Avoiding stability-induced inefficiencies in BDF methods

Kris STEWART
Department of Mathematical Sciences, San Diego State University, San Diego, CA 92182-0314, USA

Received 9 May 1989
Revised 24 August 1989

Abstract: Time-dependent partial differential equations are often treated by semidiscretization and the resulting problem solved using existing ordinary differential equation software restricted to low-order formulas. For certain classes of problems, the Backward Differentiation Formulas (BDF) are often dismissed due to their poor stability behavior near the imaginary axis for orders three and above. We explain why and for what problems this happens, what the appropriate tactic should be, and why this is not the tactic taken by most automatic codes. We present an idea that avoids this inefficiency in one automatic code.

Keywords: BDF, order selection, stability, stiff ODE.

1. Introduction

The finite-element method and the method of lines are popular semidiscretization techniques which convert a system of partial differential equations (PDEs) into a larger system of ordinary differential equations (ODEs). The parabolic equations that describe diffusion of heat yield a system of ODEs in time where the eigenvalues of the linearized problem tend to be on or near the negative real axis and exhibit a wide range of decay constants. ODEs with this property are called stiff, occur in other contexts and impose stability restrictions on the stepsize used by the ODE technique [16]. Backward Differentiation Formulas (BDF) work well in this case. Implicit Runge–Kutta techniques are also appropriate for stiff ODE systems, but their cost is much higher than BDF when the eigenvalues are primarily real and negative.

If the PDE is parabolic with both diffusion and advection active, as in Burgers' equation with moderate Reynolds number, we have a linearized ODE system with some eigenvalues having large, negative real part (stiff components) as well as eigenvalues near the imaginary axis (nonstiff components). Explicit ODE techniques are inappropriate for coupled systems with stiff and nonstiff components due to the small stepsize required for stability of the stiff component. Implicit Runge–Kutta are frequently proposed, despite their cost, because they are stable in the entire left half-plane.

Implicit BDFs of order three and above have a stability limitation when solving linear problems with eigenvalues near the imaginary axis. We show that the reason standard BDF codes exhibit poor behavior there is due to incorrect order selection. We explain why the standard order selection strategies do not work in this case and propose a technique that avoids this limitation.

2. The difficulty

We examine the behavior of a collection of codes on a problem which is stiff and which has a pair of eigenvalues near the imaginary axis. The Stiff Test Set [2] is a battery of tests often used to compare implementations of ODE methods. Though this test set is not an exhaustive test [15], it is widely available and contains a well-known problem with the characteristics we desire to explore.

The family B2 to B5 examines behavior for complex eigenvalues in linear, constant coefficient systems using a parameterized family with increasingly dominant imaginary part in the eigenvalue of largest modulus.

$$y' = \begin{bmatrix} -10 & \alpha & 0 & 0 & 0 & 0 \\ -\alpha & -10 & 0 & 0 & 0 & 0 \\ 0 & 0 & -4 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.1 \end{bmatrix} y, \quad y(0) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

with eigenvalues $\lambda_i = -0.1, -0.5, -1, -4, -10 \pm \alpha i$. B2 sets $\alpha = 1$, B3 sets $\alpha = 8$, B4 sets $\alpha = 25$ and B5 sets $\alpha = 100$. B5, in particular, produces poor behavior when most, current BDF based codes are used.

The following results are obtained for low and moderate absolute accuracy tolerances using STRUT (Stewart and Krogh) [21], LSODE (Hindmarsh) [5], LSODA (Petzold and Hindmarsh) [12], and DC03A (Curtis) [1] using the CRAY X-MP/48 at the NSF SuperComputer Center, San Diego. Error is computed at each step by comparison with the results of LSODE using an accuracy tolerance four orders of magnitude smaller than the test run. NG indicates the number of iteration matrices formed. NJ is the number of ODE Jacobians (STRUT and DC03A distinguish the need for Jacobian from need for iteration matrix). Time is CPU-time. NBDF is the number of BDF steps (LSODA and STRUT automatically detect stiffness and switch between Adams and BDF methods).

