Lawrence Berkeley National Laboratory

Title
A high-order method for stiff boundary value problems

Permalink
https://escholarship.org/uc/item/60k9g6d1

Journal
SIAM Journal on Scientific Computing, 8(September 1987)

ISSN
1064-8275

Authors
Brown, David
Lorenz, Jens

Publication Date
1987-09-05

Peer reviewed

A HIGH-ORDER METHOD FOR STIFF BOUNDARY VALUE PROBLEMS WITH TURNING POINTS*

DAVID L. BROWN† AND JENS LORENZ‡

Abstract. This paper describes some high-order collocation-like methods for the numerical solution of stiff boundary-value problems with turning points. The presentation concentrates on the implementation of these methods in conjunction with the implementation of the a priori mesh construction algorithm introduced by Kreiss, Nichols and Brown [SIAM J. Numer. Anal., 23 (1986), pp. 325–368] for such problems. Numerical examples are given showing the high accuracy which can be obtained in solving the boundary value problem for singularly perturbed ordinary differential equations with turning points.

Key words. stiff boundary value problems, turning points, collocation, mesh construction

AMS(MOS) subject classifications. 34A50, 34B05, 34E20, 65L10, 65L50

Introduction. In recent papers [6], [12], Kreiss, Nichols and Brown introduced a new numerical method for solving boundary value problems for stiff linear systems of ordinary differential equations (ODEs). The method they presented is unique among methods for boundary value problems in that it provides an a priori procedure for constructing a difference mesh which will be appropriate for resolving rapidly varying features of the solutions of these problems, such as those that arise in connection with "turning points." Once the mesh has been constructed, the differential equations are approximated using a combination of symmetric and unsymmetric two-point difference formulas in a way which essentially gives second order accuracy. In this paper, we show how high-order collocation methods can be used in a natural way in conjunction with the ideas presented in [6], [12] in order to increase the accuracy of the computed solutions. Numerical experiments show that extremely accurate results can be obtained in this way.

It should be emphasized that the use of collocation methods for solving stiff boundary value problems is not a new idea (see e.g. the papers of Ascher and Weiss [2], [3], and Ringhofer [13]). Symmetric collocation formulas, such as those used in [2], [3] have been shown to work well for stiff boundary value problems if the mesh is carefully constructed so as to resolve boundary layers, and if no turning points are present. (A certain eigenvalue condition must be satisfied as well (see [1], [10], [11]).) For problems with turning points, however, numerical experiments have shown that there can be difficulties with applying symmetric schemes directly (see e.g. [12]). For this reason, as suggested by the ideas in [12], we describe a combination of symmetric and unsymmetric formulas which can be used for a discretization of the problem. In order to apply these formulas the system of ODEs must be transformed to a form where growing, moderate, and decaying modes are essentially decoupled. This transformation is done automatically in our code, and in fact done simultaneously with the a priori mesh construction. For certain problems without turning points in which fast-growing, moderate, and fast-decaying modes are decoupled in the given differential

---

† Los Alamos National Laboratory, Center for Nonlinear Studies and Computing Division, Los Alamos, New Mexico 87545.

‡ Applied Mathematics, California Institute of Technology, Pasadena, California 91125.

system Ringhofer [13] has suggested and analyzed collocation methods similar to those we present here.

In § 1 we discuss the derivation of the collocation formulas that we use. Section 2 discusses their application to systems of ODEs. In § 3 we briefly discuss the local elimination procedure by which the interior collocation points are eliminated. In order to apply our formulas to systems of ODEs, each equation in the system is considered as a scalar equation, and depending on the size of the real part of the diagonal element in the coefficient matrix, either a symmetric formula or the appropriate unsymmetric formula is applied. The choice of which formula to apply depends on accuracy considerations; this is discussed in § 4. The a priori construction of the difference mesh is done in such a way that the resulting computed solution is guaranteed to be resolved by that mesh. A discussion of the mesh construction procedure is given in § 5. In § 6 we present numerical examples which demonstrate the accuracy of the collocation formulas when applied to a system of two second-order singularly perturbed ODEs for which the exact solution is known.

1. Derivation of the difference formulas for scalar equations. Consider the linear scalar ODE given by

$$(1.1) \qquad \frac{dy(x)}{dx} = a(x)y(x) + f(x), \qquad 0 \leqq x \leqq 1.$$

This ODE is to be approximated using collocation formulas. Let $m \geqq 2$ be the number of nodes on the normalized interval $[0, 1]$: $0 = \rho_0 < \rho_1 < \cdots < \rho_m = 1$. The ρ_j are assumed to be placed symmetrically with respect to $\rho = \frac{1}{2}$; for example, the ρ_j can denote the Lobatto points on $[0, 1]$ or the Radau points on $[0, 1)$ together with the Radau points on $(0, 1]$. For the time being, we assume that mesh points $0 = x_1 < x_2 < \cdots < x_N = 1$ have been constructed already, and we consider a fixed mesh interval $x_\nu \leqq x \leqq x_{\nu+1} = x_\nu + h$. The nodes

$$0 = \rho_0 < \rho_1 < \cdots < \rho_m = 1, \qquad m \geqq 2 \text{ fixed},$$

in the normalized interval $0 \leqq \rho \leqq 1$ lead on each mesh interval $[x_\nu, x_{\nu+1}]$ to auxiliary points $x_{\nu j} = x_\nu + h\rho_j$, $j = 0, \cdots, m$, the collocation points.

Ultimately, the formulas derived here will be applied to systems of ODEs in the following way: Each differential equation in the system is considered separately, and based on information about the corresponding diagonal element of the coefficient matrix, either a symmetric or an appropriate unsymmetric formula is selected and applied to that scalar equation. Collocation formulas of the type we will describe lead to two-point difference formulas of a high order of accuracy. The high accuracy is obtained by using values of the coefficients of the ODE at auxiliary points between the actual meshpoints as well as the values at the meshpoints. Since the differential equations for each scalar component of the unknown dependent variable in the system of ODEs involves values of the other components at these auxiliary points as well, it is clear that the same set of collocation points must be used for each scalar equation even if the same difference formulas are not used. We are thus led to difference formulas similar to those studied by Ringhofer [13] for problems without turning points.

Three formulas will be described:
(a) A symmetric formula, suitable for $h|a| = O(1)$:
(b) A right-biased formula, suitable for $\text{Re } a \ll -1$:
(c) A left-biased formula, suitable for $\text{Re } a \gg 1$.

