REPRINTED FROM
HIGH SPEED COMPUTER AND ALGORITHM ORGANIZATION

ACADEMIC PRESS, INC
NEW YORK SAN FRANCISCO LONDON

# ALGORITHMS FOR SOLVING TWO-POINT BOUNDARY VALUE PROBLEMS

Victor Pereyra
California Institute of Technology

## 1. INTRODUCTION

We will consider the numerical solution of two types of boundary value problems

I. $$y' = f(t,y), \quad t \in [a,b],$$

$$g(y(a), y(b)) = 0$$

with smooth vector functions $y$, $f$, $g \in \mathbb{R}^d$, and

II. $$y_j^{(n_j)} = f_j(t, y_1, \ldots, y_1^{(n_1-1)}, \ldots, y_d^{(n_d-1)})$$

$$j = 1, \ldots, d; \; n_j \geq 1$$

$$g(y_1(a), \ldots, y_d^{(n_d-1)}(a), y_1(b), \ldots, y_d^{(n_d-1)}(b)) = 0,$$

$$g \in \mathbb{R}^{\sum_{j=1}^{d} n_j}.$$

We will concentrate on methods for which there are known, available, and reasonably tested computer implementations.

## 2. BASIC METHODS

For problem I we have two main kinds of methods:

i) Multiple shooting, with adaptive choice of shooting points [4,9,12,30,35].
ii) Finite differences or collocation with variable order and variable step [3,13,20,22,29].

For problem II there are many spline methods for

the case of a single equation (d = 1) but no general software is available that we know of. Some very recent developments using finite differences with variable order in the spirit of what exists for first order systems show some promise [15,16]. Also mixed finite difference - collocation methods are being developed which are promising [7,8].

Methods (i), (ii) are closely related. The main attraction of method (i) is that well understood available software for initial value problems form a considerable part of the algorithms core, thus diminishing the development costs.

Outside of that,(i) and (ii) share the same difficulties: solution of systems of nonlinear equations, mesh selection, and even the linear equations to be solved in iterative procedures have the same block structure. In general, though, the sizes of the discrete problems for (i) are smaller than for (ii) and therefore (ii) is more storage consuming. On the other hand, usually the implementations of (i) do not permit control on what the initial value problem solvers do between shooting points, and therefore method (ii) usually provides finer error control and more solution detail.

The nonlinear discrete equations are usually solved by Newton's method or some of its variations. Many routine, and otherwise simple problems, present great difficulties in the determination of suitable initial values and several techniques are used to increase the robustness and global convergence properties of the iterative method being used. Among others: continuation or embedding, step control, and forcing the iteration to be of descent for a norm of the residual [6,27,28].

Most iterations require the solution of a linearized set of equations at every step. For the methods mentioned above the resulting systems have sparse matrices of coefficients with a highly structured distribution of zeros. In fact, an intelligent ordering of the equations produces essentially block tridiagonal matrices, with some further sparseness within the nonzero blocks.

Special LU decomposition schemes with alternate pivoting have been developed which solve this type of systems efficiently and accurately [13,21,31].

The systems produced by the discretization of II are somewhat more complicated in structure. Band solvers with partial pivoting has been used so far [15], but there are probably more efficient algorithms [3], although they may be more space consuming.

It is in the solution of these special systems of linear equations where parallel or vector computation can be

of use for this type of problems.

## 3. ERROR ESTIMATION AND VARIABLE ORDER

The implementations of algorithms in class (i) inherit the capabilities of their core initial value problem solvers in what error estimation and variable order is concerned. Usually error estimation and control will be limited to local truncation error as related to the shooting between station points, and therefore the relation with actual global error may be, at its best, tenuous. Of course, global error estimates similar to those employed by implementations of methods of class (ii) are possible, but no program that we know of bothers to pay the price to obtain this added, and we think, fundamental information.

Methods of class (ii), more specifically those using deferred corrections to increase the order of accuracy [20,22,24] have natural built in capabilities that provide asymptotic local and global error estimates with practically no extra cost. This is in the nature of global solvers, as opposite to marching techniques.

In our implementations the global estimates are used to steer the program through its logical tree, while the local ones are used to select dynamically a net that adapts itself to the problem at hand.

## 4. MESH SELECTION

The area of adaptive mesh selection for boundary value problems is one in which active research is being carried out [2,5,18,22,26,32].

Here again we have to separate the problems in two broad categories with fuzzy boundaries:

a) Regular problems with at worst moderately large gradients.
b) Boundary layer problems and quasi-singularities with extremely different time scales.

For this last class of problems, methods combining singular perturbation techniques [1,10], and others that make a very delicate study of the problem have been proposed [18], although no practical implementations are available at the time of this writing.

Problems of type (a) are amenable to methods that recursively adapt the mesh, building up information as

the computation proceeds. A technique that has proven to be fairly successful and for which there is some theoretical backing [5,26,32] is that of asymptotically equidistributing some integral norm of the local truncation error [5,22]. Roughly speaking, equidistributing consists of making the net finer where the local truncation error (i.e. a multiple of some high order derivative of the solution) is large.

In [32], White has proposed a very appealing method for computing equidistributed meshes with respect to various monitor functions. Unfortunately the method as stated has found some unexpected difficulties in its implementation, but nevertheless we think it deserves further consideration. It is worth pointing out that the mesh selection algorithm in [22] is actually a successful implementation of White's idea for the case in which the local truncation error is used as a monitor function. We feel now that an implementation even more closely related to White's proposal could be achieved that might share the best of both approaches.

## 5. STRUCTURE OF PROGRAMS, MINIMUM CALLING SEQUENCES AND DESIRABLE OPTIONS

With the vague descriptions we have given in the former sections, complemented if necessary with the detailed versions to be found in the quoted literature, we would like now to indicate what we think the general skeleton of an implementation should be. To make things somewhat more precise we will have in mind problem I and finite difference methods for which we have direct experience, but it should be apparent that the basic techniques indicated in most of the black boxes can be fully exchanged by alternate ones, and also applied to problems of type II.

Also, we have tried to understress the amount of automatic decisions, providing some exit nodes ◇ which could be appropriate for inputing users decisions in a semi-interactive approach. This diminishes the algorithm's complexity and may provide the program with much greater flexibility. Present implementations are fully batch mode oriented, but allowing some user interaction could enhance the class of problems that can be solved effectively. Certainly the use of available interactive graphic facilities would be desirable in many specific applications.

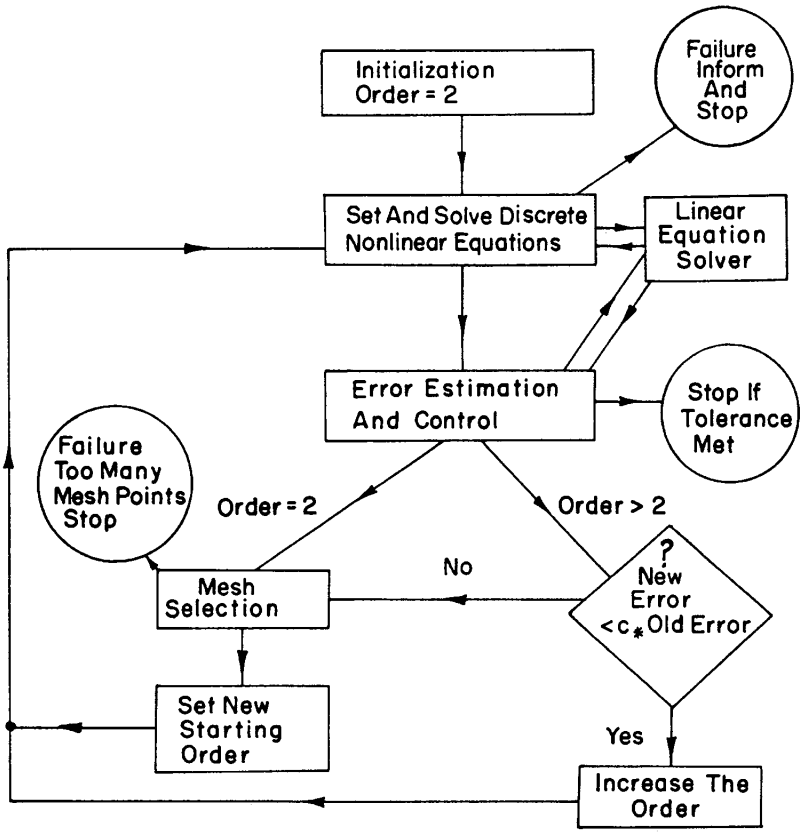
A calling sequence for a subroutine of this type should contain at least the following parameters:

M = Number of differential equations in the system.
N = Number of mesh points in input net. On output this parameter could contain the number of points of the final net.
P = Number of initial conditions.
R = Number of coupled conditions.
A = Left end point of interval of integration.
B = Right end point of interval of integration.
TOL = Desired accuracy.
X = One dimensional array containing initial mesh. On output X may contain the final mesh.
Y = Array containing initial approximation (on the mesh X). On output Y may contain the answer.
F = Subroutine specifying the right hand sides of the equations.
JACOB = Subroutine specifying the Jacobian matrix of F with respect to y (in case explicit partial derivatives are used instead of some other type of quasi-Newton iteration).

Options. In our implementation [21] we have found useful to offer some options in order to facilitate the use of the program. We employ a one dimensional array OPT to input these options. If OPT(1) = 0 then default options will be activated and no other element of OPT need be specified. If OPT(1) = 1 then all the elements of OPT must be given values.

OPT(2) controls the use of an automatic continuation option for badly nonlinear problems.
OPT(3) controls intermediary printed output.
If OPT(4) = 0 the program will set the initial values for X and Y.
OPT(5) = 0 the problem is linear.
OPT(5) $\geq$ 1 the problem is nonlinear. By setting a larger value of OPT(5) the user can indicate that the problem nonlinearity should be treated more carefully. This will usually produce a less efficient performance, but in many cases it will solve the problem where a smaller value of OPT(5) would not.

'Dynamic' array dimensioning can be simulated in FORTRAN by adding two one-dimensional work arrays (one for integer and one for real variables) as in J. Bolstad's revised version of PASVA2 [34].




FLOW CHART FOR ADAPTIVE TWO-POINT BOUNDARY VALUE PROBLEM SOLVER

## 6. HIGH ORDER SYSTEMS

Problems of class II are fairly common in the applications, at least for $n_j \leq 4$. In the past we have adopted the position that since reduction to first order systems is always feasible, it is best to concentrate in providing high quality software for problems of class I.

This is the same attitude taken by the people working in software for initial value problems, with the exception of Krogh [19]. However, recent investigations [15] have shown that there might be significant gains in efficiency if high order systems are solved as they arise, instead of reducing them to lower order.

There is a large increase in complexity when passing from class I to II, and in our preliminary investigations we have concentrated in some simpler special cases in order to explore the feasibility and properties of the new approach.

Since any nonlinear solver is based on linear solvers, as a first step we have considered only linear systems.

The type of basic finite difference methods we use are symmetric, compact as possible, second order accurate ones, i.e. they use symmetric, $(n_j + 1)$ point formulas. Only uniform meshes are employed. In order to be able to obtain such methods we need to consider odd and even order equations separately, and to be able to use existing results on convergence [11,17] we have to concentrate in systems for which $n_j \equiv n$, $j = 1,\ldots,d$. So we have an even (fixed) order system solver, and an odd order one. Deferred corrections, providing variable order of accuracy have been implemented only for the even order solver so far.

All the finite difference formulas necessary in the basic approximation and in the correction procedure are constructed with precalculated weights. This preliminary work has been highly facilitated by the use of MACSYMA [23], a very powerful symbolic manipulation system available at MIT, and which we have accessed through the ARPA net.

This is in contrast with previous implementations where fast Vandermonde solvers [33] were used to generate weights whenever they were needed. A considerable saving is attained by this new approach.

Another new feature is the incorporation of Keller's modification of the deferred correction algorithm [14] that permits the use of symmetric correction formulas

over the whole interval [a,b], by considering auxiliary mesh points external to this interval. This technique removes some theoretical difficulties and it also produces discrete solutions with orders of accuracy much closer to the theoretical ones than in earlier implementations.

## REFERENCES


1. Abrahamsson, L.R., Keller, H.B. and Kreiss, H.O., "Difference approximations for singular perturbations of systems of ordinary differential equations," Numer. Math. 22, 367-391 (1974).
2. de Boor, C., "Good approximation by splines with variable knots II," Lecture Notes in Math. No. 363, 12-20. Springer-Verlag, Berlin (1973).
3. de Boor, C. and Weiss, R., "SOLVEBLOK: a package for solving almost block diagonal linear systems, with applications to spline approximation and the numerical solution of ordinary differential equations," MRC Rep. 1625, Math. Res. Center, U. of Wisconsin, Madison (1976).
4. Bulirsch, R., Stoer, J. and Deuflhard, P., "Numerical solution of nonlinear two-point boundary value problems I." To appear in Numer. Math., Handbook Series Approximation.
5. Christiansen, J. and Russell, R.D., "Adaptive mesh selection strategies for solving boundary value problems." To appear in SIAM J. Numer. Anal.
6. Deuflhard, P., "A modified Newton method for the solution of ill-conditioned systems of nonlinear equations with applications to multiple shooting," Numer. Math. 22, 289-315 (1974).
7. Doedel, E.J., "The construction of finite difference approximations to ordinary differential equations." To appear in SIAM Numer. Anal.
8. Doedel, E.J., "Finite difference method for nonlinear two point boundary value problems." Submitted for publication to SIAM Numer. Anal.
9. England, R., Nichols, N. and Reed, J., "Subroutine DD03AD." Harwell Lab., England (1973).
10. Ferguson, W.E. Jr., "A singularly perturbed linear two-point boundary-value problem." Ph.D. Thesis, App. Math. Caltech, Pasadena, Ca. (1975).
11. Grigorieff, R.D., "Die Konvergenz des Rand-und Eigenwert problems linear gewöhnlicher Differenzengleichungen," Numer. Math. 15-48 (1970).
12. Keller, H.B., "Numerical Methods for Two-Point Boundary-Value Problems," Blaisdell, Mass. (1968).

13. Keller, H.B., "Accurate difference methods for nonlinear two-point boundary value problems," SIAM J. Numer. Anal. 11, 305-320 (1974).
14. Keller, H.B., "Numerical Solution of Two-Point Boundary-Value Problems." Regional Conf. Series in App. Math. 24 (1976).
15. Keller, H.B. and Pereyra, V., "Difference methods and deferred corrections for ordinary boundary value problems." In preparation.
16. Keller, H.B. and Pereyra, V., "Symbolic generation of finite difference formulae." In preparation.
17. Kreiss, H.O., "Difference approximations for boundary and eigenvalue problems for ordinary differential equations," Math. Comp. 26, 605-624 (1972).
18. Kreiss, H.O., "Difference approximation for singular perturbation problems." Proc. NSF Symp. on Numerical Solution of Boundary Problems for Ordinary Differential Equations. Academic Press, New York (1975).
19. Krogh, F.T., "Variable order integrators for the numerical solution of ordinary differential equations." JPL Tech. Memo., Caltech, Pasadena, Ca. (1970).
20. Lentini, M. and Pereyra, V., "A variable order finite difference method for nonlinear multipoint boundary value problems," Math. Comp. 28, 981-1004 (1974).
21. Lentini, M. and Pereyra, V., "PASVA2 - Two Point boundary problem solver for nonlinear first order systems." Lawrence Berkeley Lab., Univ. of California (1975).
22. Lentini, M. and Pereyra, V., "An adaptive finite difference solver for nonlinear two point boundary problems with mild boundary layers," to appear in SIAM J. Numer. Anal., March (1977).
23. MACSYMA Reference Manual. Mathlab Group, Project MAC. MIT, Boston, Mass. (1975).
24. Pereyra, V., "Iterated deferred corrections for nonlinear operator equations," