ELSEVIER

Applied Numerical Mathematics 15 (1994) 449–463

APPLIED NUMERICAL MATHEMATICS

A stepsize control strategy for stiff systems of ordinary differential equations

Peter K. Moore [a,*], Linda R. Petzold [b,1]

[a] *Department of Mathematics, Tulane University, New Orleans, LA 70118, USA*
[b] *Department of Computer Science, University of Minnesota, Minneapolis, MN 55455, USA*

Abstract

In solving stiff systems of ordinary differential equations using BDF methods, Jacobians needed for quasi-Newton iteration are frequently computed using finite differences. Round-off errors in the finite-difference approximation can lead to Newton failures forcing the code to choose its time steps based on "stability" rather than accuracy considerations. When standard stepsize control is used, the code can experience thrashing which increases the total number of time steps, Jacobian evaluations, and function evaluations. In this paper we investigate this situation, explaining some surprising time step selection behavior produced by the standard control mechanism. A new control mechanism is proposed which attempts to find and use a "stability" stepsize. A comparison of the new strategy with the standard strategy and with two PI controllers introduced earlier is made using the stiff test set.

1. Introduction

In solving stiff systems of ordinary differential equations using BDF methods, Jacobians which are needed for the quasi-Newton iteration are often approximated using finite differences [1,2,9]. Smaller stepsizes than allowed by accuracy considerations may be needed to guarantee convergence of the Newton iteration due to round-off errors in the finite-difference Jacobians. The standard stepsize control mechanism, such as that used in DASSL [2], is

$$h_n = (TOL/EST_{n-1})^{1/p} h_{n-1}, \tag{1.1}$$

* Corresponding author.

[1] The work of this author was partially supported by ARO Contract Number DAAL03-89-C-0038 with the University of Minnesota AHPCRC, and by ARO Contract Number DAAL03-92-G-0247.

SSDI 0168-9274(94)00042-5

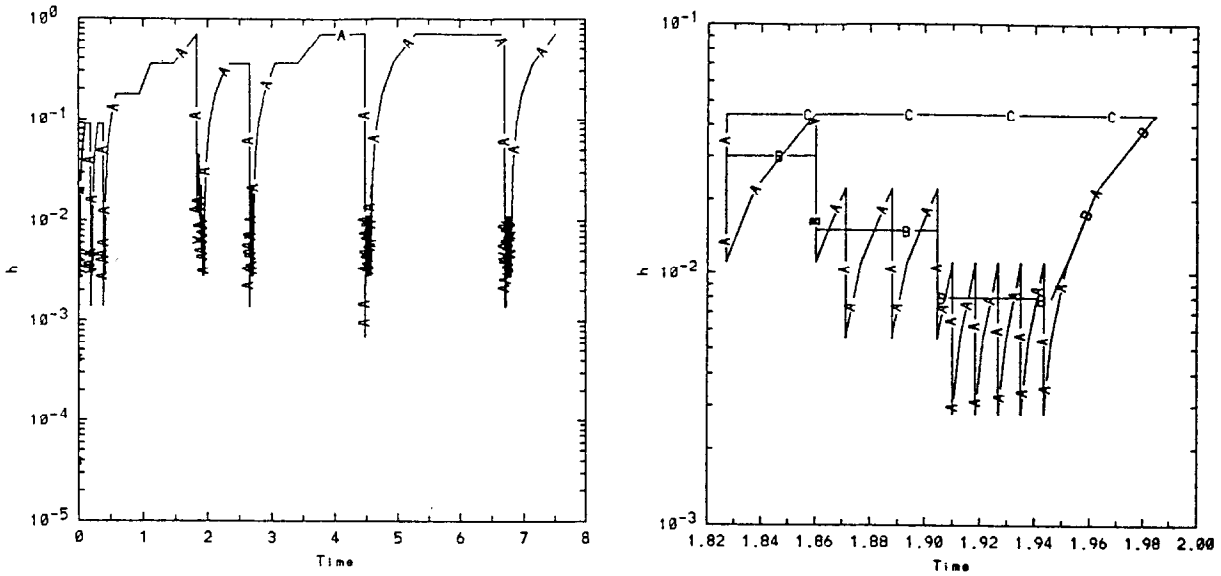

Fig. 1. Time steps used in solving problem D2 from the stiff test set [5] on $0 < t \leqslant 8$ (left) and on $1.83 \leqslant t \leqslant 1.98$ (right denoted by A) using DASSL with absolute and relative error tolerances of 0.01 and $\alpha = 0.5$. Estimates of the stepsize based on "stability", h_{stab} (right, denoted by B) and accuracy, h_{acc} (right, denoted by C).

where TOL is the user-prescribed tolerance, EST_{n-1} is the local error estimate computed at time t_{n-1}, and p is the method order. However, (1.1) is based solely on accuracy considerations. This can lead to highly oscillatory stepsize behavior (see Fig. 1).

Here we apply DASSL to the stiff test set [4,5] with an approximate Jacobian to simulate the effects of an inaccurate finite difference Jacobian. The stepsize behavior shown in Fig. 1 when DASSL is applied to problem D2 is typical. After periods of taking relatively small stepsizes the algorithm suddenly increases the stepsize by several orders of magnitude. It remains at this larger stepsize for several time steps, and then decreases the stepsize dramatically whereupon the process begins again. Although most of the time steps taken by DASSL are of the smaller size, the solution on most of the time interval is found using the larger time steps.