(a) *The symmetric formula.* For $j = 1, \cdots, m$ there are uniquely determined weights w_{jk}^0 ($k = 0, 1, \cdots, m$) such that for all polynomials $p(s)$ of degree $\leq m$

$$\int_0^{\rho_j} p(s) \, ds = \sum_{k=0}^m w_{jk}^0 p(\rho_k).$$

(These weights can be expressed as integrals of the Lagrange interpolation polynomials (see e.g. [16]).) Now let $x_{\nu+1} = x_\nu + h$, $x_{\nu j} = x_\nu + h\rho_j$, $j = 0, \cdots, m$. Integrating (1.1), we have

$$y(x_{\nu j}) - y(x_\nu) = h \int_0^{\rho_j} (ay + f)(x_\nu + hs) \, ds,$$

which motivates the difference equations

(1.2) $u_{\nu j} - u_\nu = h \sum_{k=0}^m w_{jk}^0 [a(x_{\nu k}) u_{\nu k} + f(x_{\nu k})], \qquad j = 1, \cdots, m.$

This is a system of m linear equations for the $m + 1$ unknowns $u_{\nu j}$, $j = 0, \cdots, m$.

(b) *The right-biased formula.* For $j = 1, \cdots, m$ there are uniquely determined weights w_{jk}^R ($k = 1, \cdots, m$) such that for all polynomials $p(s)$ of degree $\leq m - 1$

(1.3) $\int_0^{\rho_j} p(s) \, ds = \sum_{k=1}^m w_{jk}^R p(\rho_k).$

In contrast to the symmetric formula, the node $\rho_0 = 0$ is not allowed on the right-hand side of (1.3). As above, this motivates the difference equations

$$u_{\nu j} - u_\nu = h \sum_{k=1}^m w_{jk}^R [a(x_{\nu k}) u_{\nu k} + f(x_{\nu k})], \qquad j = 1, \cdots, m.$$

Again we have a system of m equations for $m + 1$ unknowns.

(c) *The left-biased formula.* For $j = 1, \cdots, m$ there are uniquely determined weights w_{jk}^L ($k = 0, \cdots, m - 1$) such that for all polynomials $p(s)$ of degree $\leq m - 1$

$$\int_{\rho_{j-1}}^1 p(s) \, ds = \sum_{k=0}^{m-1} w_{jk}^L p(\rho_k).$$

Integrating (1.1), we obtain

$$y(x_{\nu+1}) - y(x_{\nu j-1}) = h \int_{\rho_{j-1}}^1 (ay + f)(x_\nu + hs) \, ds$$

which motivates the difference equations

(1.4) $u_{\nu+1} - u_{\nu j-1} = h \sum_{k=0}^{m-1} w_{jk}^L [a(x_{\nu k}) u_{\nu k} + f(x_{\nu k})], \qquad j = 1, \cdots, m.$

Summary. With suitable definition for the coefficients e_{jk}^* and w_{jk}^*, each of these three sets of differences formulas can be written in the form

(1.5) $\sum_{k=0}^m e_{jk}^* u_{\nu k} = h \sum_{k=0}^m w_{jk}^* [a(x_{\nu k}) u_{\nu k} + f(x_{\nu k})], \qquad j = 1, \cdots, m.$

These difference formulas are equivalent to collocation or implicit Runge–Kutta type formulas (cf. [16]).

Remark. The local truncation error of the symmetric formula is $O(h^{m+1} \| y^{(m+2)} \|)$ whereas the one-sided formulas lead to a local truncation error of the order

$O(h^m \|y^{(m+1)}\|)$. This holds for all choices of nodes ρ_j. In particular, we have studied two choices for the set of nodes $\{\rho_j\}$, namely the Lobatto points, and the Radau points used twice (as explained below). As is well known, the special choice of Lobatto points leads to an error $O(h^{2m})$ at the mesh points for the symmetric formula. This favorable "superconvergence" result is lost for the one-sided formulas, and indeed we were not able to make use of superconvergence results for the mixed schemes which we describe in the next section. If one uses the Radau points on $(0, 1]$ as nodes $\rho_1, \cdots, \rho_m$, then superconvergence will be obtained for the right-biased formula. For symmetry reasons, the Radau points on $[0, 1)$ should be included as nodes also if both fast-growing and fast-decaying modes are present in the differential equation. However, for the examples tested these schemes with "Radau points twice" do not compare favorably with the schemes using Lobatto points. (See Table 3 in § 7 for an example.) It should also be emphasized that the concept "order of convergence" must be used with care in the present context of stiff equations since one is not studying the case $h \to 0$. The estimates of the local truncation error are relevant, however, since by the mesh construction employed the solution will be smooth with respect to the underlying grid.

2. Application of the collocation formulas to systems of equations. Consider now the system of ODEs given by

$$(2.1) \qquad \frac{dy(x)}{dx} = A(x)y(x) + f(x), \qquad 0 \le x \le 1$$

where $A(x) \in \mathbf{C}^{n,n}$, $y(x), f(x) \in \mathbf{C}^n$. It follows from the results of [12] that if the matrix $A(x)$ is essentially diagonally dominated, the pth row of (2.1) can be discretized using one of the three formulas introduced in § 1, and the choice of which formula to use for each row is determined by looking at the size and sign of the real part of the diagonal element of that row, Re $a_{pp}(x)$. Specifically, denoting by $z^{(i)}$ the ith component of a vector $z \in \mathbf{C}^n$, we can look at the scalar equation

$$(2.2) \qquad \frac{dy^{(p)}(x)}{dx} - a_{pp}(x)y^{(p)}(x) = \sum_{i \neq p} a_{pi}(x)y^{(i)}(x) + f^{(p)}(x)$$

as if the variables $y^{(i)}$, $i \neq p$ are known functions when we make the choice of which formula to use. Applying (1.5) to (2.2) leads to the system of equations

$$(2.3) \qquad \sum_{k=0}^{m} e_{jk}^{(p)} u_{\nu k}^{(p)} = h \sum_{k=0}^{m} w_{jk}^{(p)} \left[a_{pp}(x_{\nu k})u_{\nu k}^{(p)} + \sum_{i \neq p} a_{pi}(x_{\nu k})u_{\nu k}^{(i)} + f^{(p)}(x_{\nu k}) \right],$$

$$p = 1, \cdots, n, \quad j = 1, \cdots, m.$$

Here the coefficients $w_{jk}^{(p)}$, $j = 1, \cdots, m$, $k = 0, \cdots, m$ can take on the values w_{jk}^0, w_{jk}^R or w_{jk}^L and similarly for the $e_{jk}^{(p)}$. With the notation $\mathbf{u}_\nu := (u_{\nu 0}^T, \ldots, u_{\nu m}^T)^T$ (2.3) can be written as a system of mn linear equations for $\mathbf{u}_\nu$ of the form

$$(2.4) \qquad M\mathbf{u}_\nu = \mathbf{r}_\nu$$

where $M \in \mathbf{C}^{(m+1)n, mn}$, $\mathbf{r}_\nu \in \mathbf{C}^{mn}$. Identifying $u_{\nu m} = u_{(\nu+1)0}$, and including n boundary conditions for (2.1) we arrive at a system of mnN linear equations for a total of mnN unknowns, where N is the number of meshpoints x_ν to be used. As is well known, by a process of local elimination, this can be reduced to a banded system of Nn equations for the approximate solution of (2.1) at the meshpoints x_ν, $\nu = 1, \cdots, N$. Our implementation of this reduction is discussed in the next section.

