FINITE DIFFERENCE SOLUTION OF TWO-POINT BOUNDARY VALUE PROBLEMS AND SYMBOLIC MANIPULATION

V. Pereyra*

1. Introduction

Great emphasis has been put in recent years on the approximate solution of two-point boundary value problems for first order systems of differential equations. A number of authors have pointed out the advantages of doing so: mainly its generality, algorithmic simplicity, and the natural way in which arbitrary nonuniform meshes can be dealt with (cf. Keller [1974], Lentini and Pereyra [1977]).

None of these arguments have lost their standing, but we will show in this paper that a great deal of efficiency, both computational and storage wise, can be gained by treating high order systems as they come, instead of transforming them into larger first order systems.

The price to be paid for this efficiency is higher complexity, and in order to generate the wealth of finite difference formulae necessary to carry out our project in some generality we have had to resort to the use of an algebraic manipulation system: MACSYMA [1975].

The purpose of this paper is then to exhibit explicit finite difference formulae for high order linear differential systems subject to two-point boundary conditions. The basic schemes will be consistent, as compact as possible in the sense of Kreiss [1972], centered, and therefore they will be stable and second order accurate. If the data is regular it then will exist asymptotic expansions for the global error and therefore Richardson extrapolations or deferred corrections can be used to increase the accuracy (cf. Keller [1974], Pereyra [1967], Kreiss [1972]).

* This work was supported under Contract No. AT-04-3-767, Project Agree-

In Section 4 we present an explicit deferred correction algorithm. Complete details and additional results can be found in Keller and Pereyra [1977a, 1977b].

2. THE PROBLEM AND ITS DISCRETIZATION

We consider the general n-th order linear systems

$$(2.1) \qquad \mathcal{L}\, y(t) \equiv y^{(n)} + \sum_{\nu=0}^{n-1} A_\nu(t)\, y^{(\nu)}(t) = f(t) \ , \quad a \leq t \leq b \ ,$$

where y, f are d-vector functions and the $A_\nu(t)$ are $d \times d$ matrix functions. We will assume that f and A_ν are $C^N(c,d)$ functions, for N sufficiently large and $(c,d) \supset [a,b]$, and that (2.1) is subject to the nd linear boundary conditions

$$(2.2) \qquad B_k y \equiv \sum_{\nu=0}^{n-1} [\, B_{k\nu}(a)\, y^{(\nu)}(a) + B_{k\nu}(b)\, y^{(\nu)}(b)] = g_k \ , \quad 0 \leq k \leq n-1 \ ,$$

where the $B_{k\nu}(a)$, $B_{k\nu}(b)$ are $d \times d$ matrices. We assume that this nd linear constraints are independent and that the boundary value problem (2.1)-(2.2) has a unique smooth solution in (c,d).

Centered Compact Difference Schemes

To obtain centered schemes we must treat system (2.1) according to the parity of its order. We will concentrate here on the description of the even order case $n = 2m$. A more detailed description of both even and odd order cases, together with proofs will appear elsewhere (Keller and Pereyra [1977a]).

On the interval (c,d) we introduce a uniform net defined by

$$(2.3) \qquad \begin{aligned} t_j &= a + (j-\tfrac{1}{2})h \ , \quad h = (b-a)/J \ , \\ j &= -r \ , \ -r+1, \ \ldots, \ J+r \ . \end{aligned}$$

The integer r will be fixed independently of h and therefore we will need to go outside of the interval $[a,b]$ only a few mesh points. As we will see below, this is done to allow the use of centered approximations to (2.1) over the whole interval (a,b), and also to approximate the boundary

conditions (2.2) and to obtain the deferred correction operators. In this last respect, the present algorithm corrects a long standing and irritating defect of earlier implementations of deferred corrections for boundary value problems (see also Pereyra, Proskurowski and Widlund [1977] for a different cure to the same ailment). Actually the seeds of these ideas have been around for a long time as one can see in Fox [1957].