In Section 2 we analyze the behavior of DASSL on a simple linear system which leads to an understanding of the stepsize behavior discussed above. We also present a modification of the time step selection strategy used in DASSL based on the quasi-Newton algorithm of Dennis and Schnabel [3]. The revised strategy prevents the larger "anomalous" steps but leads to a much larger number of time steps with no significant improvement in accuracy. The stepsize controller of Gustafsson et al. [6], referred to henceforth as the PI controller I, is presented in Section 3 with a new interpretation. This controller was developed for explicit time integrators. Gustafsson [8] developed a PI controller (referred to herein as PI controller II) for implicit methods. We also discuss this controller in Section 3. In Section 4 we present a new control strategy and compare it with the standard stepsize controller and the PI controllers on the stiff test set [4,5]. Brief conclusions are given in Section 5.

2. Analysis of a stiff system

Consider first the standard test problem

$$y' = \lambda y, \qquad y(0) = 1, \tag{2.1}$$

where $\mathrm{Re}(\lambda) \leqslant 0$. Applying the backward Euler method together with quasi-Newton iteration yields

$$(1 - \alpha\lambda h)\Delta y_{n+1}^{i} = -y_{n+1}^{i-1}(1 - \lambda h) + y_n, \quad i \geqslant 1. \tag{2.2}$$

Typically, if an analytic Jacobian is used, $\alpha = 1$. To model the effects of an inaccurate matrix approximation, we choose α different from 1. After $k+1$ iterations of the quasi-Newton method we obtain

$$y_{n+1}^{k+1} = y_{n+1}^{0} \frac{(\alpha-1)^{k+1}(-\lambda h)^{k+1}}{(1-\alpha\lambda h)^{k+1}} + \frac{y_n}{1-\alpha\lambda h}\left[1 - \frac{(\alpha-1)\lambda h}{1-\alpha\lambda h} + \cdots + \frac{(\alpha-1)^{k}(-\lambda h)^{k}}{(1-\alpha\lambda h)^{k}}\right], \tag{2.3}$$

where y_{n+1}^{0} is the predicted solution. If $|\lambda h(\alpha-1)/(1-\alpha\lambda h)| < 1$, the quasi-Newton iterates converge to the true solution where the rate of convergence, ρ, is given by

$$\rho \equiv \frac{\left| y_{n+1}^{k+1} - y_n/(1-\lambda h)\right|}{\left| y_{n+1}^{k} - y_n/(1-\lambda h)\right|} = |\lambda h(\alpha-1)/(1-\alpha\lambda h)|. \tag{2.4}$$

In DASSL [2] the quasi-Newton iteration is said to converge if

$$\frac{\hat{\rho}}{1-\hat{\rho}}\left| y_{n+1}^{k+1} - y_{n+1}^{k}\right| < 0.33, \tag{2.5a}$$

where $\hat{\rho}$ is an approximation of the rate ρ given by

$$\hat{\rho} = \left(\left| y_{n+1}^{k+1} - y_{n+1}^{k}\right| / \left| y_{n+1}^{1} - y_{n+1}^{0}\right|\right)^{1/k}. \tag{2.5b}$$

Thus the number of iterations before convergence is determined by both the accuracy of the predictor and the rate of convergence $\hat{\rho}$.

For systems of equations, the analysis is complicated by the norm used. Although DASSL uses a weighted rms norm, herein we consider the l^2 norm which displays the same type of behavior. Consider the diagonal system

$$\boldsymbol{y}' = \boldsymbol{D}\boldsymbol{y}, \qquad \boldsymbol{y}(0) = 1, \tag{2.6}$$

where $\boldsymbol{D} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_m)$ with $\mathrm{Re}(\lambda_i) < 0$, $i = 1, 2, \ldots, m$. After two iterations the rate of convergence is given by

$$\frac{\left[\left(\rho_1\left(y_{n+1,1}^{0} - y_{n,1}/(1-\lambda_1 h)\right)\right)^2 + \cdots + \left(\rho_m\left(y_{n+1,m}^{0} - y_{n,m}/(1-\lambda_m h)\right)\right)^2\right]^{1/2}}{\left[\left(y_{n+1,1}^{0} - y_{n,1}/(1-\lambda_1 h)\right)^2 + \cdots + \left(y_{n+1,m}^{0} - y_{n,m}/(1-\lambda_m h)\right)^2\right]^{1/2}}, \tag{2.7}$$

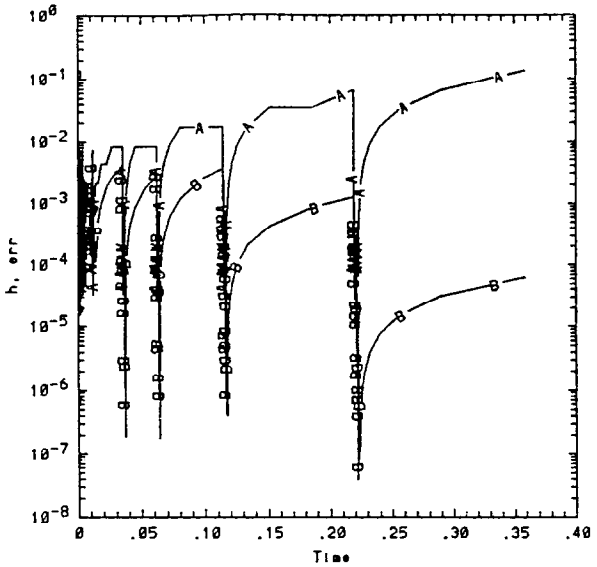

Fig. 2. Time steps (A) and error in y_{10} (B) in solving A4 from the stiff test set [5] on $0 < t \leqslant 0.4$ with DASSL with absolute and relative error tolerances of 0.01 and $\alpha = 0.5$.