As discussed in [12], the assumption that the system (2.1) is in essentially diagonally dominant form initially is usually too restrictive. As in that paper, it is generally

required that the system be transformed to a form in which the diagonal elements essentially dominate the rows of the matrix $A(x)$. In general, then, it is required that the system (2.1) be discretized in the transformed form.

Define $A_\nu = A(x_\nu)$ and assume that T_ν, $T_{\nu+1}$ are determined such that $T_\nu A_\nu T_\nu^{-1} = D_\nu$, $T_{\nu+1} A_{\nu+1} T_{\nu+1}^{-1} = D_{\nu+1}$ are essentially diagonally dominated. (It is assumed that $|T_{\nu+1} - T_\nu|$ is not large.) Set

$$T(x) = T_\nu + \frac{(x - x_\nu)}{h}(T_{\nu+1} - T_\nu), \qquad x_\nu \leqq x \leqq x_{\nu+1}$$

and introduce the transformed variable

$$\tilde{y}(x) = T(x)y(x).$$

From (2.1) we obtain

(2.5) $$\tilde{y}' = Ty' + T'y = (TAT^{-1} + T'T^{-1})\tilde{y} + Tf.$$

By our assumptions, the matrix

$$TAT^{-1} + T'T^{-1}$$

is essentially diagonally dominated: thus the equation for $\tilde{y}$ can be treated as described above. For technical reasons, we prefer to discretize directly in terms of y instead of $\tilde{y}$ and use the following equation:

$$(Ty)' = (TA + T')y + Tf.$$

With the notation

$$B(x) = T(x)A(x) + T'(x), \qquad g(x) = T(x)f(x),$$

we have

(2.6) $$(Ty)' = By + g.$$

For each component p, $1 \leqq p \leqq n$, of (2.6), the appropriate collocation formula (symmetric, right-biased or left-biased) can be determined by looking at the diagonal element of the pth row of the matrix TAT^{-1} in (2.5). Once this choice has been made, the scalar equation

(2.7) $$\frac{d(Ty)^{(p)}(x)}{dx} = (By)^{(p)}(x) + g^{(p)}(x)$$

can be discretized, and equations analogous to (2.4) can be obtained. With appropriate definitions of the matrices E_{jk}, W_{jk}, they are given by

(2.8) $$\sum_{k=0}^{m} E_{jk} T(x_{\nu k}) u_{\nu k} = h \sum_{k=0}^{m} W_{jk} [B(x_{\nu k}) u_{\nu k} + g(x_{\nu k})], \qquad j = 1, \cdots, m.$$

This is again a system of mn equations for $(m+1)n$ unknowns of the form (2.4).

3. Local elimination. For efficient numerical computation using these formulas, it is desirable to reduce the system (2.4) to one in which only $u_{\nu 0}$ and $u_{\nu m}$ appear:

(3.1) $$Bu_{\nu m} + Cu_{\nu 0} = r$$

with $B, C \in \mathbf{C}^{n,n}$, $r \in \mathbf{C}^n$. To do this, we eliminate the unknowns $u_{\nu 1}, \cdots, u_{\nu, m-1}$ locally.

After an interchange of columns in the matrix M, we write the system (2.4) as

$$(3.2) \qquad (P, M_0, M_m) \begin{pmatrix} \hat{\mathbf{u}}_\nu \\ u_{\nu 0} \\ u_{\nu m} \end{pmatrix} = \tilde{\mathbf{r}}$$

where $\hat{\mathbf{u}}_\nu = (u_{\nu 1}^T, \cdots, u_{\nu,m-1}^T)^T$, P is $mn \times (m-1)n$, M_0 and M_m are $mn \times n$. We assume now that at least $(m-1)n$ rows of the mn rows of P are linearly independent, i.e., that P has rank $(m-1)n$. Applying a Gaussian elimination process with column pivoting we transform the system (3.2) to an equivalent system

$$(3.3) \qquad \begin{bmatrix} U \mid M_{12} \mid M_{13} \\ O \mid C \mid B \end{bmatrix} \begin{bmatrix} \hat{\mathbf{u}} \\ u_{\nu 0} \\ u_{\nu m} \end{bmatrix} = \hat{\mathbf{r}}.$$

The desired matrices B, C and the n-vector r in (3.1) can now be read off from the last n equations of (3.3).

4. The choice of collocation formula to apply. So far, we have introduced the collocation formulas that we use, but we have not discussed how the choice is made of which formula to apply to each component of the ODE system. Consider again the scalar equation (1.1) with $a(x) \equiv a$ a constant and the forcing function $f(x)$ set to zero. Assume for the moment that $\lambda := \operatorname{Re} a \leq 0$, so that either the centered or right-biased formulas will be appropriate to apply. After elimination of the auxiliary variables $u_{\nu j}$, $j = 1, \cdots, m-1$, each collocation formula leads to an equation of the form

$$(4.1) \qquad u_{\nu+1} = G(ha)u_\nu$$

where $h = x_{\nu+1} - x_\nu$ is the local meshwidth. By definition, $G(z)$ is the growth function of the method. From consistency considerations, it is clear that $G(z)$ must be an approximation to e^z. Under reasonable assumptions on the ρ_j, it can be shown that for fixed m, and small $|z|$, the growth functions $G_0(z)$ of the symmetric formulas introduced in § 1 will be better approximations to e^z than will be the growth functions $G_R(z)$ for the right-biased formulas. However, as $z \to -\infty$, we have that

$$|G_0(z) - e^z| \to 1,$$

while

$$|G_R(z) - e^z| \to 0.$$

There is therefore a largest negative number $z = -z_C(m)$ for which

$$|G_0(z) - e^z| \leq |G_R(z) - e^z|.$$

Table 1 lists the values of $z_C(m)$ for the Lobatto points and for the Radau points used twice (cf. § 1). Note that as the number of collocation points increases, the symmetric formula can be used for larger and larger values of $h \operatorname{Re} a$. This is clear from looking at the graphical representations of the growth functions given in Figs. 1 and 2 for the Lobatto points.

Since a desirable property of a method is that its growth function approximate e^z as well as possible, it is natural to switch from the symmetric formula to the right-biased formula if $h \operatorname{Re} a$ is less than $-z_C$. Correspondingly, for the case $h \operatorname{Re} a > 0$, the switch should be made to the left-biased formula when $h \operatorname{Re} a$ exceeds z_C.

For the variable coefficient case, the situation is slightly more complicated, since $a(x_\nu)$ and $a(x_{\nu+1})$ may not have the same properties. Similarly to [9], we choose the

TABLE 1

Values of z_C for different numbers of collocation points. (ncol = $m+1$ is the number of collocation points per mesh interval.)

