Chapter 5. More efficient methods of numerical solution

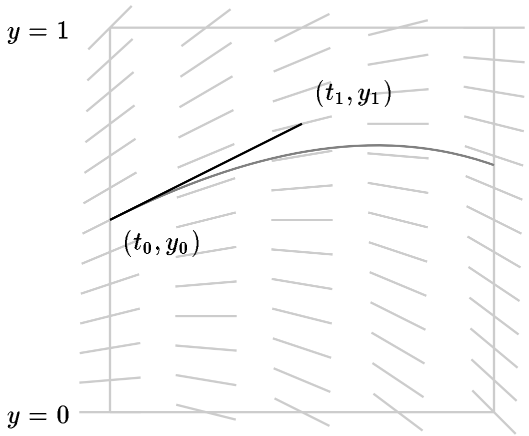
Euler's method is quite inefficient. Because the error is essentially proportional to the step size, if we want to halve our error we must double the number of steps taken across the same interval. So if N steps give us one significant figure, then $10\,N$ steps give us 2 figures, $100\,N$ steps give us 3, $100000\,N$ steps give us 6, and $10^{n-1}\,N$ steps give us n figures. Not too good, because 9 decimal accuracy is not unreasonable, and even a fast computer will think a bit about doing several billion operations. The method can be improved dramatically by a simple modification.

1. The improved Euler's method

Look again at what goes on in one step of Euler's method for solving

$$y' = f(t, y) \;.$$

Say we are at step n. We know t_n and y_n and want to calculate t_{n+1}, y_{n+1}. In Euler's method we assuming that the slope across the interval $[t_n, t_n + \Delta t] = [t_n, t_{n+1}]$ is what it is at t_n. Of course it is not true, but it is very roughly OK.



After we have taken one step of Euler's method, we can tell to what extent the assumption on the slope was false, by comparing the slope of the segment coming from the left at (t_{n+1}, y_{n+1}) with the value $f(t_{n+1}, y_{n+1})$ that it ought now to be. The new method will use this idea to get a better estimate for the slope across the interval $[t_n, t_{n+1}]$.

One step of the new method goes like this: Start at t_n with the estimate y_n. Do one step of Euler's method to get an approximate value

$$y_* = y_n + \Delta \cdot f(t_n, y_n)$$

for y_{n+1}. Calculate what the slope is at the point (t_{n+1}, y_*)—i.e. $f(t_{n+1}, y_*)$. We now make a modified guess as to what the slope across $[t_n, t_{n+1}]$ should have been by taking the average of these values $f(t_n, y_n)$ and $f(t_{n+1}, y_*)$. In other words

- In one step of the new method we calculate the sequence

$$\begin{aligned} s_0 &= f(x_n, y_n) \\ y_* &= y_n + \Delta \cdot s_0 \\ x_{n+1} &= x_n + \Delta \\ s_1 &= f(x_{n+1}, y_*) \\ y_{n+1} &= y_n + \Delta \cdot \left[\frac{s_0 + s_1}{2} \right] \end{aligned}$$

This is in contrast with one step of Euler's:

$$\begin{aligned} s_0 &= f(x_n, y_n) \\ y_{n+1} &= y_n + \Delta \cdot s_0 \\ x_{n+1} &= x_n + \Delta \end{aligned}$$

In the new method, we are doing essentially twice as much work in one step, but it should be a somewhat more accurate. It is, and dramatically so.

Let's see, for example, how the new method, which is called the **improved Euler's method**, does with the simple problem

$$y' = y - t, \quad y(0) = 0.5$$

we looked at before. Here is how one run of the new method goes in covering the interval $[0, 1]$ in 4 steps:

t	y	s_0	y_*	s_1
0.00	0.500000	0.500000	0.625000	0.375000
0.25	0.609375	0.359375	0.699219	0.199219
0.50	0.679199	0.179199	0.723999	-0.026001
0.75	0.698349	-0.051651	0.685436	-0.314564
1.00	0.652572			

And here is how the estimates for $y(1)$ compare for different step sizes:

N	estimated $y(1)$	true $y(1)$	error
2	0.679688	0.640859	0.038828
4	0.652572	$\ldots$	0.011713
8	0.644079	$\ldots$	0.003220
16	0.641703	$\ldots$	0.000844
32	0.641075	$\ldots$	0.000216
64	0.640914	$\ldots$	0.000055
128	0.640873	$\ldots$	0.000014
256	0.640863	$\ldots$	0.000003
512	0.640860	$\ldots$	0.000001
1024	0.640859	0.640859	0.000000

Halving the step size here cuts down the error by a factor of 4. This suggests:

- *The error in the improved Euler's method is proportional to* Δ^2.

The new method, which is called either the **improved Euler's** method or the **Runge-Kutta** method of order 2, is much more efficient than Euler's. If it takes N intervals to get 2 decimal accuracy, it will take $10\,N$ to get 4 decimals, $100\,N$ to get 6 (as opposed to $100000\,N$ in Euler's method). Very roughly, to get accuracy of ε it takes $1/\sqrt{\varepsilon}$ steps.

Exercise 1.1. *Do one step of the improved Euler's method for*

$$y' = y - t, \quad y(0) = 1$$

to estimate $y(0.1)$*. Then do two steps of size* $\Delta t = 0.05$*. Estimate the error in the second value.*

Exercise 1.2. *Do one step of the improved Euler's method for*

$$y' = y^2 + 1, \quad y(0) = 0$$

to estimate $y(0.1)$*. Then do two steps of size* $\Delta t = 0.05$*. How many steps would it take to get an estimate with* 6 *decimal accuracy?*

2. Relations with numerical calculation of integrals

If y is a solution of

$$y'(t) = f(t, y(t))$$

then for any interval $[t_n, t_{n+1}]$ with $t_{n+1} = t_n + \Delta t$ we can integrate to get

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

This does not amount to a formula for $y(t_{n+1})$ in terms of $y(t_n)$ because the right hand side involves the unknown function $y(t)$ throughout the interval $[t_n, t_{n+1}]$. In order to get from it an estimate for $y(t_{n+1})$ we must use some sort of approximation for the integral. In Euler's method we set

$$\int_{t_n}^{t_{n+1}} f(t, y(t))\, dt \doteq f(t_n, y(t_n))\, \Delta t \ .$$

We then replace $y(t_n)$ by its approximate value y_n to see how one step of Euler's method goes.

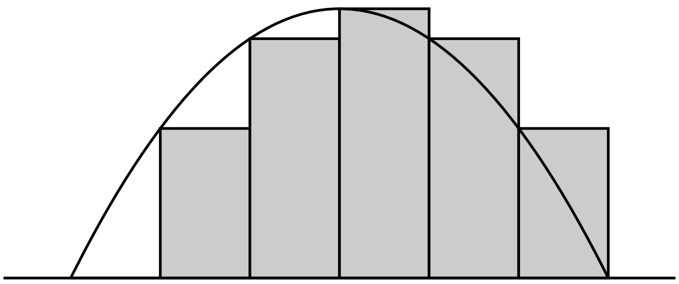
In the special case where $f(t, y) = f(t)$ does not depend on y, the solution of the differential equation & initial condition

$$y' = f(t), \quad y(t_0) = y_0$$

is given by a definite integral

$$y(t_n) = y_0 + \int_{t_0}^{t_n} f(t)\, dt$$

and Euler's method is equivalent to the rectangle rule for estimating the integral numerically. Recall that the rectangle rule approximates the graph of $f(t)$ by a sequence of horizontal lines, each one agreeing with the value of $f(t)$ at its left end.



In this special case it is simple to see geometrically why the overall error is roughly proportional to the step size.