where the second subscript indicates the component of the vector y and ρ_i is given by (2.4) with the appropriate λ_i. Unlike (2.1) the rate of convergence of (2.7) is not constant, but is instead determined by the stiffness (through ρ_i) and the accuracy of the predictor $(y^0_{n+1,i} - y_{n,i}/(1 - \lambda_i h))$. Thus if the stiff components are sufficiently more accurately predicted than the nonstiff components (which is likely since the stiff components change little from step-to-step) the rate of convergence will be controlled by the rate of convergence for the nonstiff components which have smaller rate constants. The stepsize controller may then wish to increase the stepsize (which may be low for the nonstiff components) until the errors in the stiff components are excited whereupon the stepsize undergoes a drastic reduction since the rate is now being determined by the stiff components.

That this actually happens can be seen by applying DASSL to problem A4 of the stiff test set [5] which has $\lambda_i = -i^5$, $i = 1, 2, \ldots, 10$. We solved this problem on $0 < t \leqslant 0.4$ with absolute and relative error tolerances of 0.01 and with $\alpha = 0.5$. Fig. 2 shows the time steps and the error in the stiffest component over the interval. When the stiff component becomes sufficiently accurate (after very small time steps), the time step increases rapidly, reaching a value controlled by the error in nonstiff components. Such large time steps excite errors in the stiff components until the rate of convergence becomes dominated by the stiff components and the time steps is drastically reduced, beginning the process again. Such behavior is also seen in non-diagonal systems as shown in Fig. 1.

The question arises, is it desirable to permit the large stepsizes. We observed that on some successful steps at the larger stepsizes $\| g(t, Y, Y') \|$ increased where Y is the BDF solution when we are solving

$$g(t, y, y') = 0. \tag{2.8}$$

One approach we implemented to correct this problem was to insist that $\|g(t, Y, Y')\|$ decrease before a step was converged. Specifically we required that on each Newton step

$$\frac{\|g(Y+\Delta Y)\|^2 - \|g(Y)\|^2}{-2\|g(Y)\|^2} \geqslant 0.05 \tag{2.9}$$

(cf. Dennis and Schnabel [3]), where ΔY is the quasi-Newton direction. Although using (2.9) did reduce the number and extent of the large stepsize regions, more time steps were used with no significant improvement in accuracy. Thus, it seems reasonable to allow the large steps and subsequently we do not use (2.9).

3. PI control

As was seen in Section 2 the standard stepsize control mechanism (1.1) leads to oscillatory time steps. The difficulty is finding a way to smooth out the selection of small time steps without eliminating the selection of the large time steps. One possible approach is to use PI controller I introduced by Gustafsson et al. [6–8] for explicit methods or PI controller II of Gustafsson [8] for implicit methods. In this section we present a new interpretation of these strategies along with some modifications for use in our situation.

PI controller I can be written in the form (with some modifications for maximum rate of stepsize increase and decrease) [7]

$$h_n = (TQL/EST_{n-1})^{K_I}(EST_{n-2}/EST_{n-1})^{K_P}h_{n-1}, \tag{3.1}$$

where EST_{n-2} and EST_{n-1} are estimates of the local truncation error at time t_{n-2} and t_{n-1}, respectively, and K_I and K_P are parameters whose values depend only on whether a step is successful or not. Values for K_I and K_P are given in [6] and [7] although they differ slightly. For our purposes it is important to note first that in the case of a rejected step $K_I = 1/p$ and $K_P = 0$. Thus, when a step is rejected the standard controller is used. Second, in the case of an accepted step $K_I + K_P \approx 1/p$.

We can rewrite (3.1) as

$$h_n = (EST_{n-2}/TOL)^{K_P}(TOL/EST_{n-1})^{(K_P+K_I)}h_{n-1}. \tag{3.2}$$

Assuming $K_I + K_P = 1/p$ and using

$$h_{\mathrm{acc},n-1} = (TOL/EST_{n-1})^{1/p}h_{n-1}, \tag{3.3a}$$

we obtain

$$h_n = (EST_{n-2}/TOL)^{K_P}h_{\mathrm{acc},n-1}, \tag{3.3b}$$

where $h_{\mathrm{acc},n-1}$ represents the stepsize based on the local truncation error that could have been taken at time t_{n-2}. We note that $h_{\mathrm{acc},n-1}$ also represents the stepsize based on accuracy that is typically used for the next time step $[t_{n-1}, t_n]$. Now, since (3.3a) holds at time t_{n-2} we obtain

$$h_n = (h_{n-2}/h_{\mathrm{acc},n-2})^{K_G}h_{\mathrm{acc},n-1}. \tag{3.4}$$

where $K_G = K_P/(K_P + K_I)$. Thus the new stepsize is chosen to be the stepsize based on accuracy multiplied by a factor that represents the ratio of the actual stepsize to accuracy stepsize we could have chosen on the previous step. If on the last step the accuracy and actual time step were the same, the accuracy stepsize is used on the present step. Otherwise (the previous stepsize can never be larger than the previous stepsize based on accuracy), if the previous stepsize was much smaller than the previous stepsize based on accuracy only a fraction of the accuracy stepsize is used.