Lobatto points

ncol	z_C
2	1.00
3	2.00
4	3.60
5	3.77
6	5.29
7	5.56
8	7.05
9	7.35

Radau points twice

4	3.74
6	5.03
8	6.27

appropriate method for the interval $[x_\nu, x_{\nu+1}]$ as follows. Let $\alpha_\nu := h\ \mathrm{Re}\ a(x_\nu)$, $\alpha_{\nu+1} := h\ \mathrm{Re}\ a(x_\nu)$; then there are four cases to consider:

(1) (Moderate case) If $|\alpha_\nu| \leqq z_C$ and $|\alpha_{\nu+1}| \leqq z_C$, then use the symmetric formula.

(2) (Decaying case) If $\alpha_\nu < -z_C$, and $\alpha_{\nu+1} \leqq 0$, or $\alpha_{\nu+1} < -z_C$ and $\alpha_\nu \leqq 0$, then use the right-biased formula.

(3) (Growing case) If $\alpha_\nu > z_C$ and $\alpha_{\nu+1} \geqq 0$, or $\alpha_{\nu+1} > z_C$ and $\alpha_\nu \geqq 0$, take the left-biased formula.

(4) If α_ν and $\alpha_{\nu+1}$ have opposite signs, and either $|\alpha_\nu|$ or $|\alpha_{\nu+1}|$ exceeds z_C there is no natural choice of formula. If this case occurs, Kreiss and Kreiss [9] require that
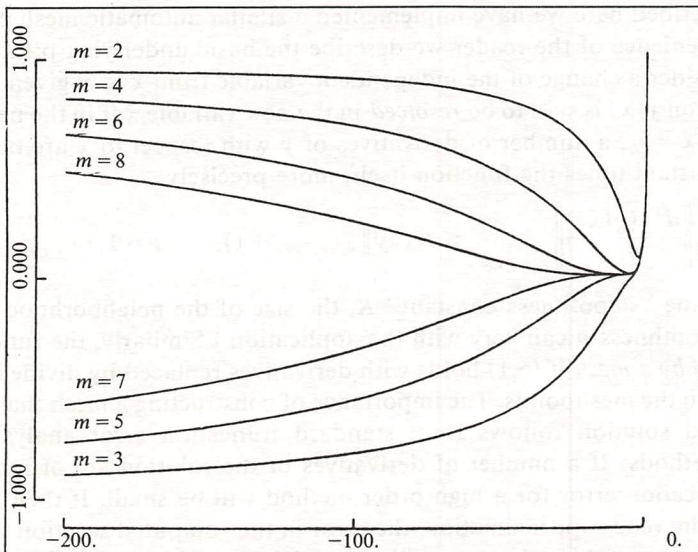

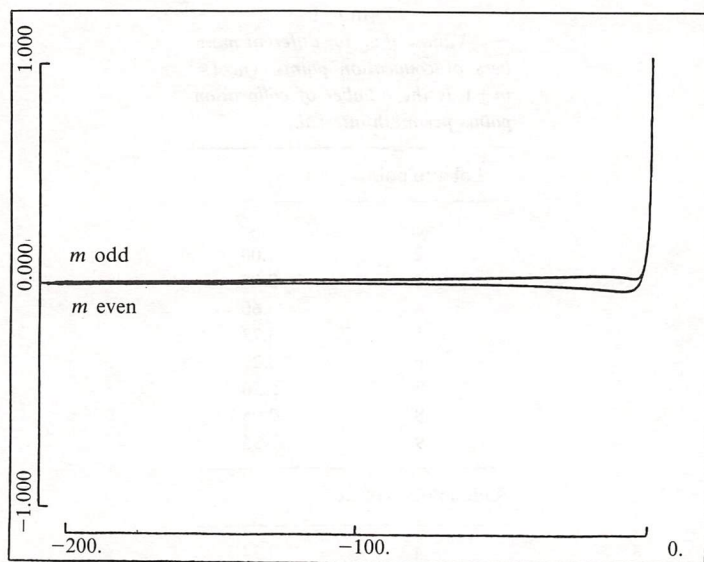FIG. 1. *Growth functions for Lobatto points, symmetric formulas.*

FIG. 2. *Growth functions for Lobatto points, right-biased formulas.*

a meshpoint be added within the interval $[x_\nu, x_{\nu+1}]$ such that on each of the resulting two intervals, one of the above 3 cases will apply. In practice, the mesh construction algorithm of Kreiss, Nichols and Brown [12] always leads to a mesh in which this fourth case never arises. This is basically because this situation would mean that the coefficient $a(x)$ was not logarithmically smooth with respect to the constructed mesh, which is not allowed (cf. § 5 and [6], [12]).

5. Basic principles of a priori mesh construction. An important feature of the method described in [12] for solving stiff two-point boundary value problems was the automatic construction of a mesh which will resolve the solution to be computed. In the code described here we have implemented a similar automatic mesh construction. For the convenience of the reader we describe the basic underlying principles in this section. Consider a change of the independent variable from x to $\tilde{x}$ given by $x = g(\tilde{x})$. Then a function $y(x)$ is said to be *resolved* in the new variable $\tilde{x}$ if in the neighborhood of any point $\tilde{x} = \tilde{x}_0$, a number of derivatives of y with respect to $\tilde{x}$ are bounded by a moderate constant times the function itself, more precisely,

$$(5.1) \qquad \left\| \frac{d^\nu y(g(\tilde{x}))}{d\tilde{x}^\nu} \right\|_{\tilde{x}_0-\varepsilon, \tilde{x}_0+\varepsilon} \leq K (\|y\|_{\tilde{x}_0-\varepsilon, \tilde{x}_0+\varepsilon} + 1), \qquad \nu = 1, \cdots, p.$$

(The size of the "smoothness constant" K, the size of the neighborhood 2ε, and the degree of smoothness p can very with the application.) Similarly, the function is said to be *resolved by a mesh* if (5.1) holds with derivatives replaced by divided differences with respect to the meshpoints. The importance of constructing a mesh that will resolve the computed solution follows from standard truncation error analysis for finite difference methods: If a number of derivatives of the solution are of moderate size, then the truncation error for a high order method will be small. It then follows that for a stable difference approximation, the error in the computed solution will be small also as long as the problem is not too ill posed. We conclude that (5.1) is the essential condition for adjusting the mesh. If (5.1) holds, we take a uniform mesh with respect

to the variable $\tilde{x}$; this will then lead in general to a nonuniform mesh with respect to the variable x.

In [12] it is shown how the stretching function g can be found using only information about the coefficients $A(x)$ and $f(x)$ of the system of ODEs (1.1). Since this information is available a priori for linear systems, a corresponding mesh can be constructed *before* the difference method is applied to the system of ODEs. This is an alternative approach to an adaptive mesh construction strategy.