We introduce now some basic difference operators:

$$
\begin{aligned}
&D_+ v_j \equiv h^{-1}(v_{j+1} - v_j) \;\; ; \;\; D_- v_j \equiv h^{-1}(v_j - v_{j-1}) \\
&D_0 v_j \equiv (2h)^{-1}(v_{j+1} - v_{j-1}) \;\; ; \;\; M_+ v_j \equiv \tfrac{1}{2}(v_{j+1} + v_j) \; .
\end{aligned} \tag{2.4}
$$

The discretizations we consider are made up with powers of these operators and for $n = 2m$ they have the form:

$$
\begin{aligned}
\mathcal{L}_h u_j \equiv (D_+D_-)^m u_j + \sum_{\nu=0}^{m-1} [& A_{2\nu}(t_j)(D_+D_-)^\nu u_j \\
& + A_{2\nu+1}(t_j)(D_+D_-)^m D_0 u_j] = f(t_j) \; , \\
& 1 \leq j \leq J \; ,
\end{aligned} \tag{2.5}
$$

$$
\begin{aligned}
B_k^h u^h \equiv \sum_{\nu=0}^{m-1} \{ & [B_{k,\,2\nu+1}(a)(D_+D_-)^\nu D_+ u_0 + \\
& B_{k,\,2\nu+1}(b)(D_+D_-)^\nu D_+ u_J] + [B_{k,\,2\nu}(a)(D_+D_-)^\nu M_+ u_0 + \\
& B_{k,\,2\nu}(b)(D_+D_-)^\nu M_+ u_J] \} = g_k \; , \quad 0 \leq k \leq n-1 \; .
\end{aligned} \tag{2.6}
$$

These are second order consistent, as compact as possible (i.e. they use at most $n+1$ mesh points) approximations and the general theory of Kreiss [1972] guarantees that they will be stable and therefore convergent of order h^2.

In terms of ordinates (2.5) and (2.6) will take the form

$$\mathcal{L}_h u_j \equiv \sum_{s=-m}^{m} C_s(t_j;h) u_{j+s} = f_j$$

$$B_k^h u^h \equiv \sum_{s=-(m-1)}^{m} [C_{ks}(a;h) u_s + C_{ks}(b;h) u_{J+s}] = g_k$$

where C_s, $C_{ks}(a;h)$, $C_{ks}(b;h)$ are $d \times d$ matrices.

We would like to make these matrix coefficients more explicit, so that they can be computed. Writing the difference operators in terms of ordinates we obtain

$$D_{2,e}^{2\nu} u_i \equiv (D_+ D_-)^\nu u_i \equiv h^{-2\nu} \sum_{s=-\nu}^{\nu} \omega_s^{2\nu, e} u_{i+s}$$

$$D_{2,e}^{2\nu+1} u_i \equiv (D_+ D_-)^\nu D_0 u_i \equiv h^{-(2\nu+1)} \sum_{s=-(\nu+1)}^{\nu+1} \omega_s^{2\nu+1, e} u_{i+s} .$$

Replacing in (2.5), and reordering we obtain

$$C_s(t_j;h) = \omega_s^{n, e} I + \sum_{\nu=|s|}^{m-1} h^{-2\nu} A_{2\nu}(t_j) \omega_s^{2\nu, e} + \sum_{\nu=\max(0, |s|-1)}^{m-1} h^{-(2\nu+1)} A_{2\nu+1}(t_j) \omega_s^{2\nu+1, e} \tag{2.7}$$

$$s = -m, \ldots, m ; \; j = 1, \ldots, J ,$$

and similarly for the boundary conditions.

Thus the problem reduces to knowing the weights $\omega_s^{\mu, e}$ for various values of μ. For small values of μ they are standard and can be found in many places. It is unlikely that any physical application will ever require μ to be larger than 4, so obtaining these weights does not present any new problem. With them we can compute the matrix coefficients in our difference approximations and assembling them together we will arrive at a linear system of equations that the unknown mesh function u^h must satisfy

$$\mathcal{A} u^h = f^h . \tag{2.8}$$

$\mathcal{A}$ is a band matrix which in block form has the following aspect

$$
(2.9)\qquad \mathcal{A} = \begin{bmatrix}
C_{1,-m+1} & C_{1,-m+2} & \cdots & C_{1m} & O & \cdots & O \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
C_{p,-m+1} & C_{p,-m+2} & \cdots & C_{pm} & O & \cdots & \vdots \\
C_{1,-m} & C_{1,-m+1} & \cdots & C_{1,m-1} & C_{1,m} & O \cdots & O \\
O & C_{2,-m} & \ddots & \vdots & \vdots & C_{2m} & O \\
O & & & & & & \\
 & & C_{J,-m} & \cdots & & & C_{J,m} \\
 & O & & & C_{p+1,-m+1} & \cdots & C_{p+1,m} \\
O & & & & \vdots & \ddots & \vdots \\
 & & & & C_{nd,-m+1} & \cdots & C_{nd,m}
\end{bmatrix}
$$