In the improved Euler's method we approximate the integral

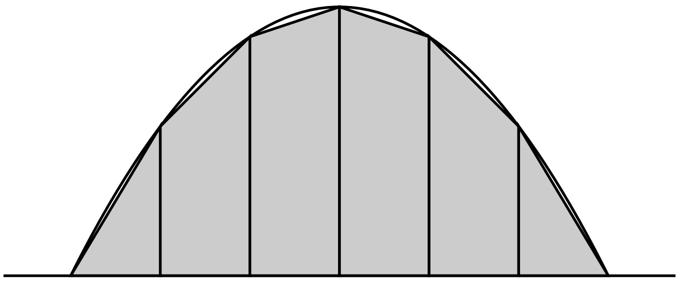
$$\int_{t_n}^{t_{n+1}} f(t, y(t))\, dt$$

by the trapezoid rule, which uses the estimate

$$\frac{\Delta t}{2}[f(t_n, y(t_n)) + f(t_{n+1}, y(t_{n+1}))]$$

for it. We don't know the exact values of $y(t_n)$ or $y(t_{n+1})$, but we approximate them by the values y_n and $y_n + \Delta t\, f(t_n, y_n)$.

In the special case $f(t, y) = f(t)$ the improved Euler's method reduces exactly to the trapezoid rule. Again, it is easy to visualize why the error over a fixed interval is proportional to the square of the step size.



3. Runge-Kutta

Neither of the two methods discussed so far is used much in practice. The simplest method which is in fact practical is one called **Runge-Kutta method of order** 4. It involves a more complicated sequence of calculations in each step. Let Δ be the step size. We calculate in succession:

$$\begin{aligned}
y_* &= y_n + \left[\frac{\Delta}{2}\right] s_* \\
x_* &= x + \left[\frac{\Delta}{2}\right] \\
s_{**} &= f(x_*, y_*) \\
y_{**} &= y_n + \left[\frac{\Delta}{2}\right] s_{**} \\
s_{***} &= f(x_*, y_{**}) \\
x_{n+1} &= x + \Delta \\
y_{***} &= y_n + \Delta \cdot s_{***} \\
s_{****} &= f(x_{n+1}, y_{***}) \\
y_{n+1} &= y_n + \Delta \cdot \left[\frac{s_* + 2s_{**} + 2s_{***} + s_{****}}{6}\right]
\end{aligned}$$

The purpose of using $*$ here is that the numbers like y_* etc. are not of interest beyond the immediate calculation they are involved in.

You wouldn't want to do this by hand, but again it is not too bad for a spread sheet. Here is a table of the calculations for

$$y' = y - t, \quad y(0) = 0.5$$

with 4 steps.

t	y	s_*	y_*	s_{**}	y_{**}	s_{***}	y_{***}	s_{****}
0.00	0.500000	0.500000	0.562500	0.437500	0.554688	0.429688	0.607422	0.357422
0.25	0.607992	0.357992	0.652740	0.277740	0.642709	0.267709	0.674919	0.174919
0.50	0.675650	0.175650	0.697607	0.072607	0.684726	0.059726	0.690582	−0.059418
0.75	0.691521	−0.058479	0.684211	−0.190789	0.667672	−0.207328	0.639689	−0.360311
1.00	0.640895							

And here is how the estimates for $y(1)$ compare for different step sizes. The method is so accurate that we must use nearly all the digits of accuracy possible:

N	estimated $y(1)$	true $y(1)$	error
2	0.64132690429688	0.64085908577048	0.00046781852640
4	0.64089503039934	...	0.00003594462886
8	0.64086157779163	...	0.00000249202116
16	0.64085924982971	...	0.00000016405923
32	0.64085909629440	...	0.00000001052393
64	0.64085908643684	...	0.00000000066636
128	0.64085908581240	...	0.00000000004192
256	0.64085908577311	...	0.00000000000263
512	0.64085908577064	...	0.00000000000016
1024	0.64085908577049	0.64085908577048	0.00000000000001

Here halving the step size reduces the error by a factor of 16. Since $16 = 2^4$, this suggests:

- *The error in RK4 is proportional to Δ^4.*

Euler's method and the improved Euler's method arise from using the rectangle rule and the trapezoid rule for estimating the integral

$$\int_{t_n}^{t_{n+1}} f(t, y(t))\, dt \; .$$

RK4 uses a variant of **Simpson's Rule** for calculating integrals and reduces to it in the special case when $f(t, y)$ doesn't depend on y.