The analysis in [12] begins with a study of systems of ODEs that are in essentially diagonally dominant form. Ultimately it will be required to make a transformation of a given system of ODEs to this form. However, for the purposes of presenting the following definition and theorem, we consider for the moment a system (2.1) for which such a transformation has already been made.

DEFINITION (Kreiss, Nichols and Brown [12]). A matrix function

$$A(x) := \begin{pmatrix} A_{11}(x) & A_{12}(x) \\ A_{21}(x) & A_{22}(x) \end{pmatrix} \in \mathbf{C}^{n,n}$$

(where $A_{11} \in \mathbf{C}^{n_+ + n_-, n_+ + n_-}$, $A_{22} \in \mathbf{C}^{n_0, n_0}$, A_{12}, A_{21} of appropriate dimensions, $n_+ + n_- + n_0 =: n$) is said to be *essentially diagonally dominated* if the elements a_{ij} of A_{11} satisfy

(5.2a) $\operatorname{Re} a_{ii} < 0$, $i = 1, \cdots, n_-$, $\operatorname{Re} a_{ii} > 0$, $i = n_- + 1, \cdots, n_- + n_+$,

(5.2b) $|\Lambda_R^{-1} B_{11}| \leq 1 - \delta$, $B_{11} := A_{11} - \Lambda_{11}$

for some $0 < \delta < 1$

(5.2c) $|\Lambda_R^{-1} \Lambda_I| \leq \rho$

for some constant $\rho = O(1)$, and

(5.2d) $|\Lambda_R^{-1} A_{12}| < 1$, $|A_{2j}| \leq K_0$, $j = 1, 2$,

where $\Lambda_R := \operatorname{diag} \{\operatorname{Re} a_{ii}\}_{i=1}^{n_+ + n_-}$, $\Lambda_I := \operatorname{diag} \{\operatorname{Im} a_{ii}\}_{i=1}^{n_+ + n_-}$, $\Lambda_{11} := \Lambda_R + i\Lambda_I$ and K_0 is a constant of moderate size (more about this later).[1] (Note that in this definition there is no restriction on the size of the elements of A_{11}; this is the stiff part of the matrix.)

If $A(x)$ is essentially diagonally dominated, then the system of ODEs (2.1) is said to be in *essentially diagonally dominant (EDD) form*. The relevant theorem given in [12] that leads to a mesh construction algorithm is repeated here.

THEOREM. *Consider the system of ODEs (2.1) on $0 < x < c$ and partition the forcing function $f(x)$ in a corresponding way to the matrix A, i.e.,*

$$f(x) = \begin{pmatrix} f^I(x) \\ f^{II}(x) \end{pmatrix}, \quad f^I \in \mathbf{C}^{n_+ + n_-}, \quad f^{II} \in \mathbf{C}^{n_0}.$$

If (2.1) is in EDD form and there are constants K_1 and K_2 of moderate size such that

(5.3a) $|\Lambda_{11}^{-1} d^\nu \Lambda_{11} / dx^\nu| \leq K_1$ (*eigenvalue smoothness*),

(5.3b) $|\Lambda_{11}^{-1} d^\nu (B_{11} | A_{12}) / dx^\nu| \leq K_1$ (*off-diagonal element smoothness*),

(5.3c) $|d^\nu A_{2j} / dx^\nu| \leq K_1$, $j = 1, 2$ (*smoothness of the $O(1)$ part*),

(5.3d) $|(|f^I| + 1)^{-1} d^\nu f^I / dx^\nu| \leq K_1$
             $|d^\nu f^{II} / dx^\nu| \leq K_1$ (*smoothness of RHS*),

---

[1] If $y \in \mathbf{C}^n$, then $|y|$ denotes its maximum norm. If $A \in \mathbf{C}^{n,n}$, then $|A| := \max_{y \neq 0} (|Ay| / |y|)$.

$\nu = 1, \cdots, p,$ and

(5.4)
$$|a_{ii}(0)| \leqq K_2, \quad i = 1, \cdots, n_-$$
$$|a_{ii}(c)| \leqq K_2, \quad i = n_- + 1, \cdots, n_+ + n_-$$
(boundary layer resolution),

then there is a constant $K = K(K_0, K_1, K_2, \rho, \delta)$ of moderate size such that the derivatives of the solution can be estimated by (5.1) with $x \equiv \tilde{x}$.

In a few places above, we have introduced "constants of moderate size." Let us explain here what this means in terms of stiff boundary value problems. The systems of ODEs (2.1) that are of interest here are ones where $|A_{11}| \gg |A_{22}|$ and hence can have solutions which vary on two (or more) scales. It is typical for these problems that the solution will be quite smooth everywhere except for isolated narrow regions where the variation is quite rapid; the latter regions are called internal or boundary layers. An example of a nonuniform mesh that would resolve such a solution is one which is quite fine in the layers, but relatively coarse in the regions where the solution is smoothly varying. Suppose h is the maximum meshsize used in the regions of smooth variation. Then typically in these regions $h|A_{11}| \gg 1$ (which is why such a problem is called "stiff") while $h|A_{22}| \ll 1$. With these ideas in mind, a *constant of moderate size*, $C > 0$, can be defined as one for which $hC \ll 1$.

6. Construction of the mesh and transformation. In this section we will discuss the practical application of the theorem of § 5. In general, of course, we cannot restrict our consideration to problems which are in EDD form. An important feature of the stiff BVP method is the ability to automatically construct a (piecewise smooth) transformation matrix $T(x)$ that transforms the differential equation to EDD form. To be precise, in our code we transform to essentially *block* diagonally dominant form with at most three diagonal blocks. This transformation is somewhat easier to implement than a transformation to EDD form and seems to be sufficient for most applications. (More refined implementations are currently under investigation.)

We use two different procedures for the construction of $T(x)$. The first one, which we call **transform-from-scratch** in the pseudocode below, transforms a given fixed matrix A to a certain block form. The second procedure, which we call **update-transformation**, is used to obtain a transformation matrix at neighboring meshpoints and can only be used as long as the appropriate block structure does not change as a function of x. The way in which these two routines are used within the mesh construction process is described below.