where we have assumed separated boundary conditions, p of them initial conditions and $nd - p$ end conditions. The scalar half band width β is

$$
(2.10)\qquad \beta = \max(nd\ ,\ p+d\ ,\ (n+1)d-p) - 1\ .
$$

3. SOLUTION OF THE LINEAR EQUATIONS

At this stage most of the complexity is concentrated on solving the large, sparse system of linear equations (2.8). The data will have to be evaluated over the whole mesh $\{t_j\}$ whatever algorithm we use, so any possible savings will come from the algorithm used for these linear equations.

We have considered two possibilities for the high order systems:

a) A standard scalar band solver with partial pivoting.

b) A Gaussian elimination code with row pivoting for "almost block diagonal systems" by de Boor and Weiss [1976].

The number of arithmetic operations for (a) on a system with dJ equations is essentially

$$
(3.1)\qquad O_n^{(a)} \equiv 2n\ Jd\beta^2
$$

while for (b) we obtain in our case

$$O_n^{(b)} = \frac{Jd^2}{2}\,((2n+1)p+d) \approx Jd^2\ np \quad . \tag{3.2}$$

In terms of storage we have respectively

$$S_n^{(a)} = Jd(2\beta + 1) \quad , \tag{3.3}$$

$$S_n^{(b)} = J(n+1)d\ (p+d) \quad . \tag{3.4}$$

In order to be able to compare these quantities more easily we will assume now that $d \leq p \leq (n-1)d$, in which case $\beta = nd - 1$ and

$$\tilde{O}_n^{(a)} \equiv 2J\,n^2 d^3 \quad , \tag{3.5}$$

$$\tilde{S}_n^{(a)} = 2J\,nd^2 \quad . \tag{3.6}$$

Thus we see that the work and storage ratios are essentially

$$W^{b,a} \equiv \frac{O_n^{(b)}}{\tilde{O}_n^{(a)}} = \frac{p}{2nd} \quad , \tag{3.7}$$

$$R^{b,a} \equiv \frac{S_n^{(b)}}{\tilde{S}_n^{(a)}} = \left(\frac{p}{2d} + \tfrac{1}{2}\right)\frac{n+1}{n} \quad . \tag{3.8}$$

Method (b) will always require fewer arithmetic operations than method (a), but for $d \leq p$ it will take more storage. So our choice of algorithm will be guided by the type of computer installation constraints we may have.

Let us now consider the equivalent first order system with $n \times d$ equations associated with (2.1)-(2.2).

For first order systems we have available another linear equations solver: the alternate pivoting algorithm for block tridiagonal systems (B3D) described in Keller [1974] and used in Lentini and Pereyra [1975] (see also Pereyra and Lee [1977]). This algorithm is optimal in terms of space and its counts are

(3.9) $$O^{B3D} = Jd^3n^3$$

(3.10) $$S^{B3D} = 2Jn^2d^2 \ .$$

The scalar band solver (method (a)) is not competitive for first order systems since it takes 3/2 as much storage and 4.5 as much work than the B3D method.

We give now some operation and storage ratios for the various methods.

The count for method (b) on a first order system with (nd) equations is obtained from (3.2), (3.4) by replacing n by 1 and d by nd:

(3.11) $$O_1^{(b)} = \frac{Jn^2d^2}{2}(3p + nd) \ ,$$

(3.12) $$S_1^{(b)} = 2\,J\,n\,d(p + nd) \ .$$

Taking as a new parameter $r = p/nd$ we compare now method (b) with method B3D and we get the ratios

(3.13) $$W^{B3D,\,b} = \frac{2}{3r + 1} \ ,$$

(3.14) $$R^{B3D,\,b} = \frac{1}{r + 1} \ .$$

Since $0 < r < 1$, this says that method (b) always takes more storage than method B3D, and that it takes less arithmetic operations only if $\frac{2}{3r + 1} > 1$, i.e. if $r < 1/3$ or if $p < nd/3$.

