. In the exact differential equation, this reaction rapidly *removes* the difference. But an Euler step with h too large *overcorrects*. The next step overcorrects in the opposite direction, with an even larger magnitude.

Thus the large factor 1000 is still present in the differential equation, even when the exact transient is practically invisible. The numerical method must remain stable with respect to it.

This also explains why merely tightening the requested error tolerance is not always the right solution. A higher-accuracy explicit solver may have to reduce its steps dramatically to remain stable, even when the curve being calculated looks smooth.

5 Can we avoid the restriction? Try backward Euler

The book introduces *implicit methods* as the usual response to stiffness. Let us see how the simplest one works.

Forward Euler evaluates the derivative using the known values at time t_n. Backward Euler instead uses the derivative at the *new* time t_{n+1}:

$$x_{n+1} = x_n + hf(t_{n+1}, x_{n+1}). \tag{6}$$

The unknown x_{n+1} appears on both sides. That is why the method is called implicit. We must solve an equation for the new value.

For our example, write Eq. (1) as

$$x' = -1000x + g(t), \qquad g(t) = 1000\cos t - \sin t.$$

The backward Euler step becomes

$$\begin{aligned} x_{n+1} &= x_n + h[-1000x_{n+1} + g(t_{n+1})], \\ (1 + 1000h)x_{n+1} &= x_n + hg(t_{n+1}). \end{aligned}$$

Thus we can solve for the new value directly:

$$\boxed{x_{n+1} = \frac{x_n + h[1000\cos(t_{n+1}) - \sin(t_{n+1})]}{1 + 1000h}.} \tag{7}$$

Nothing especially difficult happened: for this linear equation, an implicit step requires only a division. For more complicated equations the corresponding algebraic equation may be nonlinear, and we generally use a numerical solver to handle it.

5.1 What happens to a small error?

Repeat our earlier experiment with two nearby starting values. Subtracting their backward Euler updates, we find

$$\delta_{n+1} = \frac{1}{1 + 1000h}\,\delta_n.$$

For *every* positive h,

$$\boxed{\left|\frac{1}{1 + 1000h}\right| < 1.} \tag{8}$$

So disturbances always decrease. For example, with $h = 0.05$, the factor is $1/51 \approx 0.0196$, whereas explicit Euler would have the factor -49.

This is why backward Euler can take much larger steps without becoming unstable. We call this property *unconditional stability* for decaying modes of this kind. It does *not* mean that an arbitrarily large step is accurate. We still need to choose h small enough to follow the slow motion and any features we want to resolve.

6 A numerical comparison

Let us put the time-step restriction into perspective. To integrate to $t = 1$, a forward Euler step of $h = 0.001$ requires 1000 steps. Backward Euler could use $h = 0.05$, requiring only 20 steps. Both can remain stable, although their accuracies will not necessarily be the same.

For this particular equation, the exact solution is known, so we can check the error. The following table compares the methods at $t = 1$; the numerical values use $x(0) = 2$ and the update equations derived above.

Method	Time step	Number of steps	$x(1)$
Exact solution	—	—	$\cos 1 + e^{-1000}$
Forward Euler	0.001	1000	`0.540303`
Backward Euler	0.05	20	`0.540288`

The final values may be close even though the methods require very different numbers of steps. This is not a claim that backward Euler is always faster: an implicit step may be more expensive, and backward Euler is only first-order accurate. The comparison simply illustrates that the *stability barrier* can matter much more than the smoothness of the solution.

7 Is every problem with two time scales stiff?

Not necessarily. Having fast and slow processes is a warning sign, but stiffness is really about what those time scales mean for numerical integration. If a fast oscillation remains physically important, we may need small steps to represent the oscillation accurately, regardless of the numerical method. An implicit method cannot supply information that our time step fails to resolve.

In our example, the fast process *decays* rapidly. Once it has disappeared, it no longer needs to be resolved accurately if we are interested only in the slow motion. Yet explicit Euler must still obey the short-time-scale stability restriction. This is the hallmark of stiffness.

The same issue occurs in systems of equations when some components or combinations of components decay much faster than others. Chemical reaction models, electric circuits, and models with rapid relaxation often lead to this behavior.

8 From backward Euler to Gear's methods

Backward Euler solves the stability problem in our example, but it has a limitation: it is only a first-order method. Can we obtain a more accurate method without returning to the small-step restriction of explicit Euler?

One approach is to approximate the derivative at the *new* time using several values of the solution. Such methods are called *backward differentiation formulas*, usually abbreviated BDF. They are closely associated with the work of C. William Gear, and the name *Gear's methods* is often used for this family of stiff integration algorithms.

8.1 Deriving a second-order backward formula

Let us see how this works. We want an approximation to $x'(t_{n+1})$, but instead of using only x_n and x_{n+1}, as in backward Euler, we will also use x_{n-1}. Assume for the moment that the time step h is constant.