First consider the problem of transforming a given fixed matrix $A = A(x)$. (At this point and below we assume that the matrix A is real.) Bavely and Stewart [5] have described a method for doing this which is similar to the one we will describe here. Using the QR method, the matrix A is first transformed to quasi-upper triangular form. (Quasi-upper triangular form is a real block upper triangular form with diagonal blocksize at most two. If two-by-two blocks occur, they contain complex conjugate eigenvalues of the matrix.) The QR method is applied in such a way that the eigenvalues occur in the diagonal blocks ordered according to the size of their *real parts*. In [15] Stewart gives a code that arranges the eigenvalues in order of decreasing absolute value along the diagonal. We have modified this code to order the eigenvalues strictly in terms of their real parts. After this first transformation we group the eigenvalues into three groups: Let $\tilde{h}$ denote the local meshsize. $\mathbf{G}_+$ contains those eigenvalues λ of $A = A(x)$ with $\tilde{h} \operatorname{Re} \lambda > z_C$. $\mathbf{G}_-$ contains those with $\tilde{h} \operatorname{Re} \lambda < -z_C$, and $\mathbf{G}_0$ contains

those with $\tilde{h}|\text{Re }\lambda| \leqq z_C$. The transformed matrix can be partitioned accordingly:

$$\begin{pmatrix} A_+ & A_{12} & A_{13} \\ 0 & A_0 & A_{23} \\ 0 & 0 & A_- \end{pmatrix},$$

i.e., A_+ has the elements $\lambda \in G_+$ as eigenvalues, etc. As a second transformation, the nonzero off-diagonal blocks A_{12}, A_{13}, A_{23} are transformed to zero by elimination (or equivalently, by solving *linear* matrix Riccati equations [4], [12]) leaving a matrix of the form

$$\tilde{A} = TAT^{-1} = \begin{pmatrix} A_+ & 0 & 0 \\ 0 & A_0 & 0 \\ 0 & 0 & A_- \end{pmatrix}.$$

This procedure is summarized in the following segment of pseudocode:

PROCEDURE **Transform-from-scratch**
(**input:** A; **output** A_b, T_b, *blockstructure*, *eigenvalues*)
{*comments*: This procedure is used to transform a general matrix A to block-diagonal form A_b via a transformation with T_b. The eigenvalues in the blocks are also ordered appropriately}
Upper-triangularize-and-order-eigenvalues (input: A; **output:** A_u, T_u, *eigenvalues*);
{*comments*: $A_u = T_u A T_u^{-1}$ is now in quasi-upper-triangular form}
Choose-diagonal-blocks (input: *eigenvalues*, h; **output:** *blockstructure*);
Eliminate-off-diagonal-blocks (input: A_u, *blockstructure*, T_u; **output:** A_b, T_b, *success*); If (*success = false*) Stop;
{*comments*: When the matrix is quasi-upper-triangular, the diagonal block elimination can only fail if the blockstructure is incorrectly chosen}
    END PROCEDURE;

Remarks. For most applications, the block matrix $\tilde{A}$ constructed above seems to be close enough to EDD form though the occurrence of large outer diagonal entries is possible. By an additional transformation using a diagonal matrix D these large outer diagonal entries could be scaled down. In general, $|D| + |D^{-1}|$ would be large, which would increase $|T| + |T^{-1}|$. In practice, we have not used such diagonal scalings.

In principle, the procedure described above could be used repeatedly at different locations x to construct a transformation function $T(x)$. However, this has not only the disadvantage of being somewhat expensive, but also the smoothness of such a transformation as a function of x is not guaranteed. We thus use an updating procedure to compute a transformation $T(x+h)$ from the given transformation $T(x)$ whenever possible. In most cases, the matrix $\hat{A} = T(x)A(x+h)T(x)^{-1}$ is a perturbation of the matrix $\tilde{A} = T(x)A(x)T(x)^{-1}$, and so $T(x+h)$ can be determined from $\hat{A}$ without first transforming to upper triangular form. The details involve the solution of several quadratic matrix Riccati equations which is done using a simple iterative technique discussed in [12] and also mentioned by Stewart [14]. The technique has the feature that as a byproduct, the eigenvalues of the blocks are computed, so the eigenstructure of the matrix can be monitored. The updating procedure is summarized in the following piece of pseudocode:

PROCEDURE **Update-transformation**
(**input:** A, T_b, *blockstructure*; **output:** A_{b_new}, S_b, S_b, *success*)
{*comments*: This procedure first pretransforms the matrix A with the similarity

transformation T_b and then by operating on this pretransformed matrix, computes a transformation S_b which blocks the (original) matrix A.}

$A_{b\_old} = T_b A T_b^{-1}$;

Eliminate-off-diagonal-blocks (**input**: A_{b_old}, *blockstructure*; **output**: T_e, *success*);

If (*success* = *true*) Then

$S_b = T_e T_b$;

$A_{b\_new} = T_e A_{b\_old} T_e^{-1}$;

End;

ΕΝD PROCEDURE;

Remarks. The variable *success* in this procedure is included to indicate that the iterative technique for eliminating the off-diagonal blocks can fail to converge. From practical experience, it is always possible to remedy this situation, by decreasing h, the distance to the next point at which the transformation is to be constructed. Since these are effects that are relevant to the overall method, the nonconvergence of this iteration is used as another way to monitor these changes. Note that, in general, the three diagonal blocks of the transformed matrix resulting from the updating procedure are full. This is in constrast to the diagonal blocks of the transformed matrix resulting from the procedure **transform-from-scratch** which are quasi-upper-triangular. This difference is another reason for not using the full transformation procedure at every meshpoint. If the blocked matrix is forced to be upper-triangular, then necessarily at least one eigenvector of the matrix is being computed. The smoothness properties of an eigenvector of a matrix are in general worse than the smoothness properties of the corresponding eigenvalue. On the other hand, one can show that a transformation that simply decouples blocks containing well-separated groups of eigenvalues can be chosen to depend smoothly on the matrix elements (cf. Kato [8]). Since information about eigenvectors is not needed for our purposes, it is therefore more reasonable to construct a transformation function $T(x)$ that does not attempt to upper-triangularize the diagonal blocks of the matrix. The only reason for first making a full transformation of the matrix to upper triangular form is to make sure that the eigenvalues appear in the correct order and end up in the correct blocks of the matrix. The method described in "**transform-from-scratch**" is therefore used only at a few isolated points $x = x_j$, while in the intervals between these points, the method described in "**update-transformation**" is used.

Having described the two procedures for blocking a matrix, we now describe some details of the mesh construction. In order to use the theorem of § 5, the coefficient matrix $A(x)$ must be blocked in the manner described above. Then the conditions that are checked are the condition that the transformation matrix be resolved by the mesh, the smoothness of the transformed right-hand side (5.3d), the boundary layer resolution conditions (5.4) and the condition that the imaginary part of the large eigenvalues not be too large (5.2c). If the latter condition is violated, the mesh is refined until these eigenvalues become $O(1)$ with respect to the mesh, and then become part of the block A_0. The constants K_0, K_1 and K_2 are taken as $O(1)$ constants in the numerical code. Typical values are $K_0 = \frac{1}{2}$, $K_1 = \frac{1}{2}$, and $K_2 = 0.4$. In the code we have implemented, the option is also given to check the smoothness of the off-diagonal elements of the transformed matrix (conditions (5.3b) and (5.3c)), but in practice we have never found it necessary to do this.

