Math 361S Lecture Notes
Numerical integration of ODEs

Jeffrey Wong

April 12, 2018

Topics covered

- Overview
  - Brief review of ODEs
  - Perspective for numerical methods
- Euler's method
  - Local and global truncation error; consistency
  - Convergence; proof of convergence
- Runge-Kutta methods
  - (Briefly) Taylor series method
  - Runge-Kutta methods: the general idea; form of the equations
  - Truncation error
- Adaptive schemes
  - Using two methods of different order (embedded RK methods)
  - Step doubling
- Stability
  - Stiff differential equations
  - Regions of absolute stability
- Other methods
  - Multistep methods

1 Introduction

1.1 Review of ODEs

See sections 7.1-7.3 of Moler's book for a different overview.

1.1.1 Initial value problems, examples

An **initial value problem** (IVP) for a function $y(t)$ is an ODE and an initial condition:

$$y' = f(t, y), \qquad y(t_0) = y_0. \tag{1}$$

The solution $y(t)$ satisfies the initial condition and the ODE at all t in some neighborhood of the initial time t_0. Here $f(t, y)$ is assumed to be a continuous function.

Higher-order ODEs involve more derivatives, e.g.

$$y'' + y = 0, \qquad y(0) = 1, y'(0) = 1.$$

The initial problem has a number of conditions at t_0 equal to the order of the highest derivative.

1.1.2 Converting to first order systems; general form

We can always convert a higher-order ODE to a first order system of ODEs

$$\mathbf{v}' = F(t, \mathbf{v})$$

by defining

$$v_1 = y, \quad v_2 = y', \cdots \quad v_n = y^{(n-1)}.$$

Then

$$v_1' = v_2, \quad v_2' = v_3, \cdots$$

and the last equation is the ODE, now in terms of $v_1, \cdots v_n$ instead of derivatives of y. For this reason, we will only consider solving systems of ODEs.

Furthermore, most of the results here in the scalar case extend in a straightforward way to systems, so we will mostly consider scalar ODEs.

1.1.3 Examples

The pendulum equation for the angular position $\theta(t)$, starting from rest at some angle:

$$\theta'' + \sin\theta = 0, \qquad \theta(0) = \theta_0, \, \theta'(0) = 0$$

which can be written as a first-order system with $v_1 = \theta$ and $v_2 = \theta'$:

$$v_1' = v_2, \qquad v_2' = -\sin v_1.$$

The logistic equation:

$$y' = y(1-y), \qquad y(0) = y_0$$

Exponential growth:

$$y' = \lambda y, \qquad y(t_0) = y_0$$

which has exact solution $y(t) = y_0 e^{\lambda(t-t_0)}$.

1.1.4 Existence and stability

Throughout, it will be assumed that a constant L exists such that

$$\left|\frac{\partial f}{\partial y}(t,y)\right| \leq L \tag{2}$$

for all (t, y) in the domain of interest. We will be a bit casual with 'domain' here, so some results may not be quite as rigorous as they could be.

2 Euler's method: analysis

The simplest method for solving an ODE is **Euler's method**. Let h, the spacing between grid points in time, be fixed (for simplicity; it does not have to be) and consider times $t_0, t_1, \cdots$. To get $y(t+h)$ from $y(t)$, expand in a Taylor series and use the ODE to simplify the derivatives:

$$y(t+h) = y(t) + hy'(t) = y(t) + hf(t,y) + O(h^2)$$

This gives

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

where $y_n = y(t_n)$ and τ_{n+1} is the **local truncaton error** (the error incurred by the Taylor series approximation going from t_n to t_{n+1}). We could derive a formula, but the important thing is that

$$\tau_{n+1} = O(h^2).$$

The numerical method is obtained by dropping the error and iterating this formula at the discrete times:

$$y_{n+1} = y_n + hf(t_n, y_n), \qquad y_0 = y(t_0).$$

Notice that the total error is **not** just the sum of the truncation errors, because f is evaluated at the approximation $\tilde{y}_n$. The truncation error will propagate through the iteration, as a careful analysis will show.

2.1 Error analysis

Let $y^*(t)$ be the exact solution to the IVP. Assume $t_0 = 0$ and suppose we integrate up to a time $t_n = b$ using times $t_0, \cdots t_n$ with fixed time-step h[1]. Let $y_n^* = y^*(t_n)$ and define the error

$$e_n = y_n^* - y_n.$$

[1]We shall think of the end time b as a fixed quantity (the goal) and wish to know how the error in $y^*(b)$ changes as $h \to 0$ (note that n changes but t_n does not here).

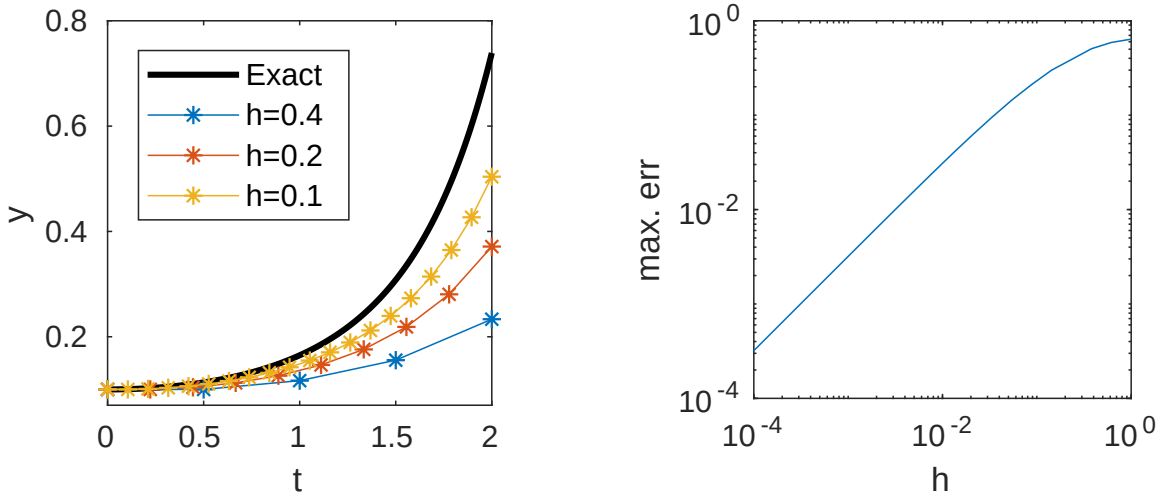
The goal is to find a bound on the maximum error between the approximation y_n and the exact solution $y(t_n)$ in the interval $[0, b]$:

$$\max_{0 \le k \le n} |e_n| = O(?).$$

As an example, consider

$$y' = ty, \qquad y(0) = 0.1$$

which has exact solution $y(t) = 0.1e^{t^2}$. Below, we plot some approximations for various time-steps h; on the right is the max. error in the interval. The log-log plot has a slope of 1, indicating the error should be $O(h)$.