As shown in Tables 2 and 4, the codes LSODE and DC03A are expensive for B5, but not for B4 given in Tables 1 and 3. For most of the integrations presented above, the order that LSODE and DC03A use is constant and the stepsize is nearly constant. We analyze the reasons for this

Table 1
Stiff test set no. B4; absolute tolerance $= 1.0 \cdot 10^{-2}$

Code	Error	Step	NFE	NG	NJ	Time	NBDF
STRUT	$1.9 \cdot 10^{-2}$	49	89	7	1	$1.9 \cdot 10^{-2}$	25
LSODA	$2.2 \cdot 10^{-2}$	67	124	6	6	$1.2 \cdot 10^{-1}$	23
LSODE	$2.5 \cdot 10^{-2}$	56	67	12	12	$1.4 \cdot 10^{-2}$	
DC03A	$2.0 \cdot 10^{-2}$	59	87	16	1	$1.8 \cdot 10^{-2}$	

Table 2
Stiff test set no. B5; absolute tolerance $=1.0\cdot10^{-2}$

Code	Error	Step	NFE	NG	NJ	Time	NBDF
STRUT	$1.6\cdot10^{-1}$	136	168	9	1	$4.5\cdot10^{-2}$	117
LSODA	$5.6\cdot10^{-2}$	166	324	7	7	$2.9\cdot10^{-1}$	23
LSODE	$2.6\cdot10^{-1}$	2354	2568	120	120	$5.0\cdot10^{-1}$	
DC03A	$1.1\cdot10^{-1}$	2371	2631	17	1	$4.6\cdot10^{-1}$	

Table 3
Stiff test set no. B4; absolute tolerance $=1.0\cdot10^{-4}$

Code	Error	Step	NFE	NG	NJ	Time	NBDF
STRUT	$2.4\cdot10^{-4}$	116	208	6	1	$3.3\cdot10^{-2}$	40
LSODA	$3.0\cdot10^{-4}$	132	252	6	6	$2.1\cdot10^{-1}$	36
LSODE	$5.4\cdot10^{-4}$	113	133	17	17	$2.7\cdot10^{-2}$	
DC03A	$2.6\cdot10^{-4}$	121	159	26	1	$3.2\cdot10^{-2}$	

Table 4
Stiff test set no. B5; absolute tolerance $=1.0\cdot10^{-4}$

Code	Error	Step	NFE	NG	NJ	Time	NBDF
STRUT	$1.8\cdot10^{-3}$	239	438	9	1	$5.5\cdot10^{-2}$	55
LSODA	$8.2\cdot10^{-4}$	318	620	9	9	$5.2\cdot10^{-1}$	44
LSODE	$1.7\cdot10^{-3}$	2350	2716	142	142	$5.4\cdot10^{-1}$	
DC03A	$1.2\cdot10^{-3}$	2460	2760	65	1	$5.0\cdot10^{-1}$	

poor performance in the next three sections using the standard absolute stability theory for constant order and stepsize on a linear, constant coefficient system, such as B5. LSODA and STRUT result in efficient integrations with slowly varying order and stepsize and this is discussed in Section 6 and 7, respectively. Absolute stability theory is not strictly applicable in this case, but does give insight to explain the success of LSODA and STRUT.

3. The standard stability theory with a z-plane focus

Consider the initial value problem

$$y' = \lambda y.$$

The kth-order BDF in ordinate form is

$$(1 - h\lambda\beta_0)y_n - \sum_{r=1}^{k} \alpha_r y_{n-r} = 0. \tag{1}$$

The *stability polynomial* is defined to be

$$\pi_k(z;\ h\lambda) = \rho(z) + h\lambda\ \sigma(z). \tag{2}$$

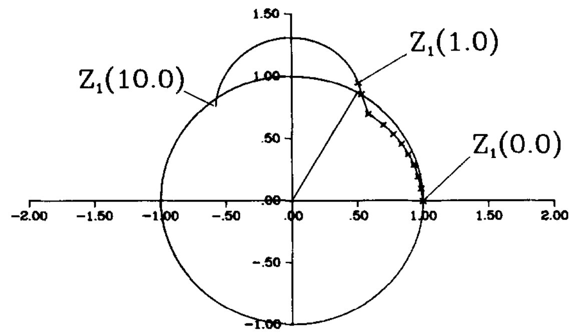

Fig. 1. z-plane stability plot for fifth-order BDF; $\text{Arg}(h\lambda) = 95°$, $|h\lambda| = 0(0.1)10$; with ray marked at $\text{Arg}(z) = 60°$.

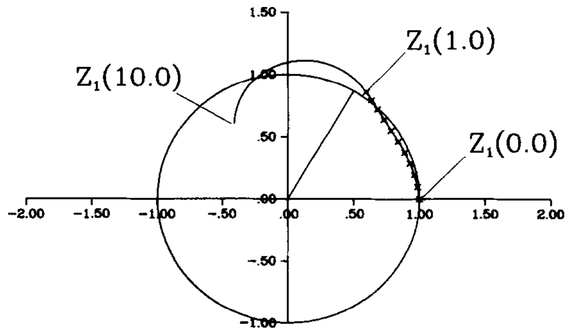

Fig. 2. z-plane stability plot for fourth-order BDF; $\text{Arg}(h\lambda) = 95°$, $|h\lambda| = 0(0.1)10$; with ray marked at $\text{Arg}(z) = 60°$.

