Ordinary Differential Equations, part of Math 18.335, Fall 2004

Plamen Koev

December 8, 2004

1 Introduction

We study

$$\dot{y}(t) = f(y(t)), \quad y(0) = y_0,$$

where f is a nice differentiable, vector-valued function.

Consider the scalar case first. To solve this equation, discretize $t_i = ih$ for a given h and use

$$\frac{y(t+h) - y(t)}{h} \approx y' = f(y(t))$$

to obtain $y(t+h) \approx y(t) + hf(y(t))$, which yields the Euler's method (y_i approximates $y(t_i)$):

$$y_{n+1} = y_n + hf(y_n).$$

Truncation error (τ_n):

$$y(t_{n+1}) = y(t_n) + hf(y(t_n)) + h\tau_n,$$

where (Taylor)

$$\tau_n = \frac{h}{2} y''(t_n + \theta_n h).$$

Let

$$e_n = \text{error} = y_n - y(t_n) = \text{computed} - \text{exact}.$$

Then the error satisfies

$$e_{n+1} = e_n + h[f(y_n) - f(y(t_n))] - h\tau_n.$$

After some manipulation we get

$$|e_n| \le \frac{e^{LT} - 1}{L} \cdot \frac{h}{2} \cdot M,$$

where $T = nh$, L is a bound on f', and M is a bound on y''.

For systems the story is a little more complicated to derive, but the answer is the same

$$\|e_n\| \le \frac{e^{LT} - 1}{L} \cdot \frac{h}{2} \cdot M,$$

but we have to use norms instead ($1, 2, \infty$ norms are all OK).

Consider the error more carefully now. We will show that there exists a function $e(t)$ such that

$$y_n = y(t_n) + he(t_n) + O(h^2).$$

Consider two calculations—one with mesh size h and another with mesh size $h/2$. Write

$$\begin{aligned} \text{comp}_h y_n &= y(nh) + he(nh) + O(h^2) \\ \text{comp}_{h/2} y_{2n} &= y(2nh/2) + (h/2)e(2nh/2) + O(h^2) \end{aligned}$$

Subtract:

$$\text{comp}_h y_n - \text{comp}_{h/2} y_{2n} = (h - h/2)e(nh) + O(h^2).$$

So we can figure out what the error is. Alternatively we can *eliminate* the error

$$2 \cdot \text{comp}_h y_n - \text{comp}_{h/2} y_{2n} = (2-1)y(t_n) + O(h^2).$$

A combination between the computed values is a more accurate estimate of the exact solution than either calculation. We obtained a second order method, instead of Euler's first order method. What did we lose? We lost our ability to tell what the new error is. This is typical. If you know what the error is you can improve the calculation, but then you don't know what the error is.

To obtain information about the error we insert $y_n = y(t_n) + he(t_n)$ in Euler's method ($y_{n+1} = y_n + hf(y_n)$), use Taylor series and obtain:

$$y(t+h) + he(t+h) + \ldots = y(t) + he(t) + hf(y(t) + he(t) + \cdots) + \cdots$$

$$y + hy' + \frac{h^2}{2}y'' + \frac{h^3}{3!}y''' + \ldots + h\left(e + he' + \frac{h^2}{2}e'' + \ldots\right) = y + he + h\left(f(y) + f'(y)he + \frac{1}{2}f''(y)(he)^2 + \ldots\right)$$

Now use that $y' = f(y)$ to obtain

$$h^2\left(\frac{1}{2}y'' + e' - f'(y)e\right) = O(h^3).$$

Divide by h^2 and let $h \to 0$. It is natural to define

$$e'(t) = f'(y(t)) \cdot e(t) - \frac{1}{2}y'(t), \quad e(0) = 0.$$

This is called *the variational equation*. Now after some additional work we can prove that

$$\text{comp } y_n = \text{exact } y(t) + he(t) + O(h^2).$$

2 Runge-Kutta Methods

We begin with

$$\begin{aligned} q_n &= y_n + \frac{h}{2}f(y_n) \\ y_{n+1} &= y_n + hf(q_n), \end{aligned}$$

which is called *Heun's method* by some, *Runge's Second Order* by others or *Improved Euler* by still others. It is explicit because

$$y_{n+1} = y_n + hf(y_n + \frac{h}{2}f(y_n)),$$

so to compute y_{n+1} we only need y_n and f. What is the truncation error of the scheme?

