Jim Lambers
MAT 461/561
Spring Semester 2009-10
Lecture 9 Notes

These notes correspond to Section 5.11 in the text.

Stiff Differential Equations

To this point, we have evaluated the accuracy of numerical methods for initial-value problems in terms of the rate at which the error approaches zero, when the step size h approaches zero. However, this characterization of accuracy is not always informative, because it neglects the fact that the local truncation error of any one-step or multistep method also depends on higher-order derivatives of the solution. In some cases, these derivatives can be quite large in magnitude, even when the solution itself is relatively small, which requires that h be chosen particularly small in order to achieve even reasonable accuracy.

This leads to the concept of a *stiff* differential equation. A differential equation of the form $y' = f(t, y)$ is said to be *stiff* if its exact solution $y(t)$ includes a term that decays exponentially to zero as t increases, but whose derivatives are much greater in magnitude than the term itself. An example of such a term is e^{-ct}, where c is a large, positive constant, because its kth derivative is $c^k e^{-ct}$. Because of the factor of c^k, this derivative decays to zero much more slowly than e^{-ct} as t increases. Because the error includes a term of this form, evaluated at a time less than t, the error can be quite large if h is not chosen sufficiently small to offset this large derivative. Furthermore, the larger c is, the smaller h must be to maintain accuracy.

Example Consider the initial value problem

$$y' = -100y, \quad t > 0, \quad y(0) = 1.$$

The exact solution is $y(t) = e^{-100t}$, which rapidly decays to zero as t increases. If we solve this problem using Euler's method, with step size $h = 0.1$, then we have

$$y_{n+1} = y_n - 100hy_n = -9y_n,$$

which yields the exponentially *growing* solution $y_n = (-9)^n$. On the other hand, if we choose $h = 10^{-3}$, we obtain the computed solution $y_n = (0.9)^n$, which is much more accurate, and correctly captures the qualitative behavior of the exact solution, in that it rapidly decays to zero. □

The ODE in the preceding example is a special case of the *test equation*

$$y' = \lambda y, \quad y(0) = 1, \quad \operatorname{Re}\lambda < 0.$$

The exact solution to this problem is $y(t) = e^{\lambda t}$. However, as λ increases in magnitude, the problem becomes increasingly stiff. By applying a numerical method to this problem, we can determine how small h must be, for a given value of λ, in order to obtain a qualitatively accurate solution.

When applying a one-step method to the test equation, the computed solution has the form

$$y_{n+1} = Q(h\lambda)y_n,$$

where $Q(h\lambda)$ is a polynomial in $h\lambda$. This polynomial is meant to approximate $e^{h\lambda}$, since the exact solution satisfies $y(t_{n+1}) = e^{h\lambda}y(t_n)$. However, to obtain a qualitatively correct solution, that decays to zero as t increases, we must choose h so that $|Q(h\lambda)| < 1$.

Example Consider the modified Euler method

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

Setting $f(t, y) = -\lambda y$ yields the computed solution

$$y_{n+1} = y_n + \frac{h}{2}[\lambda y_n + \lambda(y_n + h\lambda y_n)] = \left(1 + h\lambda + \frac{1}{2}h^2\lambda^2\right) y_n,$$

so $Q(h\lambda) = 1 + h\lambda + \frac{1}{2}(h\lambda)^2$. If we assume λ is real, then in order to satisfy $|Q(h\lambda)| < 1$, we must have $-2 < h\lambda < 0$. It follows that the larger $|\lambda|$ is, the smaller h must be. □

The test equation can also be used to determine how to choose h for a multistep method. The process is similar to the one used to determine whether a multistep method is stable, except that we use $f(t, y) = \lambda y$, rather than $f(t, y) \equiv 0$.

Given a general multistep method of the form

$$\sum_{i=0}^{s} \alpha_i y_{n+1-i} = h \sum_{i=0}^{s} \beta_i f_{n+1-i},$$

we substitute $f_n = \lambda y_n$ and obtain the *recurrence relation*

$$\sum_{i=0}^{s} (\alpha_i - h\lambda\beta_i) y_{n+1-i} = 0.$$

It follows that the computed solution has the form

$$y_n = \sum_{i=1}^{s} c_i n^{p_i} \mu_i^n,$$

where each μ_i is a root of the *characteristic polynomial*

$$Q(\mu, h\lambda) = (\alpha_0 - h\lambda\beta_0)\mu^s + (\alpha_1 - h\lambda\beta_1)\mu^{s-1} + \cdots + (\alpha_{s-1} - h\lambda\beta_{s-1})\mu + (\alpha_s - h\lambda\beta_s).$$

The exponents p_i range from 0 to the multiplicity of μ_i minus one, so if the roots are all distinct, all p_i are equal to zero. In order to ensure that the numerical solution y_n decays to zero as n increases, we must have $|\mu_i| < 1$ for $i = 1, 2, \ldots, s$. Otherwise, the solution will either converge to a nonzero value, or grow in magnitude.