For order six and below, the BDF methods converge as $h \to 0$ since they are zero-stable, that is the zeros of $\pi_k(z;\ 0)$ are inside the unit disk or on the unit circle and simple.

Though one requires zero-stability for a usable method, stiff problems also require stability for large values of $h\lambda$. For a given value of $h\lambda$, there are k independent solutions to the scalar difference equation corresponding to the k roots of $\pi_k(z;\ h\lambda) = 0$. Label them $z_1(h\lambda), \ldots, z_k(h\lambda)$ and call the plane in which we plot the roots the *z-plane*. To show stability of a method on a particular problem, one fixes $h\lambda$, computes the k roots in the z-plane and verifies that their magnitude is less than one, or simple when one.

A range of $h\lambda$ values is examined by fixing the argument of $h\lambda$, and varying its modulus. The plot that explains B5 involves the dominant eigenvalue $\lambda = -10 + 100\text{i}$, which has an argument of roughly 95°. Four roots of $\pi_5(z;\ h\lambda) = 0$ remain well within the unit circle, thus Fig. 1 presents the plot of $z_1(h\lambda)$, for $|h\lambda| = 0(0.1)10$ and with $\text{Arg}(h\lambda) = 95°$.

The principle root, $z_1(h\lambda) = e^{h\lambda} + O(h^{k+1}))$, proceeds outside the unit circle in the z-plane when $|h\lambda| > 0.9$. This explains the poor behavior of LSODE which used $K = 5$ with stepsize increasing to $h = 0.0097$, so that $|h\lambda| = |0.0097(-10 + 100\text{i})| > 0.9$. With this $|h\lambda|$, instability amplifies errors. Automatic codes see this as an error estimate that exceeds the requested tolerance and "control" it by small stepsize reductions. When the stepsize is reduced, $z_1(h\lambda)$ is then within the unit circle in the z-plane and errors are no longer magnified. After several steps the error estimator predicts that a larger stepsize can be used and the process repeats. This *ratcheting* in and out of the stability region is typical for codes with finite-stability regions when faced with stiffness and results in a stepsize effectively on the stability boundary [14].

We show in Section 4 that, if any $z_k(h\lambda)$ leaves the unit circle and goes unstable, it is the *exit angle* of that $z_k(h\lambda)$ that is crucial. In Fig. 1, the principle root, $z_1(h\lambda)$, has exit angle of 58°, $z_1(0.9) \cong e^{\text{i}58°}$. If any eigenvalue has $\text{Arg}(\lambda) = 95°$, a stable numerical solution is obtained with a stepsize such that $|h\lambda| < 0.9$, or $|h\lambda| \geqslant 8.75$, since this is where $z_1(h\lambda)$ re-enters the unit circle in the z-plane.

In Table 4 with $\text{Tol} = 10^{-4}$, DC03A does lower its order from $K = 5$ to $K = 4$, where it remains fixed for the rest of the integration. In Fig. 2, we examine the root behavior of the fourth-order BDF on a problem with $\text{Arg}(\lambda) = 95°$. Only the principle root, $z_1(h\lambda)$, has magnitude greater than one, leaving the unit circle when $|h\lambda| = 0.8$ with an exit angle of 47°. The principle root re-enters the unit circle when $|h\lambda| = 4$. Thus, for a linear problem with some

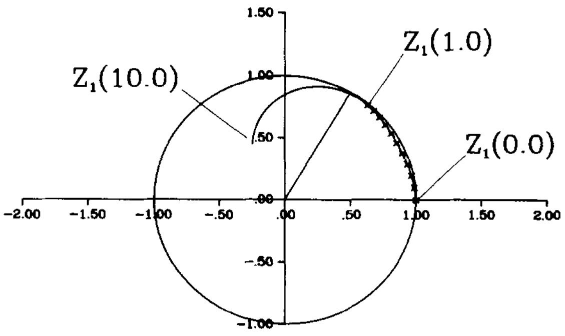

Fig. 3. z-plane stability plot for third-order BDF; $\text{Arg}(h\lambda) = 95°$, $|h\lambda| = 0(0.1)10$; with ray marked at $\text{Arg}(z) = 60°$.

eigenvalue of argument 95°, instability in the numerical method will restrict h so that $|h\lambda| < 0.8$ or $|h\lambda| > 4$.