To begin the analysis, write the iteration for y_n and the 'exact' formula with truncation error that was used to derive it:

$$y^*_{n+1} = y^*_n + hf(t_n, y^*_n) + \tau_{n+1}$$
$$y_{n+1} = y_n + hf(t_n, y_n)$$

Subtracting the second formula from the first and using the mean value theorem gives

$$e_{n+1} = e_n + hf_y(t_n, \xi_n)e_n + \tau_{n+1}$$

so

$$|e_{n+1}| \le (1 + Lh)|e_n| + |\tau_{n+1}|.$$

Iterating this, we find that

$$|e_n| \le (1 + Lh)^n |e_0| + \sum_{k=1}^{n} (1 + Lh)^{k-1} |\tau_k|$$

Since

$$(1 + Lh) \le e^{Lh}$$

we obtain

$$|e_n| \leq e^{Lnh}|e_0| + \left(\frac{e^{Lnh}-1}{Lh}\right)\max|\tau_k|$$

But $nh = b$ so

$$|e_n| \leq e^{Lb}|e_0| + \left(\frac{e^{Lb}-1}{L}\right)\frac{\max|\tau_k|}{h}.$$

But the local truncation error is $O(h^2)$, so the global truncation error is $O(h)$. If the 'initial error' e_0 is zero (true if we use the exact initial value for the ODE), then this means that the error is $O(h)$ as $h \to 0$:

$$\max_{0\leq k\leq n}|e_k| = O(h).$$

Note that the above is the'max error' in the approximation in the interval $[0, b]$:

$$\max_{k \text{ s.t. } 0\leq t_k\leq b}|e_k|$$

This is how far apart the approximation y_n is from y_n^* if plotted as a function - at least at the discrete points where it is defined[2].

2.2 Order

The **order** p of a numerical method for an ODE is the order of the **global** truncation error. Euler's method, for instance, has order 1 since the global error is $O(h)$.

The $1/h$ factor shows up in the error analysis for methods in general, so it is true that

$$\text{LTE} = O(h^{p+1}) \implies \text{global error} = O(h^p).$$

Thus a method will be order p if the LTE is $O(h^{p+1})$. **Warning:** Some texts define the LTE with an extra factor of h so that it lines up with the global error.

2.3 Taylor series methods (briefly)

It is straightforward to derive a formula using higher-order Taylor series. The one thing to note is that since $y(t)$ is a function of t,

$$(f(t,y))' = \frac{\partial f}{\partial t} + y'\frac{\partial f}{\partial y}$$

by the chain rule, e.g.

$$(\sin y)' = y'\cos y.$$

By differentiating the ODE repeatedly we can calculate $y^{(n)}$ for any n but the formulas quickly become unwieldy. However, the method will improve the local truncation error (and hence the global error).

[2]To get an approximation that exists at all t, we would need to then do some interpolation, which would introduce a bit more error in between the grid points.

3 Runge-Kutta methods

It is desirable to have a method that does not require all this computation in advance - that only needs to know how to evaluate $f(t, y)$. The trick here is to replace $y'', y''', \cdots$ with function evaluations at point in-between t_n and t_{n+1}. This gives us some easy-to-evaluate expressions whose Taylor series contain derivatives of y.

Intuitively, what we are doing is taking a sequence of partial steps to get to t_{n+1} and combining them in the right way. To illustrate via example, let us derive a method with an $O(h^3)$ LTE.

To start, note that for the solution $y(t)$ to the ODE,

$$y(t+h) = y(t) + hy'(t) + \frac{1}{2}h^2 y''(t) + O(h^3)$$

so

$$y(t+h) = y(t) + hf + \frac{1}{2}h^2(f_t + ff_y) + O(h^3)$$

where f, f_t, f_y are short for f and its partial derivatives evaluated at $(t, y(t))$. The goal is to find a way to express $y(t+h)$ as $y(t)$ plus a weighted sum of f's evaluated at certain points,

$$y(t+h) = y(t) + w_1 h f_1 + w_2 h f_2 + O(h^3).$$

The right choice turns out to be

$$f_1 = hf(t, y), \qquad f_2 = hf(t + \alpha h, y + \beta f_1).$$

Note that f_2 uses f_1 and that f_1 is easy to compute; this is important for the numerical method.

The goal is to get the Taylor series expression and the RK formula to match up to $O(h^3)$. We do this by brute force - expand out the formula and solve for coefficients. Here, only f_2 takes some work:

$$\begin{aligned} f_2 &= hf(t + \alpha h, y + \beta hf) \\ &= h(f + \alpha h f_t + (\beta hf) f_y + O(h^2) \\ &= hf + \alpha h^2 f_t + \beta h^2 f f_y + O(h^3). \end{aligned}$$

This is good because it contains the ff_y and f_t terms. Adding things up, we get

$$y(t+h) = y + h(w_1 f + w_2 f) + w_2 h^2(\alpha f_t + \beta f f_y) + O(h^3).$$

To get this to match the Taylor series, we need

$$w_1 + w_2 = 1, \qquad w_2\alpha = 1/2, \qquad w_2\beta = 1/2.$$

The numerical method is obtained by replacing t with t_n and y with $\tilde{y}_n$. For example, we could take $w_1 = 0, w_2 = 1$ and $\alpha = \beta = 1$ to obtain the **modified Euler method**

$$\begin{aligned} f_1 &= hf(t_n, y_n) \\ f_2 &= hf(t_n + \frac{1}{2}h, y_n + \frac{1}{2}f_1) \\ y_{n+1} &= y_n + f_2 \end{aligned}$$

In this case one can interpret the formula as using the midpoint rule to estimate

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(t, y(t))\, dt \approx y(t_n) + hf(t_n + h/2, y(t_n + h/2))$$

but using Euler's method to estimate the midpoint value:

$$y(t_n + h/2) \approx y_n + \frac{h}{2} f(t_n, y_n).$$

However, in general, the various intermediate quantities do not have a clear interpretation, and the formulas can appear somewhat mysterious.