Expand the two previous values about t_{n+1}:

$$x_n = x_{n+1} - hx'_{n+1} + \frac{h^2}{2}x''_{n+1} - \frac{h^3}{6}x'''_{n+1} + O(h^4),$$
$$x_{n-1} = x_{n+1} - 2hx'_{n+1} + 2h^2x''_{n+1} - \frac{4h^3}{3}x'''_{n+1} + O(h^4).$$

Here x'_{n+1} means the derivative of the *exact* solution at t_{n+1}; the prime does not denote a new time step.

If we take $3x_{n+1} - 4x_n + x_{n-1}$, the position terms cancel, as do the terms involving the second derivative. What remains is

$$3x_{n+1} - 4x_n + x_{n-1} = 2hx'_{n+1} + O(h^3).$$

Dividing by $2h$, we find

$$x'(t_{n+1}) \approx \frac{3x_{n+1} - 4x_n + x_{n-1}}{2h}.$$

The error in this derivative approximation is of order h^2, rather than order h as in the backward difference used by backward Euler.

For the differential equation $x' = f(t, x)$, we therefore obtain

$$\boxed{\frac{3x_{n+1} - 4x_n + x_{n-1}}{2h} = f(t_{n+1}, x_{n+1}).} \tag{9}$$

This is *BDF2*, the second-order backward differentiation formula. It uses the two previous solution values and evaluates the differential equation at the new time.

Just as with backward Euler, x_{n+1} appears on both sides, so we must solve for it. For our linear stiff example, $f(t, x) = -1000x + g(t)$, and the equation becomes

$$\begin{aligned} 3x_{n+1} - 4x_n + x_{n-1} &= 2h[-1000x_{n+1} + g(t_{n+1})], \\ (3 + 2000h)x_{n+1} &= 4x_n - x_{n-1} + 2hg(t_{n+1}). \end{aligned}$$

Hence

$$x_{n+1} = \frac{4x_n - x_{n-1} + 2h[1000\cos(t_{n+1}) - \sin(t_{n+1})]}{3 + 2000h}.$$

Once again, the implicit equation can be solved directly because this particular model is linear. A nonlinear equation would generally require an iterative procedure.

There is one practical detail worth noticing: BDF2 needs *two* previous solution values. If we are given only x_0, we must first calculate x_1, perhaps using backward Euler. After that, BDF2 can continue the integration.

8.2 Why does BDF2 help with stiffness?

BDF2 is not just a more accurate version of backward Euler. It also retains excellent stability properties for decaying modes. In fact, both backward Euler (BDF1) and BDF2 are *A-stable*: for the test equation $x' = \lambda x$, their numerical solutions are stable for any positive time step when $\operatorname{Re}\lambda < 0$.

This does not make the methods equally accurate at every step size. An overly large h can still miss important changes in the solution. The advantage is that stability no longer forces us to follow a fast decay that has already become unimportant to the motion we want to calculate.

8.3 What does the name Gear's method mean in practice?

We have derived BDF2 using a fixed time step. Gear's important contribution went beyond this one formula: practical stiff solvers can vary the time step and choose among backward differentiation formulas of different orders. If the solution is changing rapidly, the solver takes

smaller steps. If it becomes smooth, it may increase the step size or the order to work more efficiently.

With unequal time steps, the coefficients in Eq. (9) must be modified; we cannot simply use the same fixed-step formula. The solver must also estimate errors and solve the implicit algebraic equations, often with Newton-type iterations. These are significant algorithmic tasks, but numerical libraries take care of them for us.

Modern stiff solvers commonly use variable-order BDF methods, often with orders from one to five. Higher-order BDF methods do not all possess the full A-stability property of BDF1 and BDF2, but they can still be very effective for stiff problems. In practice, a method labeled `BDF` in a numerical library is often part of this broader tradition of Gear-type integration, rather than a single fixed formula.

The important idea is simple: *backward differentiation provides a way to combine higher accuracy with the stability properties needed for stiff equations.*

9 What should we do in practice?

When an explicit adaptive solver keeps choosing unexpectedly small steps or repeatedly rejects steps, ask whether a rapidly decaying mode might be restricting stability. The solution may look smooth, but the differential equation can still contain fast dynamics.

A common remedy is to switch to a solver designed for stiff problems. Backward Euler is the simplest example. As we have just seen, backward differentiation formulas (BDF methods) extend this idea to higher accuracy; modern solvers also offer other implicit integrators. These methods generally require solving algebraic equations at each step, so there is a trade-off between stability and the cost of a step.

> **The main point.** *Stiffness is a mismatch between the time step needed for accuracy and the much smaller time step required for stability by an explicit method.* A rapidly decaying part of the equation can impose that restriction long after it has become invisible in the solution. Implicit methods can remove this particular stability barrier, but accuracy still matters.

10 Try it yourself

Use Eq. (1) with $x(0) = 2$. Implement forward Euler and backward Euler and compare them with the exact solution, Eq. (2). Try $h = 0.001$, 0.0015, 0.002, and 0.003 for forward Euler. Which calculations remain stable? Then try $h = 0.05$ and 0.1 for backward Euler. At which step sizes does its accuracy become noticeably worse, even though the method remains stable?

As a further experiment, replace 1000 everywhere in the model by a parameter $\alpha > 0$. Derive the forward Euler stability condition in terms of α, and ask how the number of required steps changes as α increases.