A similar analysis shows that PI controller II has the form

$$h_n = (h_{n-1}/h_{n-2})(EST_{n-2}/EST_{n-1})^{1/p} h_{\mathrm{acc},n-1}, \tag{3.5}$$

if two or more successive accepted time steps have been taken (otherwise the standard controller is used). Now the accuracy time step $h_{\mathrm{acc},n-1}$ is multiplied by a factor representing the ratio of successive accepted time steps and factor proportional to the ratio of successive error on those steps.

The version of PI algorithm I we used in our testing consists of three cases.

(1) If the present step is rejected due to the error test, set $h_n = h_{\mathrm{acc},n-1}$ and $K_G = K_{LO}$.
(2) If the present step is rejected due to Newton divergence, set $h_n = h_{n-1}/4$ and $K_G = K_{HI}$.
(3) If the present step is accepted update $K_G = \max(\mathrm{fac} * K_G, K_{LO})$ if the previous step was not a Newton failure. Set $h_n = \min(1, (h_{n-2}/h_{\mathrm{acc},n-2})^{K_G}) h_{\mathrm{acc},n-1}$.

Here $\mathrm{fac} = 0.9$, $K_{LO} = 0.5$, and $K_{HI} = 0.7$ are fixed parameters. This differs slightly from the approach taken by Gustafsson [6,8], since we allow K_G to vary. We found varying K_G resulted in slightly better performance over fixed K_G. The algorithm for PI controller II is similar with (3.4) replaced by (3.5). Thus, for PI controller II, there is no K_G (and hence no K_{LO} and K_{HI}). If Newton's method diverges we always revert to the standard controller after the next step for which Newton converges.

Gustafsson [8] offers some additional ideas for PI controller II. He has a stepsize algorithm for the case of successive stepsize failure due to error control. In our situation, however, we encounter successive stepsize failure due to divergence in Newton's method so we did not incorporate this heuristic in our algorithm. In certain cases of Newton divergence he computes a second, "stability" stepsize which is based on the size of the norm of the Jacobian. Since we are also interested in solving differential-algebraic equations, we are very reluctant to use a scale-dependent quantity in our algorithm so we have neglected this feature.

4. A new controller

For reasons that will become clear from the examples in this section neither the standard controller nor the PI controllers possess the desired stepsize behavior. Using the analysis from Section 2 we present a new controller which we refer to as the STAB controller. We then present a numerical comparison of the standard, PI, and STAB controllers applied to the stiff test set [4,5].

From our observations in Section 2 we desire a controller that interferes with the standard controller as little as possible. Our goal was to smooth out the time step selection strategy only

when the code is thrashing due to Newton convergence difficulties. When the code is able to use larger stepsizes because of a good predictor for the stiff components, we want to let it do this because this is where it makes most of its progress. Additionally, as indicated in Section 2, no accuracy is lost in accepting the large steps.

We begin with the observation that there are two important time step sizes, an accuracy size h_{acc} and a stability size h_{stab}, where here, stability refers to the convergence of Newton's method. Normally h_{acc} is smaller than h_{stab} for BDF methods but when the Jacobian is poorly approximated the reverse can occur. In the graph on the right side of Fig. 1, the peaks in curve A indicate stepsizes chosen by accuracy but which caused the quasi-Newton iteration to diverge. Thus the best guess at h_{acc} is represented by the peaks, although it may be quite a bit larger (curve C on the right, Fig. 1). After each peak two successful, smaller steps are taken. The value h_{stab} is approximated by curve B in Fig. 1. Our controller seeks to detect when h_{stab} is smaller than h_{acc} and then makes two attempts at finding h_{stab}. The algorithm then limits the time step size for 10 steps to 0.87 h_{stab} after which it reverts to the standard controller. The value 0.87 is chosen to reduce the number of step failures and to reduce the number of Newton iterates required for convergence. If the time step used was h_{stab}, Newton make take several steps to converge due to a larger rate and a poorer predictor.

The new controller is invoked only when the Newton iteration fails to converge (as long as the convergence is not due to a singular Jacobian), i.e., when the criterion (2.5a) fails and when the last successful time step h_{n-1} is smaller than the first failed (Newton) step h_n. Two attempts are made at finding h_{stab}. After the first Newton failure, if $h_{n-1}/h_n \geqslant 0.8$, $h_n = 0.87h_{n-1}$ and the time step is not allowed to become larger than this value for 10 time steps (of course it can become smaller due to subsequent Newton failures or error failure). If $h_{n-1}/h_n < 0.8$, then $h_n = 0.8h_{n-1} + 0.2h_n$. Now, however, the stepsize is allowed to increase, but at a reduced maximum rate of 1.18. If Newton fails for a second (but not second consecutive) time within the 10 step limit, $h_n = 0.87h_{n-1}$ and no increase above this value is allowed for 10 steps. Consecutive Newton failures result in the algorithm reverting to the standard controller since we no longer seem to have a good approximation to h_{stab}. We limit our controller to 10 steps so that larger stepsize increases are allowed from time to time which should preserve the desirable property of the standard controller.