3.1 Higher-order methods

A general **explicit** m-step Runge-Kutta method uses m values hf_j that are f evaluated at some time and a y-value involving a linear combination of previous f_i's The formula has the form

$$\begin{aligned} f_1 &= hf(t_n, y_n) \\ f_2 &= hf(t_n + c_2 h, y_n + a_{21} f_1) \\ f_3 &= hf(t_n + c_3 h, y_n + a_{31} f_1 + a_{32} f_2) \\ &\vdots \\ f_m &= hf(t_n + c_m h, y_n + a_{m1} f_1 + \cdots + a_{m,m-1} f_{m-1}) \\ y_{n+1} &= y_n + w_1 f_1 + \cdots + w_m f_m. \end{aligned}$$

The best possible local truncation error is $O(h^{p+1})$ where $p \leq m$. For $m \leq 4$, the maximum p is equal to m (as you may expect) but for $m > 4$, the maximum p is strictly less than m. The system turns out to be underdetermined, so we can then pick coefficients so the method has other properties (see later). The downside is that the system is non-linear and the Taylor series expansions become quite complicated, so solving the system is extremely tedious.

Thankfully, just about every useful set of coefficients - at least for general purpose methods - has been calculated already, so in practice one can just look them up. They are typically arranged in the **Butcher Tableau**, which lists the c_i's, the a_{ij}'s and w_i's in a nice order (look this up for details).

3.2 The Classical Runge-Kutta method

One four step method of note is the 'classical Runge-Kutta method (sometimes called 'RK4')

$$\begin{aligned} f_1 &= hf(t_n, y_n) \\ f_2 &= hf(t_n + \frac{1}{2}h, y_n + \frac{1}{2}f_1) \\ f_3 &= hf(t_n + \frac{1}{2}h, y_n + \frac{1}{2}f_2) \\ f_4 &= hf(t_n + h, y_n + f_3) \\ y_{n+1} &= y_n + \frac{1}{6}(f_1 + 2f_2 + 2f_3 + f_4). \end{aligned}$$

This method has a good balance of efficiency and accuracy (only four function evaluations per step, and $O(h^5)$ LTE, so an $O(h^4)$ global error). It is a good first choice when integrating an ODE using a fixed time step h without any special properties.

3.3 Implicit methods

We'll revisit this shortly, but it is possible to include y_{n+1} in the formulas as well (among other variations). For example, the implicit version of Euler's method is the Backward Euler method

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

The obvious downside is that we are then using y_{n+1} in intermediate steps to compute y_{n+1} which means that the RK formula is actually a non-linear equation that defines y_{n+1} implicitly. To find y_{n+1}, we obtain a zero of

$$g(x) = x - y_n - hf(t_{n+1}, x).$$

The equation is usually solved with Newton's method; the initial guess $x = y_n$ is good since as $y_{n+1} = y_n + O(h)$. This is far less efficient per step, but as we will see later, it has some advantages.

4 Adaptive time stepping

Notice that the RK methods use only the values at the previous time step n to update to time $n+1$ (this is called a 'one step' method). For this reason, once we are at $n+1$, we are free to pick a new value of h. It is desirable to have an algorithm that can adjust the time step h to ensure that the local truncation error remains below a certain tolerance:

$$|\tau_i| < \epsilon \text{ for all } i.$$

To do so, we need only get an estimate for τ_i at each step, and then 'solve' for the value of h that will ensure the estimate is below ϵ.

4.1 Using two methods of different order

One cheap way to do this is by using two methods of different order. Suppose we have two methods of lower/higher order producing y_{n+1} and $\hat{y}_{n+1}$:

$$y_{n+1} = y_n + \psi(t_n, y_n),$$
$$\hat{y}_{n+1} = y_n + h\hat{\psi}(t_n, y_n).$$

Then, if y_n is exact,

$$y(t_{n+1}) = y_{n+1} + O(h^{p+1}), \qquad y(t_{n+1}) = \hat{y}_{n+1} + O(h^{q+1})$$

where $q > p$ (usually just $p+1$). Let us estimate the error in the first formula by Ch^{p+1} and subtract:

$$\delta := |y_{n+1} - \hat{y}_{n+1}| \approx Ch^{p+1}$$

since the other truncation error is higher order. We need a new time step h_u such that

$$Ch_{\text{new}}^{p+1} < \epsilon \implies C(h_{\text{new}}/h)^{p+1}h^{p+1} < \epsilon \implies \delta(h_{\text{new}}/h)^{p+1} < \epsilon.$$

We should therefore pick

$$h_{\text{new}} \approx (\epsilon/\delta)^{1/(p+1)}.$$

In practice, because this is just an estimate, one puts an extra coefficient in to be safe, typically something like

$$h_{\text{new}} = 0.9h(\epsilon/\delta)^{1/(p+1)}.$$

This formula decreases the step size if the error is too large (to stay accurate) and increases the step size if the error is too small (to stay efficient). Sometimes, other controls are added (like not decreasing or increasing h by too much per time step).

4.2 Embedded RK methods

For estimating error in this way, it is ideal for the two methods to share as much work as possible per step. 'Embedded pairs' of RK methods are formed by cleverly selecting coefficients so that the f_i's used for the high order approximation can also be used for the low order one. For example, the coefficients for a second order RK method can be selected so that Euler's method (1st order) shows up:

$$\begin{aligned} f_1 &= hf(t_n, y_n) \\ f_2 &= hf(t_n + \frac{1}{2}h, y_n + \frac{1}{2}f_1) \\ y_{n+1} &= y_n + f_1 \\ \hat{y}_{n+1} &= y_n + f_2 \end{aligned}$$

We showed earlier that $\hat{y}_{n+1}$ has error $O(h^3)$ and y_{n+1} has error $O(h^2)$. The value of f_1 is 'recycled' in that it is used for both methods. The timestep would then be chosen to be something like

$$h_{\text{new}} \approx h(\epsilon/|y_{n+1} - \hat{y}_{n+1}|)^{1/2}.$$

to make sure y_{n+1} (the Euler's method step) has a LTE below ϵ.

There are several higher-order embedded methods that are popular, including the **Runge-Kutta-Fehlberg** method, which is a fourth-order method that has a fifth-order estimate used for the time-step selection. It requires only 5 function evaluations in total[3].

4.3 Step doubling

We can also estimate the error using a single one-step method using a trick similar to Richardson extrapolation: take two steps of size $h/2$ and one step of size h to get two approximations, then estimate the error (details to appear on homework).

[3]MATLAB's `ode45` uses this process to integrate ODEs; it uses the Dormand-Prince method.