4. Remarks on how to do the calculations

When implementing these by hand, which you might have to do every now and then, you should set the calculations out neatly in tables. Here is what it might look like for the improved Euler's method applied to the differential equation

$$y' = 2xy - 1$$

with step size $\Delta = 0.1$.

x	y	$s_* = f(x, y)$	$y_* = y + \Delta \cdot s_*$	$s_{**} = f(x + \Delta, y_*)$
0.000000	1.000000	-1.000000	0.900000	-0.820000
0.100000	0.909000	-0.818200	0.827180	-0.669128
0.200000	0.834634	-0.666147	0.768019	-0.539189

etc.

5. How to estimate the error

When you set out to approximate the solution of a differential equation

$$y' = f(x, y)$$

by a numerical method, you are given initial conditions $y(x_0) = y_0$ and a range of values $[x_0, x_f]$ (f for 'final') over which you want to approximate the solution. The particular numerical method you are going to use is usually specified in advance, and your major choice is then to choose the **step size**—the increment by which x changes in each step of the method.

The basic idea is that if you choose a small step size your approximation will be closer to the true solution, but at the cost of a larger number of steps. What is the exact relationship between step size and accuracy? We have discussed this already to some extent, but now we shall look at the question again. Some explicit data can give you a feel for how things go. Here is the outcome of several runs of Euler's method for solving the equation

$$y' = 2xy - 1, \quad y(0) = 1$$

over the interval $[0, 1]$.

Number of steps	Step size	Estimated value of $y(1)$
4	0.250000	0.426758
8	0.125000	0.540508
16	0.062500	0.608672
32	0.031250	0.646763
64	0.015625	0.667026
128	0.007812	0.677495
256	0.003906	0.682819
512	0.001953	0.685503
1024	0.000977	0.686851

This table suggests that if we keep making the step size smaller, then the approximations to $y(1)$ will converge to something, but does not give a good idea, for example, of how many steps we would need to take if we wanted an answer correct to 4 decimals. To get this we need some idea of how rapidly the estimates are converging. The way to do this most easily is to add a column to record the difference between an estimate and the previous one.

Number of steps	Step size	Estimated value of $y(1)$	Difference from previous estimate
4	0.250000	0.426758	0.426758
8	0.125000	0.540508	0.113751
16	0.062500	0.608672	0.068164
32	0.031250	0.646763	0.038091
64	0.015625	0.667026	0.020263
128	0.007812	0.677495	0.010469
256	0.003906	0.682819	0.005323
512	0.001953	0.685503	0.002685
1024	0.000977	0.686851	0.001348

The pattern in the last column is simple. Eventually, the difference gets cut in half at each line. This means that at least when h is small, the difference is roughly proportional to the step size. Thus we can **extrapolate** entries in the last column to be

$$0.000674, \quad 0.000337, \quad 0.000169, \quad 0.000084, \quad \ldots.$$

But the total error should be the sum of all these differences, which can be written as

$$0.001348\,(1/2 + 1/4 + 1/8 + 1/16 + \ldots) = 0.001348!$$

It is not in fact necessary to look at a whole sequence of runs. We can summarize the discussion more simply: *if we make two runs of Euler's method, the error in the second of the two estimates for $y(x_f)$ will be* (roughly) *the size of the difference between the two estimates.*

The reason the error can be estimated in this way is because *the error in Euler's method is proportional to the step size.* Suppose we have made two runs of the method, one with step size h and another with step size $h/2$. Then because of the proportionality, and the meaning of error, we know that

$$\begin{aligned}
\textit{true answer} &\doteq \textit{estimate}_{\,h} \quad\; + Ch \\
\textit{true answer} &\doteq \textit{estimate}_{\,h/2} + C(h/2)
\end{aligned}$$

for some constant C. Here *estimate*$_{\,h}$ means the estimate corresponding to step size h. If we treat the approximate equalities as equalities, we get two equations in the two unknowns *true answer* and C. We can solve them to get

$$\begin{aligned}
\textit{true answer} &\doteq 2\ \textit{estimate}_{\,h/2} - \textit{estimate}_{\,h} \\
&= \textit{estimate}_{\,h/2} + (\textit{estimate}_{\,h/2} - \textit{estimate}_{\,h}) \\
\textit{error in estimate}_{\,h/2} &\doteq \textit{estimate}_{\,h/2} - \textit{estimate}_{\,h}
\end{aligned}$$