Suppose the interval on which the problem (2.1) is to be solved is given by $0 < x < 1$. Then the mesh construction procedure starts by first setting up a preliminary mesh which is uniform except near the boundaries where an exponentially stretched mesh

is put in if either of the conditions (5.4) would be violated otherwise. This preliminary mesh is used as a guide to the maximum meshsize that will be permitted at any location on the final mesh that will be constructed. Then beginning at $x = 0$ and proceeding towards $x = 1$, the transformation matrix T is constructed at every fourth meshpoint; the condition that the mesh resolve the piecewise linear interpolation of T at the meshpoints, and the remaining conditions (5.2c) and (5.3), are checked. If none of the conditions is ever violated, then the final computational mesh will be the same as the guide mesh. If any of the conditions are violated, then meshpoints will be spaced closer together until the conditions are locally not violated any more. The mesh construction procedure will, however, always try to use the maximum allowable meshsize which does not result in the violation of any of the conditions (5.2c) and (5.3) and which does not have any local meshsize larger than that indicated by the guide mesh. This somewhat complicated procedure is outlined in the diagram of Fig. 3.
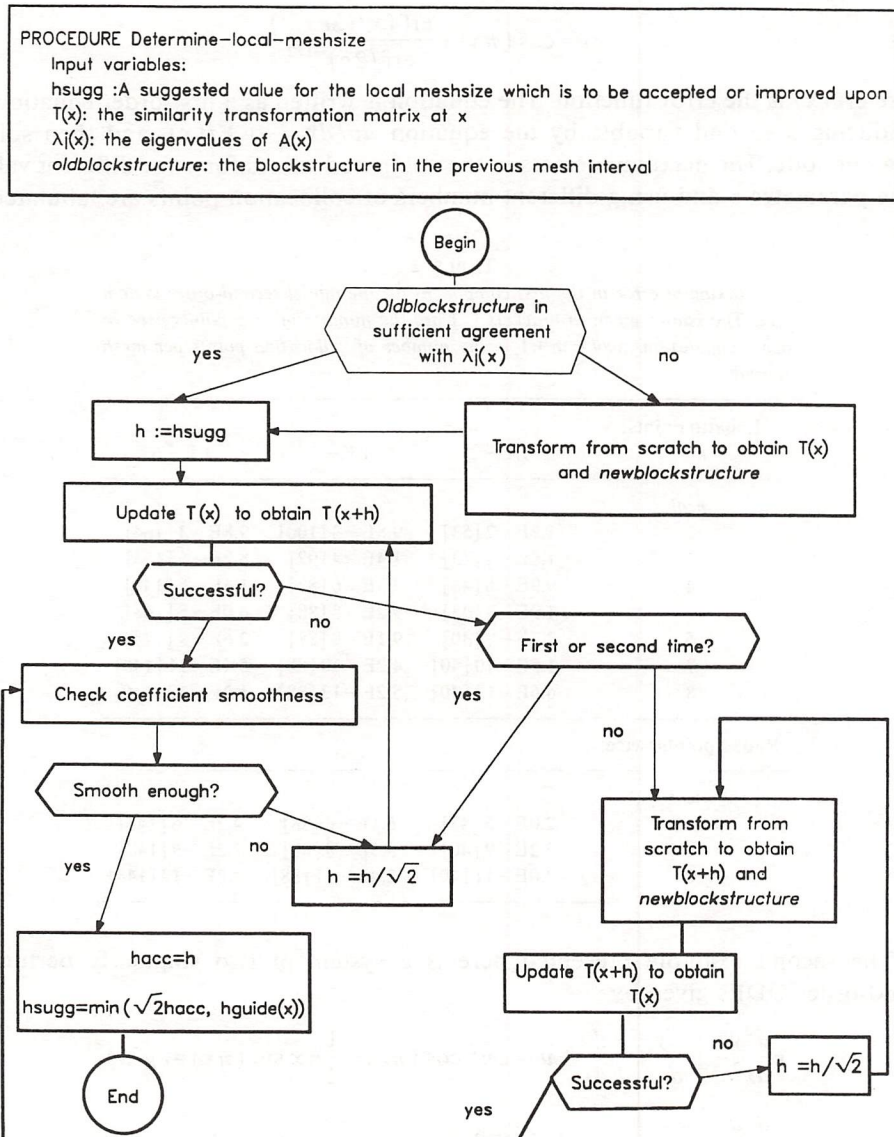

FIG. 3

As a final remark about the mesh construction, let us note that the procedure outlined so far might result in an abrupt decrease of the local mesh size. The theory presented in [12] requires a "smooth" mesh, however. Thus, it is necessary to provide the possibility of disregarding meshpoints already constructed and to approach a region again starting with a smaller mesh size at an appropriate point to the left of the "problem zone." This is implemented in our code.

7. Numerical examples. In this section we present the results of the computation of solutions to some stiff boundary value problems with turning points for which the exact solution is known. The first example is a scalar second order ODE given by [7]

$$(7.1) \qquad -\varepsilon \frac{d^2 y}{dx^2} - x \frac{dy}{dx} = \varepsilon \pi^2 \cos(\pi x) + \pi x \sin(\pi x) =: f(x), \qquad -1 \le x \le 1,$$

with boundary conditions $y(-1) = -2$, $y(1) = 0$. The exact solution is given by

$$(7.2) \qquad y(x) = \cos(\pi x) + \frac{\mathrm{erf}\,(x/(2\varepsilon)^{1/2})}{\mathrm{erf}\,(2\varepsilon)^{-1/2}}$$

where $\mathrm{erf}\,(x)$ is the error function. The equation is written as a first-order equation by introducing a second variable by the equation $dv/dx = -y + f(x)$, and then solved using our code. The maximum errors in the computed values for y for different values of the parameter ε and using different numbers of collocation points are tabulated in Table 2.

TABLE 2

Maximum error in the first component for the model second-order system (7.1). The values given in brackets [ ] are the number of meshpoints used in each computation. ncol = m + 1 is the number of collocation points per mesh interval.

Lobatto points: $\varepsilon =$	1.E−2	1.E−4	1.E−6
ncol			
2	1.2E−2 [53]	9.8E−3 [100]	9.8E−3 [164]
3	1.6E−4 [43]	1.4E−4 [92]	8.2E−5 [156]
4	9.9E−6 [43]	2.3E−6 [88]	1.4E−6 [148]
5	1.9E−7 [43]	9.2E−8 [88]	6.0E−8 [148]
6	2.7E−9 [40]	9.1E−9 [88]	2.6E−9 [148]
7	1.5E−10 [40]	4.2E−10 [88]	5.4E−11 [148]
8	6.6E−12 [40]	5.2E−12 [88]	1.2E−12 [140]
Radau points twice:			
ncol			
4	2.0E−5 [43]	6.1E−6 [88]	4.2E−6 [148]
6	7.2E−9 [40]	3.5E−8 [88]	7.2E−9 [148]
8	2.0E−11 [40]	2.4E−11 [88]	4.2E−12 [148]

The second example presented here is a system of two singularly perturbed second-order ODEs given by

$$
-\varepsilon \frac{d^2 y}{dx^2} - \frac{x}{2} \frac{dy}{dx} + \frac{x}{2} \frac{du}{dx} + u = \varepsilon \pi^2 \cos(\pi x) + \frac{1}{2} \pi x \sin(\pi x) =: g(x),
$$

$$(7.3)$$

$$
-\varepsilon \frac{d^2 u}{dx^2} + u = 0, \qquad -1 \le x \le 1,
$$

TABLE 3

Maximum error in the first component for the model fourth-order system
(7.3). The values given in brackets [ ] are the number of meshpoints used in
each computation. ncol = m + 1 is the number of collocation points per mesh
interval.