5 Stability

5.1 Introduction

Stability refers to the sensitivity of solutions to initial conditions. First, it is worth reviewing the notion of stability for IVPs. As before, consider the IVP

$$y' = f(t, y), \qquad y(t_0) = y_0 \tag{3}$$

and assume that

$$\left|\frac{\partial f}{\partial y}(t, y)\right| \leq L \tag{4}$$

The following theorem ensures a unique solution exists:

Theorem. *If (4) holds for some interval $t \in [a, b]$ containing t_0 and all y and f is continuous then the IVP (3) has a unique solution in $[a, b]$.*

Moreoever, if y_1 amd y_2 are two solutions to the ODE with different initial conditions then

$$|y_1(t) - y_2(t)| \leq e^{L(t-t_0)}|y_1(t_0) - y_2(t_0)|.$$

which describes the **stability** of the ODE - how sensitive the solution is to changes in the initial condition.

We would like to have a corresponding notion of 'numerical' stability. Roughly, if $\{y_n\}$ and $\{z_n\}$ are approximate solutions to (3) in $[a, b]$ then we seek a condition like

$$|y_n - z_n| \leq C|y_0 - z_0| + O(h^p)$$

where C is *independent* of n. Note that the best we can hope for is $C = e^{L(t-t_0)}$ since the numerical method will never be more stable than the actual IVP. In what follows, we will try to determine the right notions of stability for the numerical method.

Reminder: We found that differeniation is numerically unstable (error diverges as $h \to 0$) and composite integration rules are numerically stable (error is bounded as $n \to \infty$).

5.2 Consistency, stability and convergence

Suppose we solve (3) in the interval $[t_0, b]$ numerically. Given a time step h, we generate an approximation $y_0, \cdots y_n$ at points

$$t_0 < t_1 < t_2 < \cdots < t_n = b$$

with spacing h[4]. Recall that the method is **convergent** if the error over the whole interval goes to zero as the timestep goes to zero:

$$\lim_{h \to 0} \left(\max_{0 \leq k \leq n} |y_k - y(t_k)| \right) = 0.$$

For numerical DEs, proving convergence is done with the following steps:

[4]Note: the definitions can be easily modified to allow for varying spacings; convergence occurs as the maximum spacing goes to zero.

- Show that the 'local' error at each step is small. ('consistency')
- Show that local errors do not propagate too much. ('stability')
- Prove that the first two conditions imply convergence.

consistency: A method is called **consistent** if

$$\text{the LTE at each step is } o(h) \text{ as } h \to 0.$$

(here $o(h)$ means some order smaller than h, i.e. $\lim_{h\to 0} \tau/h = 0$). To check consistency, we need only find the LTE, which is typically done when deriving the method in the first place. Note that this is a condition on a single step of the iteration; we are not concerned with the maximum over the interval.

Checking consistency: Euler's method,

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

is consistent since the truncation error is

$$\tau_{n+1} = y(t_{n+1}) - y(t_n) - hf(t_n, y(t_n)) = \frac{h^2}{2} y''(\xi_n)$$

where ξ_n lies between t_n and t_{n+1}. For each n, the error is $O(h^2)$ as $h \to 0$. From the point of view of consistency, n is fixed so $y''(\xi_n)$ is just some constant; we do not need a uniform bound on y'' that is true for all n.

Stability: For stability, a precise statement depends on the methods involved. Ideally, we want something that is easy to check and that will imply convergence for a consistent method. One useful notion of stability is called **0-stability**. Informally, this says that

$$\text{for a fixed end time } b, \text{ the approx. } \{y_n\} \text{ stays bounded as } n \to \infty.$$

An example illusrates this best:

An unstable method: Consider two methods:

$$\begin{aligned} y_{n+1} &= y_n + hf(t_n, y_n) \\ -y_{n+2} &= -4y_{n+1} + 3y_n + 2hf(t_n, y_n) \end{aligned}$$

Note: The second method comes from approximating y' by a forward difference with three points:

$$y'(t) = \frac{-y(t+2h) + 4y(t+h) - 3y(t)}{2h} + O(h^2).$$

For the first method (Euler's method), we showed a bound

$$|y_n - y(t_n)| \leq C_1|y_0 - y(t_0)| + C_2 h$$

directly. so the method is stable (this, of course, also shows convergence).

The second method is unstable! It is enough to check for the trivial ODE $y' = 0$. The iteration reduces to

$$y_{n+2} = 4y_{n+1} - 3y_n.$$

Plugging in $y_n = r^n$ we get a solution when

$$r^2 - 4r + 3 = 0 \implies r = 1,\, 3$$

so the general solution is

$$y_n = a + b \cdot 3^n.$$

If initial values y_0, y_1 are chosen so that $y_0 = y_1$ then $y_n = y_0$ for all n with exact arithmetic; however, if there are any errors, they will grow exponentially (like 3^n). We will revisit this stability issue later in discussing multistep methods.

As written, the zero stability condition is not easy to check. However, one can derive easy to verify conditions that imply zero stability.

Convergence: With some effort, one can then prove a convergence theorem of the form

$$\text{consistency} \;+\; \text{`stability'} \implies \text{convergence}$$

where 'stability' refers to 0-stability. Moreover, it is true that

$$LTE = O(h^{p+1}) \implies \text{global error} = O(h^p).$$

Note that we verified this was true for Euler's method directly; for multi-step methods the proof is much more involved.

This paradigm is common throughout numerical analysis of differential equations. For some class of methods / problems, one tries to find the right definition of 'stability' so that consistency and stability will imply convergence, and so that 'stability' can be proven without too much trouble.

6 Stiffness

Why not use an adaptive RK method for every ODE? This question leads to a critical idea in numerical analysis of differential equations. Consider the ODE[5]

$$y' = -20(y - \sin t) + \cos t, \quad y(0) = 1 \tag{5}$$

for $t \in [0, 3]$. The solution is

$$y(t) = e^{-20t} + \sin t.$$

It starts out decaying very rapidly, then 'settles down' and looks like $\sin t$ after this transient is done. To obtain a reasonable solution for small t ($\sim 1/20$), we need to pick h small enough that the numerical solution can resolve the features of the actual solution. To do so, the grid of time points must be fine enough to capture the variations in $y(t)$.

However, once the initial transient e^{-20t} is gone, we would like to use larger time steps to compute the solution. What happens if Euler's method is used?

The solution $y(t)$ is quite well behaved, and moreover, it is also stable in the sense that if perturbed a bit, the solution will not change much. But to get a reasonable solution, we need to choose a small h. This is a **stability** constraint; it is a requirement for the numerical method to give anything meaningful.