We solved the stiff set using DASSL with the three controllers and absolute and relative error tolerances of 10^{-k}, $k = 2, 3, \ldots, 6$ and $\alpha = 0.5$. Tables 1–4 contain the number of time steps used (including successful and unsuccessful time steps), the number of Jacobian evaluations, the number of function evaluations, and the l^2 error at the final time by each of the four algorithms with tolerances of 0.0001, respectively. None of the algorithms was able to solve F1 or F4 in 10000 time steps and only PI controller I was able to do so for F5 with this poor approximation to the iteration matrix. In almost all cases the STAB controller outperforms the standard controller. In a number of cases (A1, B2, B4, B5, C3, C5, D6, E1, E4) the standard controller has a smaller error than the STAB controller (cf. Table 4). However, except for cases C5 and D6 the STAB controller is superior with respect to the other measures. For problems A2 and D3 where the standard controller appears to outperform the STAB controller, the error produced using the STAB controller was at least a factor of three smaller. Also note the anomalous behavior of the standard controller on F3. PI controller II is slightly better than PI controller I. The STAB controller is more efficient than the PI controllers for a significant

Table 1
The number of time steps needed by DASSL with the standard controller, the STAB controller, and the PI controllers in solving the problems of the stiff test set [4,5] with tolerances 0.0001 and $\alpha = 0.5$. None of the three algorithms solved F1 or F4 in fewer than 10000 steps and only PI controller I was able to solve F5 in fewer than 10000 steps. The standard controller and PI controllers I and II failed to reach the final time in 15000 steps for problem F3

Stiff set problem	Number of time steps			
	Standard controller	STAB controller	PI controller I	PI controller II
A1	367	217	286	321
A2	1546	1627	1990	816
A3	2446	1467	1946	525
A4	1469	1417	1201	652
B1	1154	892	2578	3173
B2	110	106	120	104
B3	145	120	132	119
B4	277	184	360	355
B5	1633	1071	1413	780
C1	347	238	153	134
C2	328	242	258	270
C3	322	152	257	304
C4	348	281	420	474
C5	272	233	393	493
D1	2099	621	1928	2042
D2	2767	1576	2408	1694
D3	376	449	358	346
D4	154	497	99	87
D5	144	182	131	160
D6	517	604	476	511
E1	62	40	73	74
E2	156	156	871	981
E3	2121	1063	1973	1618
E4	1298	972	1339	789
E5	19	19	19	19
F2	159	135	112	96
F3	15002	152	15002	15002

number of the test problems. Although PI controller II appears to perform better than the STAB controller on the early problems A2, A3, and A4, the STAB controller produced solutions with errors at least a factor of 10 smaller than PI controller II. On problems D4 and D5 the STAB controller is significantly more accurate (cf. Table 4). The results at this tolerance were typical of the performance of the control algorithms at the other tolerances.

The data in Table 2 clearly indicate that the STAB controller significantly reduces the number of Jacobian evaluations over the standard controller. It also frequently uses fewer such evaluations than either PI controller I or II. The improvement in the number of function evaluations (cf. Table 3) is not as dramatic but still noticeable. From Table 4 we see that the standard and STAB controllers consistently produce more accurate solutions (with respect to the l^2 error at the final time) than do the PI controllers.

Table 2
The number of Jacobian evaluations needed by DASSL with the standard controller, the STAB controller, and the PI controllers in solving the problems of the stiff test set [4,5] with tolerances 0.0001 and $\alpha = 0.5$

Stiff set problem	Number of Jacobian evaluations			
	Standard controller	STAB controller	PI controller I	PI controller II
A1	303	55	176	248
A2	1429	383	1813	655
A3	2242	410	1603	139
A4	1333	306	1000	454
B1	1154	892	266	851
B2	35	20	18	17
B3	59	24	18	24
B4	139	29	159	220
B5	945	153	858	294
C1	270	51	26	26
C2	254	63	151	192
C3	258	39	147	239
C4	285	86	94	417
C5	196	71	249	427
D1	2021	168	1577	1903
D2	2685	388	2194	1523
D3	294	119	97	190
D4	152	153	89	86
D5	118	72	476	135
D6	515	230	57	511
E1	46	19	33	66
E2	22	22	57	276
E3	1998	325	1693	1454
E4	1058	339	948	450
E5	19	19	19	19
F2	128	41	31	23
F3	15002	152	15002	15002

In Table 5 we have listed the number of Jacobian evaluations, function evaluations, and errors (in the l^2 norm) for several cases. The STAB controller is clearly more efficient than the standard controller in almost every case. For several problems it uses a factor of 6 fewer Jacobian evaluations and less than half the number of function evaluations while obtaining a smaller error. Even in the cases where it uses more function evaluations and time steps, such as D4, its solution is more accurate. PI controller II is generally superior to PI controller I. The PI controllers have, in general, larger errors and are often between the STAB and standard controllers in the other measures.

Fig. 3 presents the time steps used in the standard controller and the STAB controller for $0 < t \leqslant 8$ with tolerances of 0.00001 and $\alpha = 0.5$. Clearly the STAB controller produces a smoother time step history than the standard controller.