Thus, for $p \geq nd/3$ method B3D is to be preferred, and for $p < nd/3$ one must balance the gain in computational speed against the loss in storage in order to choose the most appropriate method.

However, our main interest now is to compare the best method for an nth order system with that for its equivalent first order system. Thus we list now the work and storage ratios for method (b) as an nth order solver as compared to methods B3D and (b) again as a first order system solver:

$$W_{1,n}^{b,b} = \frac{O_1^{(b)}}{O_n^{(b)}} = \frac{n}{2r} + \frac{3n}{2} \quad ,$$

(3.15)

$$R_{1,n}^{b,b} = \frac{S_1^{(b)}}{S_n^{(b)}} = 2 + 2/r$$

(3.16) $$W_{1,n}^{B3D,b} = \frac{n}{r} \quad , \quad S_{1,n}^{B3D,b} = 2/r \quad .$$

Formulas (3.15)-(3.16) show some impressive gains to be obtained if one solves the problem as an nth order system. For instance, for $n = 4$, $r = \frac{1}{2}$ we gain a factor 8 in speed by using method (b) as compared to B3D, while using only 1/4 as much storage.

For $n = 4$, $r = 1/4$ we use (b) as a first order solver since it is faster, but still (b) as an nth order solver will run 14 times faster and use 1/10 as much storage.

In conclusion, for first order systems the best solvers are Keller's Block Tridiagonal solver with alternate pivoting and de Boor and Weiss almost block diagonal solver.

For high order systems, it is best to solve them as they come using the methods of this paper instead of transforming them to first order. If storage is no concern then the de Boor and Weiss algorithm should be used since it is faster. However, if storage is a problem then the slower but less space consumming scalar band solver with partial pivoting should be preferred.

4. HIGHER ORDER METHODS

We would like to obtain now methods with orders of accuracy higher than 2, while preserving the structure of the linear systems (2.8) to be solved. An efficient way of doing this is by deferred corrections.

If we use the exact solution $y(t)$ in (2.5)-(2.6) we obtain the local truncation error:

(4.1) $$\tau_h \equiv \mathcal{L}_h[y]$$

and similarly for B_k^h. Expanding in Taylor's series we get

$$\tau_{hj} = \sum_{\nu=1}^{L} h^{2\nu} T_\nu[y_j] + \mathcal{O}(h^{2L+2}) , \tag{4.2}$$

where

$$T_\nu[y_j] = \sum_{\mu=0}^{n} \alpha^e_{\nu\mu} A_\mu(t_j) y^{(\mu+2\nu)}(t_j) . \tag{4.3}$$

An important consequence of using exclusively centered formulae is that the expansion (4.2) has only even powers of h.

The coefficients $\alpha^e_{\nu\mu} y_j^{(\mu+2\nu)}$ of (4.3) can be obtained from the expansions of the basic difference operators (2.4) with the aid of MACSYMA. For instance, for even $\mu = 2k$, we start from the well known expansion for (D_+D_-):

$$(D_+D_-)y_j \equiv D^2_{2,e} y_j = [D^2 + \sum_{\nu=1}^{L} \frac{2D^{2\nu+2}}{(2\nu+2)!} h^{2\nu}] y_j + \mathcal{O}(h^{2L+2}) , \tag{4.4}$$

where $D^s y_j \equiv y_j^{(s)}$.

Taking kth (operator) powers in (4.4) we obtain

$$(D_+D_-)^k y_j = [D^2 + \sum_{\nu=1}^{L} \frac{2D^{2\nu+2}}{(2\nu+2)!} h^{2\nu}]^k y_j + \mathcal{O}(h^{2L+2})$$

or

$$(D_+D_-)^k y_j = D^\mu y_j + \sum_{\nu=1}^{L} \alpha^e_{\nu\mu} h^{2\nu} y^{(2\nu+\mu)} + \mathcal{O}(h^{2L+2}) . \tag{4.5}$$

Thus, the exact rational coefficients $\alpha^e_{\nu\mu}$ are obtained through MACSYMA which very simply computes symbolically the kth power of the expansion, reorganizes it in powers of h^2 and drops the terms of order higher than $2L+2$.