In other words, *one run of Euler's method will tell us virtually nothing about how accurate the run is, but two runs will allow us to get an idea of the error and an improved estimate for the value of $y(x_f)$ as well.*

The same is true for any of the other methods, since we know that in each case the error is roughly proportional to some power of h—for improved Euler's it is h^2 and for $RK4$ it is h^4. In general, if the error is proportional to h^k and we make two runs of step size h and $h/2$ we get

$$\begin{aligned}
\textit{true answer} &\doteq \textit{estimate}_{\,h} + Ch^k \\
\textit{true answer} &\doteq \textit{estimate}_{\,h/2} + C(h/2)^k \\
2^k\ \textit{true answer} &\doteq 2^k\,\textit{estimate}_{\,h/2} + Ch^k
\end{aligned}$$

and subtract getting

$$\begin{aligned}
(2^k - 1)\ \textit{true answer} &\doteq 2^k\ \textit{estimate}_{\,h/2} - \textit{estimate}_{\,h} \\
\textit{true answer} &\doteq \frac{2^k\ \textit{estimate}_{\,h/2} - \textit{estimate}_{\,h}}{2^k - 1} \\
&= \textit{estimate}_{\,h/2} + \frac{\textit{estimate}_{\,h/2} - \textit{estimate}_{\,h}}{2^k - 1}
\end{aligned}$$

and finally:

- *If you do one run with step size h and another of size $h/2$ with a method of order k then*

$$\begin{aligned}
\textit{error in estimate}_{\,h/2} &\doteq \frac{\textit{estimate}_{\,h/2} - \textit{estimate}_{\,h}}{2^k - 1} \\
\textit{true answer} &\doteq \textit{estimate}_{\,h/2} + \textit{error in estimate}_{\,h/2}
\end{aligned}$$

The earlier formula is the special case of this with $k = 1$.

6. Step size choice

When we use one of the numerical methods to approximate the solution of the differential equation, we usually have ahead of time some idea of how accurate we want the approximation to be—to within 6 or 7 decimals of accuracy, say. The question is, how can we choose the step size in order to obtain this accuracy? The discussion above suggests the following procedure:

- *Make two runs of the method over the given range, with step size h and then with step size $h/2$.*

We get two estimates for $y(x_f)$, *estimate*$_h$ and *estimate*$_{h/2}$. If the method has error proportional to h^k (i.e. if the error is of order k) then we know that the error in *estimate*$_h$ is approximately

$$\textit{error in estimate}_h \doteq \frac{\textit{estimate}_h - \textit{estimate}_{h/2}}{1 - 2^{-k}} \,.$$

If we multiply the step size by α we multiply the error by α^k, so if we want an error of about ϵ:

- *Solve this equation to get the new step size h_ϵ:*

$$\left(\frac{h_\epsilon}{h}\right)^k = \left|\frac{\epsilon}{\textit{error in estimate}_h}\right|$$
$$h_\epsilon = h\left|\frac{\epsilon}{\textit{error in estimate}_h}\right|^{1/k}$$

The procedure looks tedious, but plausible: we choose some initial step size h which we judge (arbitrarily) to be reasonable. We do two runs at step sizes h and $h/2$ to see how accuracy depends on step size (to get the constant of proportionality). We then calculate the step size h needed to get the accuracy we want, and do a third run at the new size.

Example. Here is a typical question:

In solving the differential equation

$$y' = xy + 1, \quad y(0) = 1$$

by $RK4$, with step sizes 0.0625 and 0.03125, the estimates $3.05940\,72706\,92$ and $3.05940\,73971\,09$ for $y(1)$ were calculated. (a) Make an estimate of the error involved in these calculated values. (b) What step size should be chosen to obtain 16-decimal accuracy?