This leads us to a first definition of stiffness:

> **(Informal definition I:)** An IVP is **stiff** in an interval if Euler's method requires a much smaller time step h to be stable than it does to be accurate.

Here the 'accuracy requirement' is that all the solution features are resolved as in the first example and the 'stability requirement' is this mysterious constraint to be explored shortly. Our goal here is to explain the phenomenon and what determines the required step size.

6.1 Another interpretation

Nearby solutions with some other initial condition rapidly converge to $y(t)$ (if the solution is perturbed, it will quickly return). The numerical method should, but doesn't, always behave the same way. if h is not small enough, Euler's method will overshoot due to the large slope. This process will continue, causing aggressive oscillations that will diverge. If h is at a certain value, then the oscillations will stay bounded. Below this value, the overshoot is not severe and the method provides a reasonable approximation. This leads us to another version of stiffness:

[5]The definition of stifness and example are borrowed from Ascher & Petzold, *Computer methods for ordinary differential equations.*

(Informal definition II:) An IVP with solution $y(t)$ (in an interval) is stiff if the nearby trajectories to $y(t)$ vary rapidly compared to $y(t)$ itself.

Again, note that this is not the same as instability of the IVP itself because these rapidly varying trajectories may converge to $y(t)$ (i.e. the solution $y(t)$ could be stable from a theoretical perspective). The unstable behavior is a deficiency of the numerical method.

6.2 Analysis for Euler's method

Now let us provide a mathematical description and derive this stability condition. Let us consider a 'test equation'

$$y' = \lambda y, \qquad y(0) = y_0 \tag{6}$$

where λ is a (complex) number. The solution is just $y(t) = y_0 e^{\lambda t}$ so

- If $\mathrm{Re}(\lambda) < 0$ then $y(t) \to 0$ as $t \to \infty$
- If $\mathrm{Re}(\lambda) > 0$ then $y(t)$ grows exponentially.

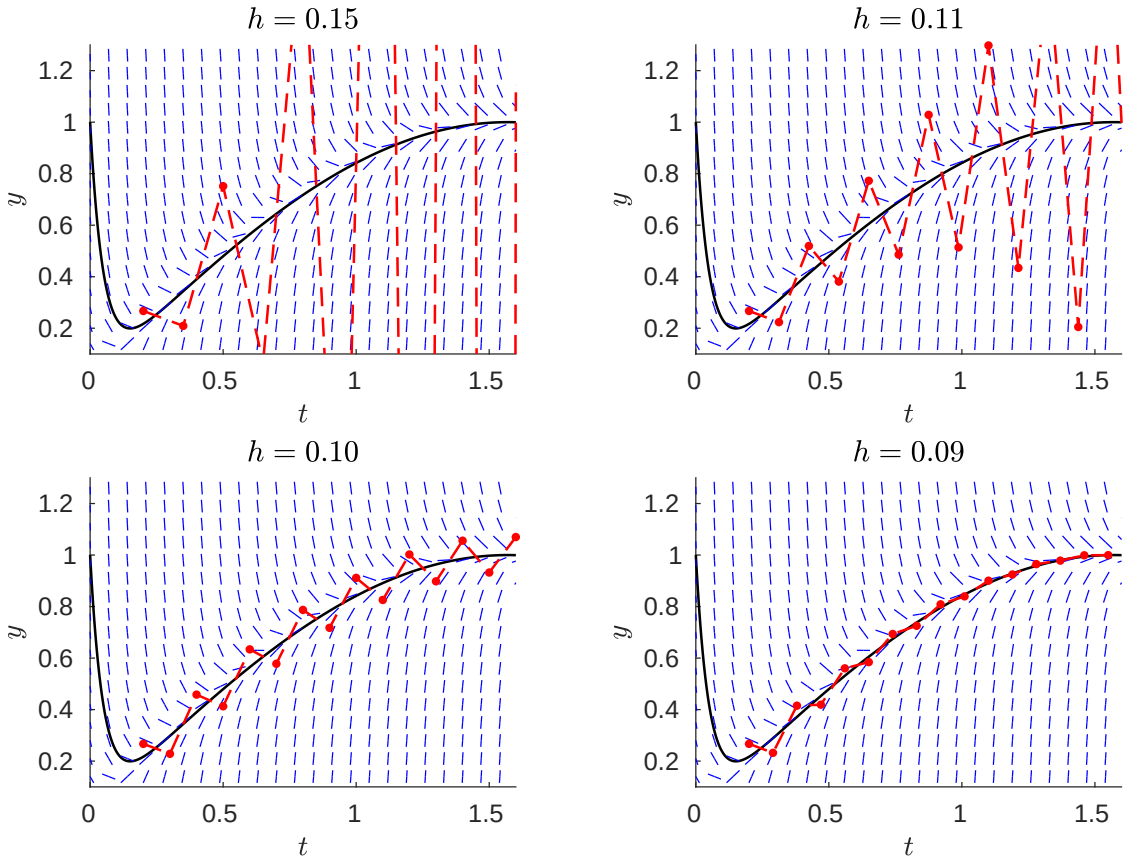


Figure 1: Euler's method applied to the stiff IVP (5). The starting point is $(t_0, y(t_0))$ where $t_0 = 0.2$. The blue lines are the lope y' at that value of (t, y).

What does the numerical method do? For Euler's method, we have

$$y_{n+1} = y_n + hf(t_n, y_n) = (1 + h\lambda)y_n \implies y_n = (1 + h\lambda)^n y_0.$$

Now define a region R of the complex plane as the set of $h\lambda$ for which the iteration has $|y_n| \to 0$ as $n \to \infty$ (i.e. the set of $h\lambda$ such that Euler's method applied to (6) gives a sequence that converges to zero). Then

$$R = \{z \in \mathbb{C} : |1 + z| < 1\}.$$

This is a circle of radius 1 centered at $z = -1$. In particular, $|y_n| \to 0$ if

$$h < \frac{2}{|\lambda|}.$$

Moreover, if $h > 2/|\lambda|$ then $|y_n|$ will increase exponentially. This means that in order to have y_n decay exponentially when the exact solution does, the condition (**??**) must hold. Otherwise, y_n will grow exponentially even though it should go to zero.

For a general ODE, we can linearize about y_n to obtain

$$y' \approx f(t_n, y_n) + \frac{\partial f}{\partial y}(y - y_n) + \frac{\partial f}{\partial t}(t - t_n) + \cdots$$

so the analogue of λ here is $\frac{\partial f}{\partial y}$. Indeed, this matches the observed behavior from the earlier example where $\frac{\partial f}{\partial y} = -20$.