$$y(t_{n+1} = y(t_n) + hf(y(t_n) + \frac{h}{2}f(y(t_n))) + h\tau_n$$

Using Taylor series we get

$$y(t) + hy'(t) + \frac{h^2}{2}y''(t) + \frac{h^3}{3!}y'''(t+\theta h) = y(t) + h\left(f(y(t)) + f'(y(t))\frac{h}{2}f(y) + \frac{1}{2}f''(y(t))\left(\frac{h}{2}f(\theta)\right)^2\right) + h\tau_n$$

Since $y' = f(y)$ we see that $y'' = f'(y)y' = f'(y) \cdot f(y)$ and we can cancel all terms up to h^2 to get

$$\tau_n = h^2\left(\frac{1}{3!}y'''(t+\theta h) - \frac{1}{8}f''(y(t) + \theta\frac{h}{2}f(y))(f(y))^2\right),$$

where the two θ's are not the same. Therefore $\tau_n = O(h^2)$.

Heun's method became very popular because it required very little storage (an important consideration at the time). We can forget all about $f(y_n)$ once we have q_n. This is not the case for the *Improved Euler* method

$$\begin{aligned} q_n &= y_n + f(y_n) \\ y_{n+1} &= y_n + \frac{h}{2}\left(f(y_n) + f(q_n)\right) \end{aligned}$$

We derive it as follows:

$$y(t+h) - y(t) = \int_t^{t+h} f(y(z))dz,$$

then using the trapezoidal rule we get

$$y(t+h) - y(t) = \frac{h}{2}(f(y(t)) + f(y(t+h))),$$

we don't have $y(t+h)$ available in order to get $f(y(t+h))$, but we can replace $f(y(t+h))$ by the result we get from Euler's method $f(y(t) + hf(y(t)))$. We can show that the truncation error for modified Euler

$$y_{n+1} = y_n + \frac{h}{2}\left(f(y_n) + f(y_n + hf(y_n))\right)$$

is

$$\tau_n = h^2\left(\frac{1}{3!}y'''(t+\theta h) - \frac{1}{4}f''(y(t) + \theta hf(y)) \cdot (f(y))^2\right)$$

3 Stiff ODEs

Consider

$$y' = -\lambda(y-1)$$

for large λ, e.g. $\lambda = 100$, or $\lambda = 1000$, or $\lambda = 10^6$. The solution is

$$y(t) = e^{-\lambda t}(y(0) - 1) + 1.$$

All solutions start off at $y(0)$ and "seek" $y \equiv 1$ and the change for small t is very rapid.

To be specific, take $\lambda = 1000$ and use Euler's method with $h = \frac{1}{100}$, $h = \frac{2}{1000}$ and $h = \frac{1}{10000}$.

$$u_{n+1} = u_n - h\lambda(u_n - 1) = u_n - h \cdot 1000(u_n - 1) = (1 - 1000h)u_n + 1000h.$$

Try $h = \frac{1}{100}$ with $u_0 = 1 + \varepsilon$. Then

$$\begin{aligned} u_1 &= 1 - 9\varepsilon \\ u_2 &= 1 + 9^2\varepsilon \\ u_3 &= 1 - 9^3\varepsilon, \end{aligned}$$

and in general $u_n = 1 + (-9)^n\varepsilon$. This is terrible with wild growth no matter how tiny ε is.

Try $h = \frac{2}{1000}$ with $u_0 = 1 + \varepsilon$. Then from the equation

$$u_{n+1} = -u_n + 2$$

and we have

$$\begin{aligned} u_1 &= 1 - \varepsilon \\ u_2 &= 1 + \varepsilon \\ u_3 &= 1 - \varepsilon, \end{aligned}$$

and in general $u_n = 1 + (-1)^n \varepsilon$. Our numerical solution still does not tend to the correct solution, but at least the wild oscillations are gone.

Try $h = \frac{1}{10000}$ with $u_0 = 1 + \varepsilon$. Then from the equation

$$u_{n+1} = -\frac{9}{10} u_n + \frac{1}{10}$$

and we have

$$\begin{aligned} u_1 &= 1 + \frac{9}{10}\varepsilon \\ u_2 &= 1 + \left(\frac{9}{10}\right)^2 \varepsilon \\ u_3 &= 1 + \left(\frac{9}{10}\right)^3 \varepsilon \end{aligned}$$

and in general $u_n = 1 + \left(\frac{9}{10}\right)^n \varepsilon$ and the numerical solution will finally converge to the correct solution $y \equiv 1$.

It seems fair to use small step size when the solution is changing rapidly, but it seems unfair to take small steps past $t = 1/10$. For example if $y(0) = 2$, then $y(t) = e^{-1000t} + 1$ and $e^{-1000/10} + 1 = e^{-100} + 1 = 1$ in double precision floating point arithmetic.