One thing to notice is that we don't need to know at all what the differential equation is, or even the exact method used. *The only important thing to know is the order of the method used*, which is $k = 4$.

We set

$$y_1 = 3.05940\,72706\,92$$
$$y_2 = 3.05940\,73971\,09$$

The formula for the error in the second value, say, is simple. We estimate it to be

$$\frac{y_2 - y_1}{2^k - 1} = \frac{0.00000\,01264\,17}{15} = 0.00000\,00084\,28$$

To figure out what step size to use to get a given accuracy, we must know how the error will depend on step size. It is

$$c\,\Delta^4$$

for some constant c, where Δ is the step size. In the second run the step size is 0.03125, so we estimate

$$c = \frac{0.00000\,00084\,28}{(0.03125)^4} = 0.008829$$

To get the step size we want we solve

$$c\,\Delta^4 = 0.008829\,\Delta^4 = 10^{-16}, \quad \Delta = \left(\frac{10^{-16}}{0.008829}\right)^{1/4} = \frac{10^{-4}}{0.3017} = 0.00033$$

Exercise 6.1. *Consider the initial value problem*

$$y' = 1 + (y-t)^2, \quad y(0) = 1/2\,.$$

Using the Improved Euler's method, we get the following estimates for $y(1)$ for different stepsizes Δt:

Δt	$y(1)$
0.04	4.09067160
0.02	$4.09758004\,.$

Approximately how many steps does it take to estimate $y(1)$ correctly to 6 decimals using the improved Euler method?

Using these data, what is the best estimate you can make for $y(1)$?

7. Summary of the numerical methods introduced so far

We have seen three different methods of solving differential equations by numerical approximation. In choosing among them there is a trade-off between simplicity and efficiency. Euler's method is relatively simple to understand and to program, for example, but almost hopelessly inefficient. The third and final method, the order 4 method of Runge-Kutta, is very efficient but rather difficult to understand and even to program. The improved Euler's method lies somewhere in between these two on both grounds.

Each one produces a sequence of approximations to the solution over a range of x-values, and proceeds from an approximation of y at one value of x to an approximation at the next in one more or less complicated step. In each method one step goes from x_n to $x_{n+1} = x_n + \Delta$. The methods differ in how the step from y_n to y_{n+1} is performed. More efficient single steps come about at the cost of higher complexity for one step. But for a given desired accuracy, the overall savings in time is good.

Method	The step from (x_n, y_n) to (x_{n+1}, y_{n+1})	Overall error
Euler's	$s_* = f(x_n, y_n)$	proportional to Δ
	$y_{n+1} = y_n + \Delta \cdot s_*$	
	$x_{n+1} = x_n + \Delta$	
improved Euler's	$s_* = f(x_n, y_n)$	proportional to Δ^2
(RK of order 2)	$y_* = y_n + \Delta \cdot s_*$	
	$x_{n+1} = x_n + \Delta$	
	$s_{**} = f(x_{n+1}, y_*)$	

$$y_{n+1} = y_n + \Delta \cdot \left[\frac{s_* + s_{**}}{2}\right]$$

Runge-Kutta of order 4 proportional to Δ^4

$$\begin{aligned}
s_* &= f(x_n, y_n) \\
y_* &= y_n + \left[\frac{\Delta}{2}\right] s_* \\
x_* &= x + \left[\frac{\Delta}{2}\right] \\
s_{**} &= f(x_*, y_*) \\
y_{**} &= y_n + \left[\frac{\Delta}{2}\right] s_{**} \\
s_{***} &= f(x_*, y_{**}) \\
x_{n+1} &= x + \Delta \\
y_{***} &= y_n + \Delta \cdot s_{***} \\
s_{****} &= f(x_{n+1}, y_{***}) \\
y_{n+1} &= y_n + \Delta \cdot \left[\frac{s_* + 2s_{**} + 2s_{***} + s_{****}}{6}\right]
\end{aligned}$$

In practice, of course, you cannot expect to do several steps of any of these methods except by computer. You should, however, understand • how to one step of each of them; • how to estimate errors in using them; • how to use error estimates to choose efficient step sizes.