The only problem which presented any difficulty to DASSL with the standard controller and $\alpha = 1$ was F5. We solved F5 with the standard, the STAB, and the PI controllers with $\alpha = 1$ and

Table 3
The number of function evaluations needed by DASSL with the standard controller, the STAB controller, and the PI controllers in solving the problems of the stiff test set [4,5] with tolerances 0.0001 and $\alpha = 0.5$

Stiff set problem	Number of function evaluations			
	Standard controller	STAB controller	PI controller I	PI controller II
A1	960	563	803	818
A2	3560	3811	4779	974
A3	2242	410	4515	1074
A4	3590	3388	2896	1460
B1	2547	2130	5190	6379
B2	274	241	241	209
B3	351	295	265	252
B4	754	426	1021	897
B5	3639	2527	3987	1841
C1	845	590	307	271
C2	795	603	606	617
C3	792	406	596	689
C4	891	713	959	1086
C5	740	607	904	1159
D1	5600	1624	5574	5516
D2	6408	3724	5824	4001
D3	899	1048	721	717
D4	342	1138	260	214
D5	302	431	265	348
D6	1069	1433	1132	1099
E1	128	86	150	157
E2	315	315	1743	1964
E3	5338	2686	5498	4226
E4	3041	2477	2871	1656
E5	39	39	39	39
F2	366	333	240	198
F3	25393	355	25292	25119

with both analytic and finite difference Jacobians for tolerances of 10^{-k}, $k = 2, 3, \ldots, 6$. The results shown in Table 6 (analytic Jacobians) and Table 7 (finite-difference Jacobians) indicate that the STAB controller enhances the performance of DASSL in both cases. PI controller II has surprisingly poor performance for this problem.

5. Conclusions

When finite-difference approximations to the Jacobian are used in stiff solvers such as DASSL, thrashing of the time step can occur because the accuracy stepsize exceeds the stepsize required for convergence of the quasi-Newton iteration. After damping the stiff components at small stepsizes, such algorithms using the standard stepsize control mechanism are able to take larger time steps based on the error in the nonstiff components. However, the stiff components

Table 4
The l^2 error at the final time of the solution computed by DASSL with the standard controller, the STAB controller, and the PI controllers in solving the problems of the stiff test set [4,5] with tolerances 0.0001 and $\alpha = 0.5$. The standard controller and PI controllers I and II failed to reach the final time in 15,000 steps for problem F3; hence, no error was computed

Stiff set problem	l^2 error at the final time			
	Standard controller	STAB controller	PI controller I	PI controller II
A1	0.73×10^{-6}	0.24×10^{-5}	0.32×10^{-4}	0.14×10^{-4}
A2	0.10×10^{-4}	0.62×10^{-7}	0.12×10^{-4}	0.23×10^{-4}
A3	0.79×10^{-2}	0.48×10^{-3}	0.13×10^{-1}	0.18×10^{-1}
A4	0.20×10^{-2}	0.20×10^{-5}	0.21×10^{-2}	0.39×10^{-2}
B1	0.18×10^{-7}	0.18×10^{-7}	0.18×10^{-7}	0.18×10^{-7}
B2	0.59×10^{-4}	0.22×10^{-3}	0.53×10^{-3}	0.73×10^{-3}
B3	0.76×10^{-4}	0.80×10^{-4}	0.54×10^{-3}	0.71×10^{-3}
B4	0.18×10^{-4}	0.51×10^{-4}	0.16×10^{-4}	0.14×10^{-4}
B5	0.29×10^{-3}	0.42×10^{-3}	0.39×10^{-3}	0.28×10^{-2}
C1	0.45×10^{-8}	0.34×10^{-8}	0.74×10^{-5}	0.19×10^{-5}
C2	0.64×10^{-8}	0.20×10^{-8}	0.27×10^{-7}	0.14×10^{-7}
C3	0.26×10^{-8}	0.11×10^{-4}	0.41×10^{-7}	0.22×10^{-7}
C4	0.80×10^{-3}	0.40×10^{-5}	0.32×10^{-2}	0.77×10^{-5}
C5	0.28×10^{-3}	0.63×10^{-1}	0.50×10^{-3}	0.90×10^{-2}
D1	0.11×10^{-1}	0.81×10^{-3}	0.12	0.11
D2	0.39×10^{-1}	0.20×10^{-2}	0.32×10^{-1}	0.57×10^{-1}
D3	0.44×10^{-7}	0.97×10^{-13}	0.52×10^{-9}	0.59×10^{-8}
D4	0.55×10^{-3}	0.64×10^{-3}	0.32×10^{-2}	0.31×10^{-2}
D5	0.23×10^{-2}	0.79×10^{-3}	0.19×10^{-2}	0.17×10^{-2}
D6	0.21×10^{-4}	0.68×10^{-4}	0.15×10^{-5}	0.18×10^{-5}
E1	0.11×10^{-6}	0.22×10^{-6}	0.45×10^{-5}	0.17×10^{-5}
E2	0.81×10^{-2}	0.81×10^{-2}	0.76×10^{-1}	0.72×10^{-1}
E3	0.35×10^{-1}	0.20×10^{-2}	0.52×10^{-1}	0.67×10^{-1}
E4	0.90×10^{-5}	0.12×10^{-4}	0.17×10^{-6}	0.28×10^{-4}
E5	0.74×10^{-5}	0.74×10^{-5}	0.74×10^{-5}	0.74×10^{-5}
F2	0.17×10^{-3}	0.31×10^{-4}	0.22×10^{-2}	0.25×10^{-2}
F3	–	0.73×10^{-8}	–	–