Note that the $\lambda > 0$ is not as much of a concern here; if $\lambda > 0$ then $1 + h\lambda > 1$ for any positive h so y_n will grow exponentially as desired.

6.3 Stability regions

Now consider a method that produces a sequence y_n of approximations. Let us define the **Region of absolute stability** for a method to be the set $R \subset \mathbb{C}$ of $z = h\lambda$ such that $|y_n| \to 0$ when the method is applied to the test equation (6) with fixed time step h.

Note that we have a set of $h\lambda$ but h and λ appear separately; this is because the region itself only depends on the product for all the methods discussed here.

For example, for the backward Euler method

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

we get

$$y_{n+1} = \frac{1}{(1 - h\lambda)} y_n$$

so $|y_n| \to 0$ when

$$|1 - h\lambda| > 1.$$

The region of absolute stability is then the set $\{z \in \mathbb{C} : |1-z| > 1\}$ which is the complement of a circle centered at $z = 1$ of radius 1. In particular, R contains the entire half plane $\{z < 0\}$, which means that if $\mathrm{Re}\lambda < 0$ then $|y_n| \to 0$ for any positive value of h.

(pictures with rays)

A method for which R contains all of $\{z < 0\}$ is called **A-stable**. For such a method, there is no stability condition required when $\lambda < 0$, which is ideal for integrating stiff equations.

It turns out that explicit methods always have bounded regions of absolute stability, so there is always some stability constraint (true, e.g. for every explicit RK method). Some implicit methods can have unbounded stability regions, in particular ones that at least contain the real negative value $(-\infty, 0)$. This means that to integrate a stiff equation, one typically needs to use an implicit method (e.g. Backward Euler).

Our first definition of stiffness can be made a bit more specific:

> **(Informal definition III:)** An ODE is stiff if $|\frac{\partial f}{\partial y}|$ is large in the sense that $1/|\frac{\partial f}{\partial y}|$ is much less than the typical width of a 'change' in $y(t)$.

Note that RK methods will all have stability regions with some maximum radius; for the most accurate ones, like Euler's method, this radius is roughly 2 (give or take an order of magnitude) so the constant 1 in the definition above is a bit arbitrary but close enough.

6.4 More stability regions

Stability regions for explicit RK methods and some implicit methods are shown below. It is not too hard to show that for a second-order, two-step explicit RK method (as derived earlier), the stability region is

$$\{z \in \mathbb{C} : |1 + z + \frac{1}{2}z^2| < 1\},$$

independent of the choice of the 'free' coefficients.

6.5 Extra: other notions of stability

There is much more to the story than just absolute stability. Many other notions of stability exist, describing other constraints and finer properties of numerical solutions. For example, we may want it to be true that

$$\text{the rate at which } |y_n| \to 0 \text{ increases as } \mathrm{Re}\lambda \to -\infty.$$

For instance, if $\lambda = -1000$ we would want the approximation to decay to zero much faster than if $\lambda = -10$. Backward Euler has this property (called L-stability or 'stiff decay') since

$$|y_{n+1}| = \frac{1}{|1 - h\lambda|}|y_n|$$

so as $\mathrm{Re}\lambda \to -\infty$, the coefficient goes to zero in magnitude. The property is nice because it means that **fast-decaying transients are damped out quickly** by the method.

6.6 Limitations of A-stability

It is worth noting that A-stability is not necessarily the ideal for a method. If $\mathrm{Re}\lambda$ is small in magnitude but $\mathrm{Im}\lambda$ is large, then solutions oscillate rapidly; the 'stability' $|y_n| < |y_{n-1}|$ offered by an A-stable method does not really help with this. Thus it can be okay for a method to have a stability region that includes part of the real axis $(-\infty, 0)$ but does not include the imaginary axis, e.g. something like:

...

A-stability is a rather strong demand to make, so often a region like the one above is the best one can do while still having good accuracy.

7 Multistep methods

A **linear multistep method** has the form

$$\sum_{j=0}^{m} a_j y_{n-j} = h \sum_{j=0}^{m} b_j f_{n-j} \tag{7}$$

where $f_{n-j} = f(t_{n-j}, y_{n-j})$. The methods are so named because they involve values from the m previous steps, and are linear in f (unlike Runge-Kutta methods). For example, Euler's method

$$y_n = y_{n-1} + h f_{n-1}$$

and the Backward Euler method

$$y_n = y_{n-1} + h f_n$$

are both (trivially) linear multistep methods that only involve the previous step; we call these **one step methods**. Note that the method (7) is implicit if $b_0 \neq 0$ and explicit otherwise. We are interested here in methods that use more than just the previous step.

For simplicity, we will assume throughout that the discrete times are $t_0, t_1, \cdots$ with fixed timestep h.

7.1 Adams methods

The starting point is the integrated form of the ODE:

$$y(t_n) = y(t_{n-1}) + \int_{t_{n-1}}^{t_n} y'(t)\, dt.$$

Note that $y'(t) = f(t, y(t))$, but it is convenient to leave it as y'.

Change variables to $t = t_n + sh$. Then

$$y(t_n) = y(t_{n-1}) + h \int_{-1}^{0} y'(t_n + sh)\, ds. \tag{8}$$

To get an explicit method using the previous m values, we estimate the integral using the previous times $t_{n-1}, t_{n-2}, \cdots t_{n-m}$, or equivalently $s = -1, -2, \cdots - m$:

$$\int_{-1}^{0} g(s)\, ds = \sum_{j=1}^{m} b_j g(-j) + C g^{(m)}(\xi).$$

We already know how to derive such formulas. For example, for $m = 2$,

$$\int_{-1}^{0} g(s)\, ds = \frac{3}{2} g(-1) - \frac{1}{2} g(-2) + \frac{5}{12} g^{(2)}(\xi).$$

Now plugging in $y'(t_n + sh)$ into the formula, we get

$$y(t_n) = y(t_{n-1}) + \frac{3h}{2} y'(t_{n-1}) - \frac{h}{2} y'(t_{n-2}) + \frac{5}{12} y^{(3)}(t_n + h\xi) h^3$$

for some $\xi \in (-m, 0)$. The numerical method is then

$$y_n = y_{n-1} + \frac{3h}{2} f_{n_1} - \frac{h}{2} f_{n-2}.$$

In the notation of (7), we have $m = 2$ and $a_0 = 1, b_1 = 3/2, b_2 = -1/2$. The other a-values are zero: $a_1 = a_2 = 0$.