Numerical methods are often in practice the *only* way to solve a differential equation. They can be of amazing accuracy, and in fact the methods we have described here are not too different from those used to send space vehicles on incredibly long voyages. But often, too, in practice you want some overall idea about how solutions of a differential equation behave, in addition to precise numerical values. There is no automatic way of obtaining this.

In all these methods, one of the main problems is to choose the right step size. The basic principle in doing this is that:

- *Two runs of any method will let you estimate the error in the calculation, as well as decide what step size to use in order to get the accuracy you require.*

8. Remarks on rounding error

Computers can only deal with numbers of limited accuracy, normally about 16 digits of precision. They cannot calculate even the product and sums of numbers exactly, much less calculate the exact values of functions like $\sin$. Errors in computations due to this limited accuracy are called **rounding errors**.

Rounding errors are not usually a problem in the methods we have seen, except in the circumstance where Euler's method is used to obtain extremely high accuracy, in which case it will require enough steps for serious error to build up. But then Euler's method should never be used for high accuracy anyway.

There is one circumstance, however, in which there will be a problem. Consider this differential equation:

$$y' = \cos t + y, \quad y(0) = y_0$$

The general solution is

$$y = (-\cos t + \sin t)/2 + (y_0 + 1/2)e^t$$

Then whenever $y_0 \neq 1/2$ the solution will grow exponentially. In the exceptional case $y_0 = 1/2$ it will simply oscillate. Here, however, is the graph of a good numerical aproximation matching initial condition $y(0) = 1/2$:

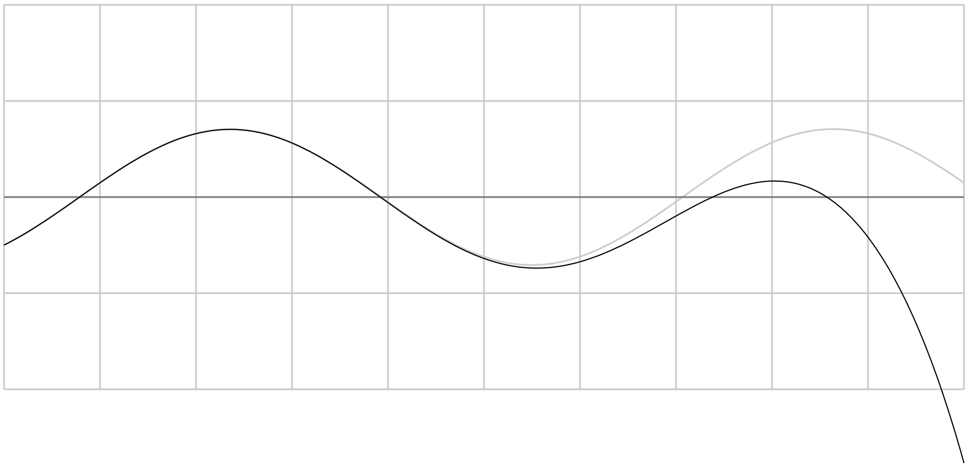


Figure 8.1. *How rounding errors affect the calculations for a stiff equation.*

Rounding errors are inevitably made at some point in almost any computer calculation, and as soon as that happens the estimated $y(x)$ will make an almost certainly irreversible move from the graph of the unique bounded solution to one of the nearby ones with exponential growth. From that point on it must grow unbounded, no matter how well it has looked so far. This is not a serious problem in the following sense: in the real world, we can *never* measure initial conditions exactly, so that we can never be sure to hit the one bounded solution exactly. Nor can nature. Since the bounded solution is **unstable** in the sense that nearby solution eventually diverge considerably from it, we should not expect to meet with it in physical problems.

- *Unstable solutions do not realistically occur in nature. Rounding errors simulate the approximate nature of our measurements.*

Another way to say this:

- *Because of rounding errors, the best we can hope to do with machine computations is to find the exact answer to a problem near to the one that is originally posed.*