Example Consider the 3-step Adams-Bashforth method

$$y_{n+1} = y_n + \frac{h}{12}[23f_n - 16f_{n-1} + 5f_{n-2}].$$

Applying this method to the test equation yields the characteristic polynomial

$$Q(\mu, h\lambda) = \mu^3 + \left(-1 - \frac{23}{12}h\lambda\right)\mu^2 + \frac{4}{3}h\lambda\mu - \frac{5}{12}h\lambda.$$

Let $\lambda = -100$. If we choose $h = 0.1$, so that $\lambda h = -10$, then $Q(\mu, h\lambda)$ has a root approximately equal to -18.884, so h is too large for this method. On the other hand, if we choose $h = 0.005$, so that $h\lambda = -1/2$, then the largest root of $Q(\mu, h\lambda)$ is approximately -0.924, so h is sufficiently small to produce a qualitatively correct solution.

Next, we consider the 2-step Adams-Moulton method

$$y_{n+1} = y_n + \frac{h}{12}[5f_{n+1} + 8f_n - f_{n-1}].$$

In this case, we have

$$Q(\mu, h\lambda) = \left(1 - \frac{5}{12}h\lambda\right)\mu^2 + \left(-1 - \frac{2}{3}h\lambda\right)\mu + \frac{1}{12}h\lambda.$$

Setting $h = 0.05$, so that $h\lambda = -5$, the largest root of $Q(\mu, h\lambda)$ turns out to be approximately -0.906, so a larger step size can safely be chosen for this method. □

In general, larger step sizes can be chosen for implicit methods than for explicit methods. However, the savings achieved from having to take fewer time steps can be offset by the expense of having to solve a nonlinear equation during every time step.

The *region of absolute stability* of a one-step method or a multistep method is the region R of the complex plane such that if $h\lambda \in R$, then a solution computed using h and λ will decay to zero, as desired. That is, for a one-step method, $|Q(h\lambda)| < 1$ for $h\lambda \in R$, and for a multistep method, the roots $\mu_1, \mu_2, \ldots, \mu_S$ of $Q(\mu, h\lambda)$ satisfy $|\mu_i| < 1$.

Because a larger region of absolute stability allows a larger step size h to be chosen for a given value of λ, it is preferable to use a method that has as large a region of absolute stability as possible. The ideal situation is when a method is *A-stable*, which means that its region of absolute stability contains the entire left half-plane, because then, the solution will decay to zero regardless of the choice of h.

An example of an A-stable one-step method is the *Backward Euler method*

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

an implicit method. For this method,

$$Q(h\lambda) = \frac{1}{1-h\lambda},$$

and since $\operatorname{Re}\lambda < 0$, it follows that $|Q(h\lambda)| < 1$ regardless of the value of h. The only A-stable multistep method is the *implicit trapezoidal method*

$$y_{n+1} = y_n + \frac{h}{2}[f_{n+1} + f_n],$$

because

$$Q(\mu, h\lambda) = \left(1 - \frac{h\lambda}{2}\right)\mu + \left(-1 - \frac{h\lambda}{2}\right),$$

which has the root

$$\mu = \frac{1 + \frac{h\lambda}{2}}{1 - \frac{h\lambda}{2}}.$$

The numerator and denominator have imaginary parts of the same magnitude, but because $\operatorname{Re}\lambda > 0$, the real part of the denominator has a larger magnitude than that of the numerator, so $|\mu| < 1$, regardless of h.

Implicit multistep methods, such as the implicit trapezoidal method, are often used for stiff differential equations because of their larger regions of absolute stability. Because y_{n+1} appears on both sides of the difference equation for such methods, it is necessary to use an iterative method such as Newton's method to compute y_{n+1}. For a general implicit multistep method, for which $\beta_0 \neq 0$, Newton's method is applied to the function

$$F(y) = \alpha_0 y + \sum_{i=1}^{s} \alpha_i y_{n+1-i} - h\beta_0 f(t_{n+1}, y) - h\sum_{i=1}^{s} \beta_i f_{n+1-i}.$$

The resulting iteration is

$$\begin{aligned} y_{n+1}^{(k+1)} &= y_{n+1}^{(k)} - \frac{F(y_{n+1}^{(k)})}{F'(y_{n+1}^{(k)})} \\ &= y_{n+1}^{(k)} - \frac{\alpha_0 y_{n+1}^{(k)} + \sum_{i=1}^{s} \alpha_i y_{n+1-i} - h\beta_0 f(t_{n+1}, y_{n+1}^{(k)}) - h\sum_{i=1}^{s} \beta_i f_{n+1-i}}{\alpha_0 - h\beta_0 f_y(t_{n+1}, y_{n+1}^{(k)})}, \end{aligned}$$

with $y_{n+1}^{(0)} = y_n$.