The problem has nothing to do with Euler's method. *All* explicit methods suffer from the same problem.

The choice of $y' = -\lambda(y-1)$ was somewhat arbitrary, we could have had the solution approaching *any* function by choosing to solve instead

$$y' = -\lambda(y - \phi(t)) + \phi'(t),$$

which has a solution

$$y(t) = \phi(t) + e^{-\lambda t}(y(0) - \phi(0)).$$

So let's look at the simplest case $\phi(t) \equiv 0$

$$y' = -\lambda y.$$

There is an interesting way out of this trouble, namely to use implicit methods. Implicit Euler:

$$u_{n+1} = u_n + hf(u_{n+1}).$$

The trouble, of course, is computing u_{n+1} from here, but we will worry about this later. Attack $y' = -\lambda y$:

$$\begin{aligned} u_{n+1} &= u_n + h(-\lambda)u_{n+1} \\ (1 + h\lambda)u_{n+1} &= u_n \\ u_{n+1} &= \frac{1}{1 + h\lambda} u_n. \end{aligned}$$

So $h = \frac{1}{10}, \frac{1}{100}, \frac{1}{1000}, \frac{1}{10000}$ gives

$$\begin{aligned} u_n &= \left(\frac{1}{1 + \frac{1}{10}1000}\right)^n u_0 = \left(\frac{1}{101}\right)^n u_0 \to 0 \text{ rapidly} \\ u_n &= \left(\frac{1}{1 + \frac{1}{100}1000}\right)^n u_0 = \left(\frac{1}{11}\right)^n u_0 \to 0 \text{ rapidly} \\ u_n &= \left(\frac{1}{1 + \frac{1}{1000}1000}\right)^n u_0 = \left(\frac{1}{2}\right)^n u_0 \to 0 \text{ sort of} \\ u_n &= \left(\frac{1}{1 + \frac{1}{10000}1000}\right)^n u_0 = \left(\frac{10}{11}\right)^n u_0 \to 0 \text{ pretty slowly.} \end{aligned}$$

Only the last case follows the accurate solution accurately. For $h = \frac{1}{10000}$

$$\begin{aligned} y(t_n) &= e^{-\lambda t} y(0) = e^{-(1/10)n} y(0) \\ u_n &= e^{n \ln(10/11)} u_0 = e^{-n \ln(1+1/10)} u_0 \sim e^{-n\left(\frac{1}{10} - \frac{\left(\frac{1}{10}\right)^2}{2} + \frac{\left(\frac{1}{10}\right)^3}{3} + \dots\right)} u_0 \end{aligned}$$

Does the implicit Euler's method converge?

$$y(t_{n+1}) = y(t_n) + hf(y(t_{n+1})) + h\tau_n.$$

$$\begin{aligned} y + hy' + \frac{h^2}{2} y''(t + \theta h) &= y + hf(y + hy'(t + \theta h)) + h\tau \\ &= y + h\left(f(y) + f'(y + \theta h y'(t + \theta h) \cdot h \cdot y'(t + \theta h)\right) + h\tau \end{aligned}$$

Therefore

$$\tau_n = h\left(\frac{1}{2} y''(t + \theta h) - f'(y + \theta h y'(t + \theta h)) \cdot y'(t + \theta h)\right)$$

and

$$|\tau_n| \le h \left| -\frac{1}{2} y''(t) + O(h) \right| \le \frac{h}{2} C$$

Then we can repeat the proof for the explicit Euler:

$$\begin{aligned} y_{n+1} &= y_n + hf(y_{n+1}) \\ y(t_{n+1}) &= y(t_n) + hf(y_(t_{n+1})) + h\tau_n \\ y_{n+1} - y(t_{n+1}) &= y_n - y(t_n) + h\left(f(y_{n+1}) - f(y_(t_{n+1})\right) - h\tau_n \\ |e_{n+1}| &\le |e_n| + hL|e_{n+1}| + h|\tau_n| \\ |e_{n+1}| &\le (1 - hL)^{-1}|e_n| + h|\tau_n| \end{aligned}$$

Assume $hL < \frac{1}{2}$. Then $(1 - hL)^{-1} \le 1 + 2hL$ and

$$\begin{aligned} |e_n| &\le (1 + 2hL)|e_{n-1}| + h^2 C \\ &\le \dots \\ &\le (1 + 2Lh)^n |e_0| + \left(\sum_{i=0}^{n-1} (1 + 2Lh)^i\right) h^2 C \\ &= \frac{(1 + 2Lh)^n - 1}{1 + 2Lh - 1} h^2 C \\ &\le \frac{e^{2LT} - 1}{2L} hC. \end{aligned}$$