A method that uses the m previous steps to estimate the integral is called an **Adams-Bashforth** method. For a general m, it takes the form

$$y_n = y_{n-1} + h \left(\sum_{j=1}^{m} b_j f_{n-j} \right).$$

The local truncation error is $O(h^{m+1})$, so the method has order m. There is a trick for deriving the coefficients; see example below.

We can also derive a method from (8) by including t_n (or $s = 0$). With $m = 1$ this would be the trapezoidal rule:

$$\int_{-1}^{0} g(s)\, ds = \frac{1}{2} g(-1) + \frac{1}{2} g(0) + C g^{(2)}(\xi).$$

Note that this method uses $m + 1$ points (the m previous points plus t_n). The result is then the **trapezoidal method**

$$y_n = y_{n-1} + \frac{h}{2}(f_n + f_{n-1})$$

which has order 2. In general, the method will have the form

$$y_n = y_{n-1} + h\left(\sum_{j=0}^{m} b_j f_{n-j}\right).$$

and is called an **Adams-Moulton** method. The local truncation error is $O(h^{m+2})$, so the method is order $m+1$. Note that the method is implicit, since f_n appears on the RHS.

Examples: The Adams-Bashforth method for $m = 3$ requires a formula

$$\int_{-1}^{0} g(s)\,ds \approx b_1 g(-1) + b_2 g(-2) + b_3 g(-3).$$

We will omit the details of the error term here. The easiest way to derive the coefficients is to use undetermined coefficients on the Newton basis:

$$1, \quad s+1, \quad (s+1)(s+2), \cdots$$

The reason is that the resulting linear system will be triangular and easy to solve (and can in fact there is a nice general formula). The method should have degree of accuracy 2, so we require that it be exact for $1, s+1$ and $(s+1)(s+2)$. Plugging these in, we get

$$\begin{aligned} 1 &= b_1 + b_2 + b_3 \\ \frac{1}{2} &= -2b_2 - b_3 \\ \frac{5}{6} &= 2b_3 \end{aligned}$$

so $b_1 = 23/12, b_2 = -16/12$ and $b_3 = 5/12$. This gives the method

$$y_{n+1} = y_n + \frac{h}{12}(23f_{n-1} - 16f_{n-2} + 5f_{n-3})$$

which has order of accuracy 3. The Adams-Moulton method with the same order of accuracy has $m = 2$ and requires the formula

$$\int_{-1}^{0} g(s)\,ds \approx b_0 g(0) + b_1 g(-1) + b_2 g(-2).$$

Requiring that the formula is exact for $1, s, s(s+1)$ (same trick as before) yields

$$\begin{aligned} 1 &= b_0 + b_1 + b_2 \\ -\frac{1}{2} &= -b_1 - 2b_2 \\ -\frac{1}{6} &= 2b_2. \end{aligned}$$

After solving, the result is the method

$$y_n = y_{n-1} + \frac{h}{12}\left(8f_n + 5f_{n-1} - f_{n-2}\right)$$

with order of accuracy 3.

7.2 Properties of the Adams methods

The main benefit of these methods is efficiency: one function evaluation is required per step.

It is also true that

- The order p Adams-Moulton method has a (much) smaller error than the order p Adams-Bashforth method (the constant for the LTE is much smaller).
- When $m \geq 2$, the Adams-Moulton methods have a bounded stability region of a moderate size. (When $m = 0$ or $m = 1$ the method is A-stable)
- Adams-Bashforth methods have a bounded stability region that is very small.

For this reason, neither type of method (even the implicit one!) is useful for stiff problems. The Adams-Moulton method is superior in terms of accuracy and (absolute) stability, but it is implicit - so it requires much more work per step.

In practice, the two are combined to form an explicit method that has some of the stability of the implicit method, but is easy to compute. The implicit term $f_n = f(t_n, y_n)$ is estimated using the result $\tilde{y}_n$ from an explicit method. This strategy is called a **predictor-corrector method**. For example, we can combine the two-step explicit formula with the one-step implicit formula:

$$\tilde{y}_n = y_{n-1} + \frac{h}{2}(3f_{n-1} - f_{n-2}) \tag{9}$$

$$\tilde{f}_n = f(t_n, \tilde{y}_n) \tag{10}$$

$$y_n = y_{n-1} + \frac{h}{2}(\tilde{f}_n + f_{n-1}). \tag{11}$$

Now the formula is not implicit (and as a bonus, the error can be estimated using y_n and $\tilde{y}_n$ by standard techniques). It turns out that this trick gives a method that has a reasonably good stability region (not as good as the implicit one!) and is essentially strictly better than the pure explicit method (9) alone.

7.3 Starting values

To start, we need m previous values, which means $y_1, \cdots y_{m-2}$ must be computed by some other method (not a multistep formula). A high-order RK method is the simplest choice, because it only requires one previous point.

7.4 Other multistep methods

The Adams methods only use previous f_n's and not y_n's. There are other classes of multistep methods of the form (7) that are derived in different ways. One such class are the **Backward differentiation formulas** (BDFs). These are derived by approximating y' using a backward difference:

$$hy'(t_n) \approx c_0 y(t_n) + c_1 y(t_{n-1}) + \cdots + c_m y(t_{n-m}).$$

For example, the first order BDF is Backward Euler; the second order BDF is

$$\frac{-3y_n + 4y_{n-1} + y_{n-2}}{h} = f(t_n, y_n)$$

which rearranges to

$$y_n = -\frac{4}{3}y_{n-1} + \frac{1}{3}y_{n-2} + \frac{2h}{3}f_n.$$

This method is sometimes called **Gears' method**. BDFs are valuable because for orders up to 6, they do well on stiff problems (see next section) compared to Adams-Moulton methods, and moreover have the stiff decay property.