Lobatto points: $\varepsilon =$	1.E−4	1.E−6	1.E−8
ncol			
2	6.5E−2 [143]	6.3E−2 [250]	6.3E−2 [332]
3	1.2E−4 [122]	7.0E−5 [231]	4.1E−5 [315]
4	3.4E−7 [122]	3.5E−7 [223]	9.0E−6 [315]
5	1.1E−8 [122]	1.2E−8 [223]	1.4E−8 [308]
6	9.3E−10 [122]	1.0E−9 [223]	4.1E−10 [308]
7	5.4E−11 [122]	5.7E−11 [223]	2·5E−11 [308]
8	2.4E−12 [122]	3.0E−12 [122]	1.5E−12 [308]

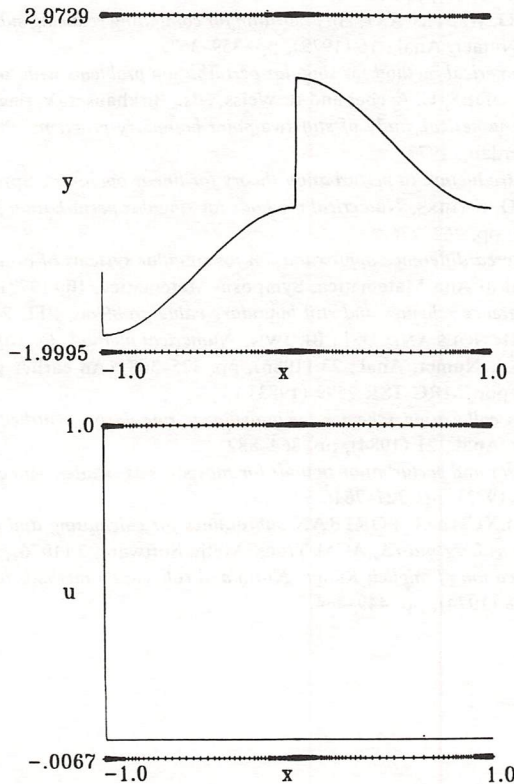| Radau points twice:			
ncol			
4	3.5E−6 [122]	3.6E−6 [223]	2.0E−6 [315]
6	5.0E−9 [122]	5.1E−9 [223]	4.8E−9 [308]
8	1.2E−11 [122]	1.2E−11 [223]	1.2E−11 [308]

FIG. 4

with boundary conditions $y(-1) = -1$, $u(-1) = 1$, $y(1) = u(1) = e^{-2/\sqrt{\varepsilon}}$. The exact solution for this problem is given by

$$(7.4) \qquad y(x) = \frac{\mathrm{erf}\,(x/2\sqrt{\varepsilon})}{\mathrm{erf}\,(1/2\sqrt{\varepsilon})} + u(x) + \cos\,(\pi x), \qquad u(x) = e^{-(x+1)/\sqrt{\varepsilon}}.$$

Equation (7.3) is written as a first-order system by introducing additional variables w and v by $\varepsilon\, du/dx = v$ and $dw/dx = \frac{1}{2}y + \frac{1}{2}xv + u - g(x)$. The maximum computed errors in the variable y are shown in Table 3 for three values of ε and for various numbers of collocation points. A plot of the solution is included in Fig. 4. It is clear from the tables of errors that the collocation formulas we have discussed here can compute very accurate solutions to singular perturbation problems with turning points.

Acknowledgment. We would like to thank Heinz Kreiss for many valuable discussions.

REFERENCES

[1] U. ASCHER, *On some difference schemes for singular singularly-perturbed boundary value problems*, Numer. Math., 46 (1985), pp. 1–30.

[2] U. ASCHER AND R. WEISS, *Collocation for singular perturbation problems* I: *First order systems with constant coefficients*, SIAM J. Numer. Anal., 20 (1983), pp. 537–557.

[3] ———, *Collocation for singular perturbation problems* II: *Linear first order systems without turning points*, Math. Comp., 43 (1984), pp. 157–187.

[4] R. H. BARTELS AND G. W. STEWART, *Algorithm 432: Solution of the matrix equation $AX + XB = C$*, Comm. ACM, 15 (1972), pp. 820–826.

[5] C. A. BAVELY AND G. W. STEWART, *An algorithm for computing reducing subspaces by block diagonalization*, SIAM J. Numer. Anal., 16 (1979), pp. 359–367.

[6] D. L. BROWN, *A numerical method for singular perturbation problems with turning points*, in Numerical Boundary Value ODES, U. Ascher and R. Weiss, eds., Birkhäuser–Verlag, Basel, Switzerland, 1985.

[7] P. W. HEMKER, *A numerical study of stiff two-point boundary problems*, Ph.D. thesis, Mathematisch Centrum, Amsterdam, 1977.

[8] T. KATO, *A short introduction to perturbation theory for linear operators*, Springer-Verlag, Berlin, 1982.

[9] B. KREISS AND H.-O. KREISS, *Numerical methods for singular perturbation problems*, SIAM J. Numer. Anal., 10 (1981), pp. 262–276.

[10] H.-O. KREISS, *Centered difference approximation for singular systems of ordinary differential equations*, Istituto Nazional di Alta Matematica, Symposia Matematica, 10 (1972), pp. 454–465.

[11] ———, *Central difference schemes and stiff boundary value problems*, BIT, 24 (1984), pp. 560–567.

[12] H.-O. KREISS, N. NICHOLS AND D. L. BROWN, *Numerical methods for stiff two-point boundary value problems*, SIAM J. Numer. Anal., 23 (1986), pp. 325–368. (An earlier version appeared as Univ. of Wisconsin Report MRC TSR 2599 (1983).)

[13] R. RINGHOFER, *On collocation schemes for quasilinear singularly perturbed boundary value problems*, SIAM J. Numer. Anal., 21 (1984), pp. 864–882.

[14] G. W. STEWART, *Error and perturbation bounds for subspaces associated with certain eigenvalue problems*, SIAM Rev., 15 (1973), pp. 727–764.

[15] ———, *HQR3 and EXCHNG: FORTRAN subroutines for calculating and ordering the eigenvalues of a real upper Hessenberg matrix*, ACM Trans. Math. Software, 2 (1976), pp. 275–280.

[16] R. WEISS, *The application of implicit Runge–Kutta and collocation methods to boundary-value problems*, Math. Comp., 28 (1974), pp. 449–464.