The proper tactic for the code to maintain stability and allow a larger stepsize is to lower order. For $\text{Arg}(\lambda) = 95°$ the third-order BDF is stable for all h, as is seen from the plot of the position of the largest root of $\pi_3(z;\ h\lambda) = 0$ in Fig. 3. There is no exit angle when $\text{Arg}(\lambda) = 95°$.

The question: "Why didn't DC03A and LSODE lower order?"

4. Cause of difficulty

We show that the cause of the difficulty in solving B5 is due to the order selection mechanism. Automatic codes estimate the local error after each step. Order selection is typically based on comparing error estimates for the different order formulas available. Although several strategies are possible, Hall and Watt [4, p.99] outline a commonly used technique. Let E_k estimate the local error of a method of order k,

$$E_k \approx \delta^*_{k+1} h^{k+1} y^{(k+1)} \approx \delta^*_{k+1} \nabla^{k+1} y,$$

where ∇^k is the kth backward difference and $\delta^*_k = 1/k$. Analogous expressions hold for E_{k-1} and E_{k+1}. Estimates of a new stepsize, $h_{\text{new}} = r\, h_{\text{old}}$, to match the user's accuracy request Tol are computed for the different order formulas, with the change r_k for a method of order k given by

$$r_k = \left(\frac{\text{Tol}}{|E_k|} \right)^{1/(k+1)}.$$

Let order be selected to allow the largest stepsize increase. Production codes use more elaborate strategies, but the basic design is along these lines. This provides an effective order selection strategy when accuracy is the main concern. When stability is the concern, this still provides an effective strategy for many practical methods, but not for the third- through sixth-order BDF on problems near the imaginary axis.

To see why the order selection strategy did not lower order, we examine the numerical solution in detail. The computed solution is given by $y_n = c_1 z_1(h\lambda)^n + \cdots + c_4 z_4(h\lambda)^n$ for the fourth-order BDF. When stable, each root $z_j(h\lambda) = \rho\, e^{i\theta}$ has $\rho < 1$. We assume we are accurate and are following the principle root, so $|c_1| \gg |c_j|$, $j = 2,\ldots,4$, and examine the behavior of the backward differences of $[z_1(h\lambda)]^n \approx y_n$. As the stability boundary is approached, we expect $|z_1(h\lambda)| \approx 1$ and set $z_1(h\lambda) = e^{i\theta}$. Using $\nabla^j z^n = z^n(1 - 1/z)^j$, the differences are given by

$$\nabla^j y_n = \nabla^j[z_1(h\lambda)]^n = [z_1(h\lambda)]^n\left(1 - \frac{1}{z_1(h\lambda)}\right)^j = e^{in\theta}(1 - \cos\theta + i\sin\theta)^j.$$

The function

$$D(\theta) = |(1 - \cos\theta + i\sin\theta)| = \sqrt{2 - 2\cos\theta} = 2\sin\tfrac{1}{2}\theta \tag{3}$$

gives the magnitude of ratios of successive differences of $z_1(h\lambda)$ when it is near the stability boundary as a function of the angle θ.

To lower order from k, we require $r_{k-1} > r_k$, i.e.,

$$\left(\frac{\text{Tol}}{|E_{k-1}|}\right)^{1/k}\left(\frac{\text{Tol}}{|E_k|}\right)^{-1/(k+1)} > 1.$$

when the difficulty occurs in B5, LSODE and DC03A remain essentially at fixed order. The stepsize smoothly increases until $z_1(h\lambda)$ leaves the unit circle in the z-plane. The error estimator then grows and the stepsize is reduced. Due to ratcheting, the stepsize is such that $z_1(h\lambda)$ is kept close to the stability boundary while maintaining $E_k \approx \text{Tol}$. The method is on the perceived accuracy boundary due to the principle root being on the stability boundary. $\text{Tol}/|E_k| \approx 1$, so its $(k+1)$st power is nearly its kth power. With this assumption, the left-hand side above is

$$\left(\frac{|E_k|}{|E_{k-1}|}\right)^{1/k} = \left(\frac{\delta^*_{k+1}|\nabla^{k+1}y|}{\delta^*_k|\nabla^k y|}\right)^{1/k} = \left(\frac{k}{k+1}D(\theta)\right)^{1/k}.$$

We are concerned with the value of θ when z_1 leaves the unit circle, which we have already labeled the exit angle in the z-plane. Order reduction will depend on $(k/(k+1))D(\theta)$ being greater than one. For different orders k, we present the value of the angle θ where $(k/(k+1))D(\theta)$ equals one, thus the smallest exit angle of the method to allow the order mechanism described above to indicate order reduction.