then become excited resulting in drastic decreases in the stepsize. We analyzed a simple linear system to explain this stepsize phenomena. To smooth out the smaller stepsizes we tried modified versions of two PI controllers proposed by Gustafsson et al. [6–8] for explicit Runge–Kutta methods and implicit Runge–Kutta methods, respectively, which we reinterpreted to our situation. A new controller was also proposed which attempts to find and use a "stability" stepsize. In comparing the four controllers on the stiff test set with altered Jacobian we found the new controller superior in almost every case. In fact for a number of cases it offers dramatic improvement over the standard and the PI controllers.

Although most of the test results of Section 4 were somewhat artificial the new controller does produce significantly better results especially in terms of the number of Jacobian evaluations. This continued to be true even when analytic and finite difference Jacobians were

Table 5
The number of Jacobian evaluations (JACS), function evaluations (FNS), time steps (STEPS), and error in the l^2 norm for selected problems from the stiff test [5] using DASSL with standard, STAB, and PI controllers with $\alpha = 0.5$

Stiff set problem	TOL	Standard controller	STAB controller	PI controller I	PI controller II	
A2	10^{-5}	3349	697	2361	584	JACS
		8649	6423	7182	2668	FNS
		3595	2632	2918	1190	STEPS
		0.42×10^{-5}	0.11×10^{-6}	0.68×10^{-5}	0.13×10^{-4}	l^2 ERROR
	10^{-6}	8241	1077	2679	868	JACS
		22570	10237	10037	5631	FNS
		8630	4110	3958	2709	STEPS
		0.19×10^{-5}	0.54×10^{-7}	0.25×10^{-5}	0.50×10^{-5}	l^2 ERROR
A3	10^{-5}	6002	905	3666	479	JACS
		15458	8554	11065	3375	FNS
		4375	2933	4312	1582	STEPS
		0.36×10^{-2}	0.43×10^{-3}	0.35×10^{-2}	0.60×10^{-2}	l^2 ERROR
	10^{-6}	9901	1207	1906	3232	JACS
		25928	11786	10087	12022	FNS
		10595	4594	4109	5166	STEPS
		0.96×10^{-3}	0.14×10^{-3}	0.20×10^{-2}	0.18×10^{-2}	l^2 ERROR
D2	10^{-5}	5223	704	3092	2302	JACS
		13226	6431	10014	7122	FNS
		5377	2607	3788	2884	STEPS
		0.11×10^{-1}	0.12×10^{-2}	0.12×10^{-1}	0.19×10^{-1}	l^2 ERROR
	10^{-6}	9123	967	4519	3588	JACS
		24408	9806	18768	14089	FNS
		9403	3761	6762	5547	STEPS
		0.28×10^{-2}	0.29×10^{-3}	0.44×10^{-2}	0.65×10^{-2}	l^2 ERROR
D4	10^{-5}	488	434	1029	585	JACS
		1129	3520	2500	1392	FNS
		506	1538	1066	606	STEPS
		0.19×10^{-3}	0.12×10^{-3}	0.64×10^{-3}	0.92×10^{-3}	l^2 ERROR
	10^{-6}	1998	589	209	359	JACS
		4859	5353	1058	1436	FNS
		2129	2330	482	649	STEPS
		0.18×10^{-3}	0.35×10^{-4}	0.30×10^{-3}	0.28×10^{-3}	l^2 ERROR
E3	10^{-6}	5403	521	3393	1400	JACS
		14102	5905	12103	9960	FNS
		5609	2120	4293	4293	STEPS
		0.21×10^{-4}	0.28×10^{-4}	0.31×10^{-3}	0.93×10^{-3}	l^2 ERROR

used for problem F5. Smoothing out the smaller stepsizes also seems to be beneficial for the error. At the suggestion of the referee we made an initial attempt at a method that combined our STAB controller and PI controller II. The results were not favorable and further study is