8 Analysis of multistep methods

8.1 Consistency

The general multi-step method (7) can be shown to be consistent by Taylor expanding $y(t_{n-1}), \cdots$ and $y'(t_{n-1})$ (in place of f) around t_n, then canceling out terms to get the local truncation error. We have

$$y(t_{n-j}) = y(t_n) - jhy'(t_n) + \frac{1}{2}(jh)^2 y''(t_n) + \cdots$$

and

$$y'(t_{n-j}) = y'(t_n) - jhy''(t_n) + \frac{1}{2}(jh)^2 y'''(t_n) + \cdots .$$

The truncation error is

$$\begin{aligned}
\tau_n &= \sum_{j=0}^{m} a_j y(t_{n-j}) - h\sum_{j=0}^{m} b_j y'(t_{n-j}) \\
&= \sum_{j=0}^{m} a_j y(t_n) - h\sum_{j=0}^{m} j a_j y'(t_n) - h\sum_{j=0}^{m} b_j y'(t_n) + O(h^2) \\
&= y(t_n)\sum_{j=0}^{m} a_j + hy'(t_n)\left[\sum_{j=0}^{m}(b_j - ja_j)\right] + O(h^2).
\end{aligned}$$

Thus the method (7) is consistent if and only if

$$\sum_{j=0}^{m} a_j = 0, \qquad \sum_{j=0}^{m} b_j = \sum_{j=0}^{m} j a_j.$$

Further conditions can be derived by taking more terms. Of course, for the Adams formulas and BDFs, we derived the method and proved consistency by other (more elegant) means, but with the above we can construct more general methods.

8.2 Zero stability

One necessary condition for convergence is that when applied to the trivial ODE $y' = 0$, solutions remain bounded. If $f = 0$ then the method reduces to

$$\sum_{j=0}^{m} a_j y_{n-j} = 0 \tag{12}$$

which is a linear recurrence for y_n. This turns out to be sufficient! The following theorem is true:

Theorem 1 (Dahlquist equivalence theorem)**.** *A linear multistep method of the form (7) is convergent if and only if it is consistent and all solutions to (12) remain bounded as $n \to \infty$.*

The equation (12) can be solved directly [see last section for a review], leading to the condition

Theorem 2. *All solutions of the linear recurrecne (12) are bounded if and only if all the roots of the characteristic polynomial*

$$p(r) = \sum_{j=0}^{m} a_j r^{m-j} \tag{13}$$

have magnitude ≤ 1.

The Dahlquist Theorem lets us verify a multistep method is convergent simply by proving consistency and finding the roots of the characteristic polynomial (13).

8.3 Strongly stable methods

From the consistency result, we have that

if the method is consistent, $r = 1$ is a root of $p(r)$.

For zero stability, this is the 'good' root, since it means that if $y' = 0$ then $y_n = $ const. is a solution. The other roots are 'bad', in that they are just spurious terms that appear due to errors. If one of these roots is also 1, the bad term will stay bounded but not decay, which is not ideal.

A method is called **strongly stable** if there is one root $r = 1$ and all other roots are strictly less than 1 in magnitude. The method is **weakly stable** if there are multiple roots iwth magnitude 1.

Weakly stable methods tend not to damp out errors as much, which makes them undesirably unless they are needed for some other compelling reason.

8.4 Absolute stability

The definition of absolute stability is the same. When more than one previous y-value is involved, the recurrence has more than two terms, so the analysis is more involved. When the general method (7) is applied to $y' = \lambda y$ we get

$$\sum_{j=0}^{m} a_j y_{n-j} = h \sum_{j=0}^{m} \lambda b_j y_{n-j}.$$

Let $z = h\lambda$ as before. The above is a linear recurrence with characteristic polynomial

$$p(r; z) = \sum_{j=0}^{m} (a_j - z b_j) r^{m-j}.$$

To have solutions to the recurrence go to zero as $n \to \infty$, we need all the roots of $p(r, z)$ to have magnitude less than one. Thus, with R the region of absolute stability,

$$z \in R \text{ if and only if the roots of } p(r; z) = 0 \text{ all have } |r| < 1.$$

This is now a condition that can, with some effort, be computed and we can plot the stability region.

Example: Stability region for Adams methods The two-step Adams-Bashforth method is

$$y_n = y_{n-1} + h\left(\frac{3}{2} f_{n-1} - \frac{1}{2} f_{n-2}\right).$$

Applied to $y' = \lambda y$, this becomes

$$y_n = y_{n-1} + z\left(\frac{3}{2} y_{n-1} - \frac{1}{2} y_{n-2}\right).$$

with $z = h\lambda$. The characteristic polynomial for this recurrence is

$$p(r; z) = r^2 - r - z(\frac{3}{2} r + \frac{1}{2}).$$

Rearranging, we get (equivalently) that the roots satisfy

$$2r^2 - (2 + 3z)r + z = 0.$$

Solving, we get

$$r = \frac{2 + 3z \pm \sqrt{(2 + 3z)^2 - 8z}}{4}.$$

Thus z is in the region of absolute stability if and only if the two roots above are less than 1 in magnitude. This is not a nice looking condition. However, we can determine the largest b such that $(-b, 0)$ is in the interval of absolute stability.

Observe that on the boundary of R, one of the roots must have $|r| = 1$. If $z < 0$ is real then both roots are real (by the calculations above). Thus we need only find the values of z for which $r = \pm 1$ is a root, which occurs when

$$z = 0 \text{ or } z = -1.$$

Thus the interval of absolute stability is $(-1, 0)$, which is half the size of Euler's method!

Some facts absolute stability regions for multistep methods:

- A linear multistep method with order ≥ 2 cannot be A-stable. (This is the **second Dahlquist barrier**)
- The interval of absolute stability for BDFs up to order 6 contains $(-\infty, 0)$
- The stability region for Adams-Moulton methods is bounded for $m > 2$
- The stability region for Adams-Bashforth methods shrinks as the order increases.

Plots of the stability regions are shown below. It follows from these plots that higher order Adams-Moulton methods are not to be used on stiff problems even though they are implicit, and that Adams-Bashforth methods should not be used on anything remotely stiff[6]

8.5 Difference equations: review

Consider the linear difference equation

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots c_m a_{n-m}.$$

We solve by looking for solutions of the form

$$a_n = r^n.$$

This is a solution if r is a root of the characteristic polynomial

$$p(r) := r^n - c_1 r^{n-1} - c_2 r^{n-2} - \cdots - c_{m-1} r - c_m.$$

There are m (complex) roots $r_1, \cdots r_m$. Since the equation is linear, any linear combination is a solution. The general solution is therefore

$$a_n = \sum_{j=1}^{m} b_j r_j^n$$

for constants $b_1, \cdots b_m$. Note that the solution is bounded for any initial conditions if and only if

$$|r_j| \leq 1 \text{ for all } j$$

and that $a_n \to 0$ for any initial conditions if and only if

$$|r_j| < 1 \text{ for all } j.$$

[6] More to the point, predictor-corrector methods using Adams-Bashforth and Adams-Moluton formulas work much better, so there is not much reason to use the Adams-Bashforth formula alone.