Note, $D(60°) = 1$ and D is monotonically increasing, so a method whose exit angle is greater than $60°$ results in an increase in the norm of successive differences when the method goes unstable. Examining the ratios of successive error estimators instead of differences only increases the minimum exit angle of the method to be beyond $60°$ to indicate an order reduction when the method goes unstable. To simplify the discussion, we focus on successive differences and the angle $60°$. The plots in Figs. 1 and 2 have the ray at $60°$ marked and show that the fourth- and fifth-order BDF on B5 have exit angles less than $60°$, thus no order reduction will occur.

One can summarize the previous z-plane plots for all $\text{Arg}(h\lambda)$ by a single plot in the $h\lambda$-plane if one displays the argument of z that produces the absolute stability plots. The absolute stability

Table 5
Minimum exit angle of BDF method for error estimator to indicate order reduction

k	2	3	4	5	6
θ	97.18°	83.62°	77.35°	73.74°	71.37°

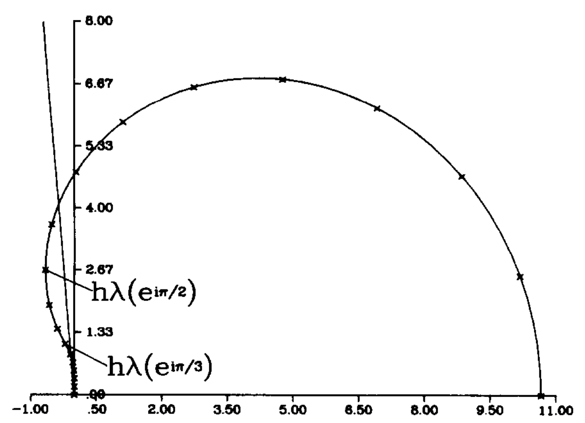

Fig. 4. $h\lambda$-plane stability plot for fourth-order BDF; "$\times$" for each $\text{Arg}(z) = 0°(10°)180°$; with ray marked at $\text{Arg}(h\lambda) = 95°$.

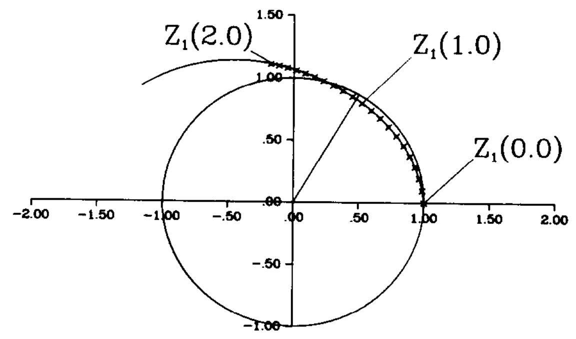

Fig. 5. z-plane stability plot for third-order Adams–Moulton; $\text{Arg}(h\lambda) = 95°$, $|h\lambda| = 0(0.1)5$; with ray marked at $\text{Arg}(z) = 60°$.

regions for Adams and BDF methods computed by Krogh [6–10] include Arg(z). Lately, the common practice [3,4,17] is to display absolute stability regions, without this information.

To demonstrate the advantage of retaining Arg(z), we develop the absolute stability region for the fourth-order BDF. Absolute stability plots are obtained using the boundary locus method [11, p.82]. Instead of finding the k zeros z_j of $\pi_k(z;\ h\lambda)$ for fixed $h\lambda$, one maps the unit circle in the z-plane into the $h\lambda$-plane using the difference equation and the substitution $y_n = z^n$. The equation

$$h\lambda z^n = \sum_{r=1}^{k} \frac{1}{r} \nabla^r z^n \tag{4}$$

is satisfied by any zero of $\pi_k(z,\ h\lambda)$. Using $\nabla^r z^n = z^n(1 - 1/z)^r$ and $z = e^{i\theta}$ in (4), we compute the stability boundary in the $h\lambda$-plane from

$$h\lambda = \sum_{r=1}^{k} \frac{1}{r}(1 - e^{-i\theta})^r$$

for values of $e^{i\theta}$ with $\theta = 0°(10°)180°$ for the fourth-order BDF in Fig. 4. We see the lobe of instability in the left half-plane and the arguments of z that correspond. The ray drawn for $\text{Arg}(h\lambda) = 95°$ intersects the unstable lobe at $|h\lambda| \approx 0.8$ with Arg(z) near 50°. This is in agreement with the z-plane plot in Fig. 2 which showed the root $z_1(h\lambda)$ leaving the root unit circle when $|h\lambda| = 0.8$ with exit angle 47°.

5. Why LSODE and DCOA became "stuck"

The third- through sixth-order BDFs have an exit angle less than 60° if Arg(λ) lies between 90° and 95°. Generally stiff problems have transients in the initial part of the solution, and these transients cause the error control mechanism to select a stepsize small enough that $|z_j(h\lambda)| \ll 1$ for all j. As the transients decay out of the solution, h may increase to the point that $|z_1(h\lambda)| \approx 1$. Since order is not likely to be reduced if Arg(λ) is near the imaginary axis for methods with an unstable lobe in the left half-plane and exit angles less than 60°, the stepsize is constrained by stability rather than accuracy. This defeats the purpose of the stiff solver. Hence, DC03A and LSODE became "stuck" because the code reduces stepsize instead of order on B5.

Some methods, such as Adams, do not exhibit this problem for two reasons. First, the exit angle is large when Arg(λ) is near the imaginary axis causing order to be reduced, as we see in Fig. 5 for the third-order Adams-Moulton. The exit angle is 84°, yielding $D(84°) = 1.34$, growth in the differences and a likely order reduction. Second, the required value of h might not occur during the solution process due to accuracy considerations. As Krogh [7] notes, when solving $y'' = -y$, with $\lambda = \pm i$ and solution $y(t) = \sin t$, a stepsize of $h = 1.047$ radians = 60° would be too large to be accurate.

6. Why LSODA did not become stuck

LSODA is a modification (by Petzold [12]) of LSODE with automatic switching between Adams and BDF formulas based on an estimate of the Lipschitz constant [14]. In Table 2, LSODA is using the Adams method with order $K = 2$ when the switch is made to second-order BDF. The second-order BDF is A-stable, and the code remains at low order with increasing stepsize for the rest of the integration. In Table 4 with Tol = 10^{-4}, again the switch occurred with the Adams method at second order and $h = 0.0284$. Over several steps, the order was increased to $K = 4$ with stepsize increasing to $h = 0.277$ causing $z_1(h\lambda)$ to be stable since this places $h\lambda$ beyond the lobe of instability in Fig. 4. Equivalently this is the case for $z_1(h\lambda)$, with $|h\lambda| > 4$ in Fig. 2 where the principle root has re-entered the stability region.

LSODA was fortunate because the order selection mechanism of the Adams code had already reduced the order to $K = 2$ when the switch to BDF occurred. The eigenvalue which controls stiffness detection, $\lambda = -10 \pm 100i$, in B5 is the same complex eigenvalue that causes the BDF codes to become stability-limited. When problem B5 is altered by adding an additional differential equation

$$y_7' = -1000\ y_7$$

Table 6
Results for B5 modified to increase $|\lambda_{\max}|$

Stiff set no. B5—modified; absolute tolerance = $1.0 \cdot 10^{-2}$

Code	Error	Step	NFE	NG	NJ	Time	NBDF
STRUT	$2.4 \cdot 10^{-1}$	152	199	14	1	$5.7 \cdot 10^{-2}$	117
LSODA	$9.0 \cdot 10^{-2}$	1208	2271	62	62	$1.4 \cdot 10^{+0}$	1187
LSODE	$8.8 \cdot 10^{-2}$	128	156	20	20	$3.3 \cdot 10^{-2}$	
DC03A	$9.7 \cdot 10^{-2}$	1224	1382	32	1	$3.1 \cdot 10^{-1}$	

Stiff set no. B5—modified; absolute tolerance = $1.0 \cdot 10^{-4}$

Code	Error	Step	NFE	NG	NJ	Time	NBDF
STRUT	$4.2 \cdot 10^{-3}$	242	282	18	1	$1.0 \cdot 10^{-1}$	218
LSODA	$1.0 \cdot 10^{-3}$	306	503	25	25	$9.2 \cdot 10^{-1}$	71
LSODE	$1.7 \cdot 10^{-3}$	282	354	30	30	$7.3 \cdot 10^{-2}$	
DC03A	$1.2 \cdot 10^{-3}$	1328	1501	45	1	$3.4 \cdot 10^{-1}$	

to separate the eigenvalue controlling stiffness from the eigenvalue that induces the stability limitation, the following results were obtained (see Table 6).