Once the asymptotic expansion for the local error is available, the deferred correction procedure calls for approximating segments of this expansion to increasing orders in h. In the past we have used fast Vandermonde

solvers to generate the weights for the necessary difference formulae (cf. Pereyra [1973]). This turns out to be quite an expensive procedure in this application and we prefer to use additional symbolic manipulation in order to generate a priori (and exactly) all the necessary weights and store them for later use.

The procedure is explained in detail and quite extensive tables of coefficients are presented in Keller and Pereyra [1977b].

At the qth step of the deferred correction procedure we will need to approximate q terms of the local truncation error to order h^{2q}. We denote by $\{u_j^{(q-1)}\}$ the (q-1)-corrected discrete solution which is accurate to order h^{2q}. The correction operator is

$$S_q(u_j^{(q-1)}) = \sum_{s=-(m+q)}^{m+q} \sum_{\nu=1}^{q} \Big[\sum_{\mu=\max(0,\,|s|-q)}^{m} \omega_s^{2(\mu+\nu),\,\tilde{q},\,e}\, \alpha^e_{\nu,\,2\mu}\, A_{2\mu}(t_j) h^{-2\mu}$$

$$+ \sum_{\mu=\max(0,\,|s|-q-1)}^{m-1} \omega_s^{2(\mu+\nu)+1,\,\tilde{q},\,e}\, \alpha^e_{\nu,\,2\mu+1}\, h^{-(2\mu+1)}\, A_{2\mu+1}(t_j) \Big] u_{j+s}^{(q-1)}$$

where $\tilde{q} = q-\nu+1$, and the weights $\omega_s^{\ell,\,q}$ are coefficients in the expansions of $D^r_{2q,\,e}$, the $O(h^{2q})$ approximation to D^r. Both the approximations and their expansions are obtained with MACSYMA.

The qth corrected value is then obtained by solving the linear system

$$\mathcal{A}\, u = f + S_q(u^{(q-1)}) \ ,$$

and

$$u^{(q)} - y = \mathcal{O}(h^{2q+2}) \ .$$

REFERENCES

de Boor, C. and R. Weiss, "SOLVEBLOK: A package for solving almost block diagonal linear systems, with applications to spline approximation and the numerical solution of ordinary differential equations," MRC Techn. Rep. No. 1625, Madison, Wisc. (1976).

Fox, L., Numerical Solution of Two-Point Boundary Value Problems, Clarendon Press, Oxford (1957).

Keller, H. B., "Accurate difference methods for nonlinear two-point boundary value problems," SIAM J. Numer. Anal. 11, 305-320 (1974).

Keller, H. B. and V. Pereyra, "Difference methods and deferred corrections for ordinary boundary value problems," To appear.

Keller, H. B. and V. Pereyra, "Symbolic generation of finite difference formulae," To appear.

Kreiss, H.-O., "Difference approximations for boundary and eigenvalue problems for ordinary differential equations," Math. Comp. 26, 605-624 (1972).

Lentini, M. and V. Pereyra, "PASVA2-Two point boundary value problem solver for nonlinear first order systems," Lawrence Berkeley Lab. program documentation report (1975).

Lentini, M. and V. Pereyra, "An adaptive finite difference solver for nonlinear two-point boundary problems with mild boundary layers," SIAM J. Numer. Anal. 14, 91-111 (1977).

Pereyra, V., "Iterated deferred corrections for nonlinear operator equations," Numer. Math. 10, 316-323 (1967).

Pereyra, V. High Order Finite Difference Solution of Differential Equations. STAN-CS-73-348. Stanford Univ., California (1973).

Pereyra, V. and W. H. K. Lee, "Solving two-point seismic ray-tracing problems in a heterogeneous medium. Part I: A general numerical method based on adaptive finite differences," To appear.

MACSYMA Reference Manual. The Math Lab. Group, Project MAC. MIT, Boston (1975).

Pereyra, V., W. Proskurowski and O. Widlund, "High order fast Laplace solvers for the Dirichlet problem on general regions," Math. Comp. 31, 1-16 (1977).