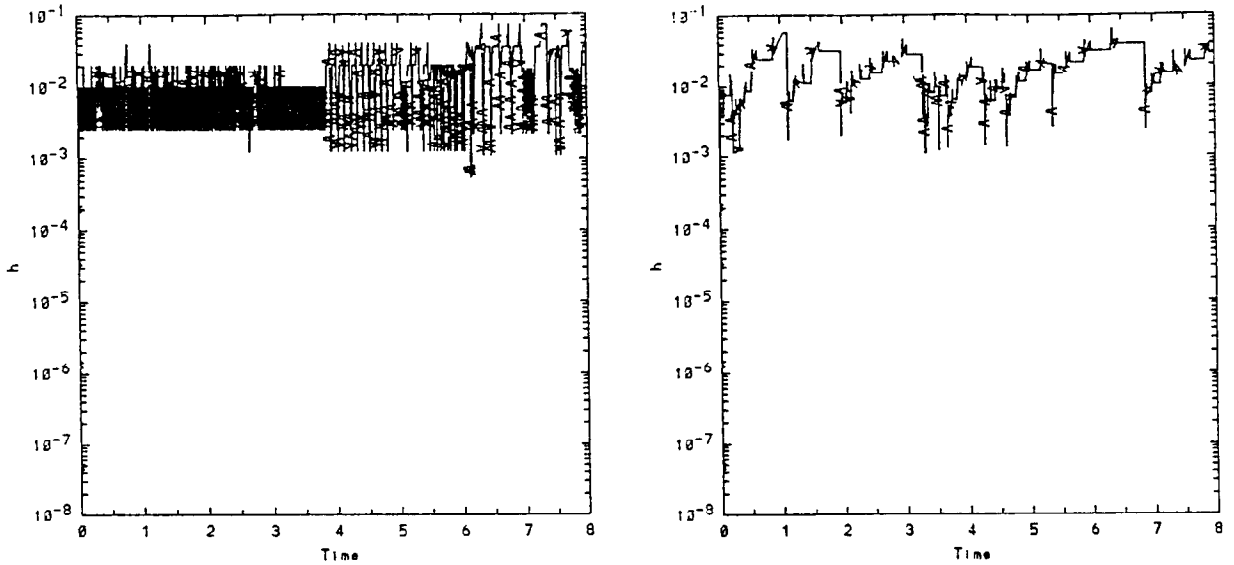

Fig. 3. Time steps used in solving problem D2 from the stiff test set [4] on $0 < t \leqslant 8$ using DASSL with the standard controller (left) and the STAB controller (right) and with absolute and relative error tolerances of 0.00001 and $\alpha = 0.5$.

Table 6
The number of Jacobian evaluations (JACS), function evaluations (FNS), and time steps (STEPS) needed to solve problem F5 from the stiff test set with DASSL and analytic Jacobians using the standard, STAB, and PI controllers

TOL	Analytic Jacobian				
	Standard controller	STAB controller	PI controller I	PI controller II	
10^{-2}	49	28	51	47	JACS
	82	83	86	79	FNS
	49	44	52	47	STEPS
10^{-3}	39	51	84	48	JACS
	71	170	142	88	FNS
	39	87	85	48	STEPS
10^{-4}	356	100	313	857	JACS
	637	314	545	1303	FNS
	370	145	329	863	STEPS
10^{-5}	2575	119	2137	7298	JACS
	4138	435	3495	11509	FNS
	2604	195	2180	7355	STEPS
10^{-6}	9296	3547	12576	14037	JACS
	14798	12166	20006	22379	FNS
	9339	5675	12710	14305	STEPS

Table 7
The number of Jacobian evaluations (JACS), function evaluations (FNS), and time steps (STEPS) needed to solve problem F5 from the stiff test set with DASSL and finite-difference Jacobians using the standard, STAB, and PI controllers

TOL	Finite-difference Jacobian				
	Standard controller	STAB controller	PI controller I	PI controller II	
10^{-2}	41	34	43	35	JACS
	202	226	248	202	FNS
	35	47	44	35	STEPS
10^{-3}	74	48	39	117	JACS
	418	346	230	655	FNS
	74	74	40	117	STEPS
10^{-4}	411	146	38	617	JACS
	2328	1043	261	3443	FNS
	425	197	54	622	STEPS
10^{-5}	2805	897	3202	4616	JACS
	15783	6764	17954	25741	FNS
	2834	1443	3245	4667	STEPS
10^{-6}	2683	1383	4655	12513	JACS
	15160	10362	26317	70046	FNS
	2726	2269	4789	12739	STEPS

needed. More testing needs to be done to verify the usefulness of the algorithm in a wider setting, when used to solve partial differential equations by the method-of-lines and differential-algebraic equations.

Acknowledgement

The authors would like to thank the referee for several helpful suggestions that improved an earlier version of this paper.

References

[1] M. Berzins, R.M. Furzeland and P.M. Dew, Software tools for time-dependent differential equations, in: A.J. Osiadacz, ed., *Simulation and Optimization of Large Systems* (Oxford University Press, Oxford, 1988).

[2] K.E. Brenan, S.L. Campbell and L.R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations* (North-Holland, New York, 1989).

[3] J.E. Dennis and R.B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations* (Prentice-Hall, Englewood Cliffs, NJ, 1983).

[4] W.H. Enright and T.E. Hull, Comparing numerical methods for the solution of stiff systems of ODEs arising in chemistry, in: L. Lapidus and W.E. Schiesser, eds., *Numerical Methods for Differential Systems, Recent Developments in Algorithms, Software and Applications* (Academic Press, New York, 1976), 45–66.

[5] W.H. Enright, T.E. Hull and B. Lindberg, Comparing numerical methods for stiff systems of ODEs, *BIT* 15 (1975) 10–48.
[6] K. Gustafsson, M. Lundh and G. Soderlind, A PI stepsize control for the numerical solution of ordinary differential equations, *BIT* 28 (1988) 270–287.
[7] K. Gustafsson, Control theoretic techniques for stepsize selection in explicit Runge–Kutta methods, *ACM Trans. Math. Software* 17 (1991) 533–554.
[8] K. Gustafsson, Control of error and convergence in ODE solvers, Ph.D. Thesis, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden (1992).
[9] A.C. Hindmarsh, ODEPACK, a systematized collection of ODE solvers, in: R.S. Stepleman et al., eds., *Scientific Computing* (North-Holland, Amsterdam, 1983) 55–64.