With Tol $= 10^{-2}$, LSODA showed the same stability-limited behavior as in B5. LSODE had a double step failure on one step near $x = 0.3$ which caused an order reduction to three on the first failure and severe stepsize reduction on the second failed step. This reduced order allowed LSODE to avoid the stability-limited region. DC03A was never this fortunate and produced an expensive integration. LSODE was lucky this time and resulted in the most effective CPU-time of all the codes above. An additional run with Tol $= 10^{-6}$ resulted in all codes but STRUT experiencing an expensive, stability-limited integration. It is clear from the LSODE and LSODA performance that stability-limited stepsize does not always occur, but when it does, the resulting integration is prohibitively expensive.

7. Resolving the difficulty within STRUT

The code STRUT was implemented by Stewart [19] from algorithms due to Krogh [9] as well as near daily interactions with Krogh to discuss alternative ideas that came up during development. The code switches between Adams and BDF based on an estimate of the Lipschitz constant [14], with the addition of a check of the spectral radius (obtained from a power iteration) before the switch occurs. The Adams code in STRUT is STEP written by Shampine and Gordon [17] with the addition of the stiffness detector. The stiff code in STRUT is a semi-implicit BDF method [10,19] using direct estimates of the relative error in the correction matrix, to determine the number of corrections at each time step as well as the frequency of linear algebra updates. STRUT can change order and stepsize on each step.

The SIBDF inherits many of the properties of the BDF. Absolute stability of the SIBDF and BDF is identical when the error in the correction matrix is zero. B5 presents a situation where this is approximately the case.

For Tol $= 10^{-2}$, STRUT switches to the SIBDF with order $K = 4$ and $h = 0.00947$. For Tol $= 10^{-4}$, STRUT switches to the stiff code with order $K = 5$ and $h = 0.0132$. In both cases, the stepsize puts $z_1(h\lambda)$ near the stability boundary, but the code does successfully lower order. In STRUT, order is selected to maintain a "clean" difference table. A "clean" difference table reflects an adequate amount of convergence in the high differences, say

$$\|\nabla^{k+2}y_n\| \leq \tfrac{1}{2}\|\nabla^{k+1}y_n\| \leq \tfrac{1}{4}\|\nabla^{k}y_n\|.$$

When STRUT solved B5 in its early developmental stages, it became stuck at fifth order with stepsize $h = 0.00897$. One expects the high-order differences to reveal information about the stability-limiting root. Examining the largest component of the vector of differences over a sequence of time steps, $\nabla^{k+2}y_n$ was cyclic (roughly $-1.7 \cdot 10^{-7}$ to $2.0 \cdot 10^{-7}$) and went through a sign change after every third step. This continued for hundreds of steps with order and stepsize constant and zero matrix error, so BDF absolute stability theory was applicable. A stability-limited root which regularly goes through two sign changes every six steps has argument approximately $360^\circ/6 = 60^\circ$, the case for B5 being solved with a fifth-order BDF.

This led us to recognize the need for a modified order-selection strategy for a BDF code compared to an Adams code. The present test that allows STRUT to effectively solve this problem relies on several of the properties of the implementation of SIBDF idea. The goal is to force the code to reduce order when the sequences of differences fail to appear sufficiently convergent. We examine differences directly instead of error estimators because ratios of error estimators increase the required exit angle of the method, as shown in Table 5. Once order is selected, the stepsize is computed using the appropriate error estimators.

The order is lowered if

$$\left\| \beta_2 \, 1.2 \, \nabla^{k+2} y \right\| \geqslant \max\{ \| \nabla^{k+1} y \|, 0.9 \| \nabla^k y \| \}.$$

Here the backward difference notation is used, but the code actually uses modified divided differences [9], equivalent to backward differences after a sequence of steps with constant stepsize.

A key is β_2 which represents the history of recent stepsize changes,

$$\beta_2 = \frac{h_n}{h_{n-1}} \, \frac{h_n + h_{n-1}}{h_{n-1} + h_{n-2}}.$$

If the stepsize has been increasing over the past few steps, then $\beta_2 > 1$, increasing the likelihood the order will be reduced. This is due to the investigation by Rubner-Petersen [13] for stability of the BDF when the stepsize grows geometrically. Not surprisingly, his results show that a uniformly increasing stepsize reduced the absolute stability regions of his modified BDF. The factor of 1.2 is similar to the smoothness test proposed by Skelboe [18] where he examined this difficulty with the high-order BDF. The use of both $\nabla^{k+1} y$ and $\nabla^k y$ is to protect against the possibility of a high difference passing through zero.

Experimental results using STRUT [20,21] exhibit good performance on nonlinear problems. On some accuracy-restricted stiff problems, STRUT does not raise its order as rapidly as LSODE or DC03A, causing more steps to be taken. This is partially offset by less frequent linear algebra updates due to algorithms presented in [10].

8. Conclusions

When absolute stability regions are computed for any method, the argument of z which produces these plots should be displayed. This enables researchers to identify methods for which special techniques for order selection in the face of instability must be developed.

Adams method should be used to start BDF codes not only because they are more effective on the initial transients common to stiff problems [12,14], but also because the standard order-selection mechanism will reduce order appropriately on problems with eigenvalues near the imaginary axis as evidenced by the performance of LSODA on B5.

Order-selection strategies for the Adams methods and BDF methods should be different. If a BDF method of order three and above is used in a general ODE solver, then the order-selection mechanism must not rely on the relative size of error estimators or differences because the exit angles are not large enough when there are eigenvalues near the imaginary axis. The additional bias toward low order based on stepsize history over the past three steps is effective for the code STRUT because the stepsize is allowed to vary on each step. Codes that hold the stepsize constant for several steps would have $\beta_2 = 1$ and so no order-reduction bias results.

References

[1] A.R. Curtis, DC03A/AD stiff ODE software from Harwell Library, Comput. Sci. Systems Division, A.E.R.E., Harwell, Oxon.

[2] W.H. Enright, T.E. Hull and B. Lindberg, Comparing numerical methods for the solution of stiff systems of equations, *BIT* **15** (1975) 10–48.

[3] C.W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations* (Prentice-Hall, Englewood Cliffs, NJ, 1971).

[4] G. Hall and J.M. Watt, *Modern Numerical Methods for Ordinary Differential Equations* (Clarendon Press, Oxford, 1976).

[5] A.C. Hindmarsh, LSODE and LSODI, two new initial value ordinary differential equation solvers, *ACM/SIGNUM Newsletter* **15** (4) (1980) 10–11.

[6] F.T. Krogh, Predictor-corrector methods of high order with improved stability characteristics, *J. Assoc. Comput. Mach.* **13** (1966) 374–385.

[7] F.T. Krogh, A test for instability in the numerical solution of ordinary differential equations, *J. Assoc. Comput. Mach.* **14** (1967) 351–354.

[8] F.T. Krogh, The numerical integration of stiff differential equations, TRW Systems Group Report 99900-6573-R000, 1968.

[9] F.T. Krogh, Changing stepsize in the integration of differential equations using modified divided differences, *Proc. Conf. on Numerical Solution of Ordinary Differential Equations*, Lecture Notes in Math. **362** (Springer, New York, 1974).

[10] F.T. Krogh and K. Stewart, Asymptotic ($h \to \infty$) absolute stability for BDFs applied to stiff differential equations, *ACM Trans. Math. Software* **10** (1984) 45–57.

[11] J.D. Lambert, *Computational Methods in Ordinary Differential Equations* (Wiley, Chichester, 1973).

[12] L.R. Petzold, Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations, *SIAM J. Sci. Statist. Comput.* **4** (1983) 136–148.

[13] T. Rubner-Petersen, An efficient algorithm using backward time-scaled differences for solving stiff differential-algebraic systems, Institute of Circuit Theory and Telecommunication, Tech. Univ. Denmark, Lyngby, 1973.

[14] L.F. Shampine, Lipschitz constants and robust ODE codes, in: J.T. Oden, Ed., *Computational Methods in Nonlinear Mechanics* (North-Holland, Amsterdam, 1980) 424–449.

[15] L.F. Shampine, Evolution for a test set for stiff ODE solvers, *ACM Trans. Math. Software* **7** (1981) 409–420.

[16] L.F. Shampine and C.W. Gear, A user's view of solving stiff ordinary differential equations, *SIAM Rev.* **21** (1979) 1–17.

[17] L.F. Shampine and M.K. Gordon, *Computer Solution of Ordinary Differential Equations* (Freeman, San Francisco, CA, 1975).

[18] S. Skelboe, The control of order and steplength for backward differentiation formulas, *BIT* **17** (1977) 91–107.

[19] K. Stewart, Semi-implicit backward differentiation formulas, Ph.D. Diss., Dept. Mathematics, Univ. New Mexico, 1987.

[20] K. Stewart, Computational results for solving Burgers' equation using the method of lines, Tech. Report 1-89, Dept. Math. Sci., San Diego State Univ., 1989.

[21] K. Stewart and F.T. Krogh, Preliminary comparison test results for STRUT, LSODE and LSODA, Section 366 Internal Memorandum No. 499, Jet Propulsion Laboratory, Pasadena, CA, 1983.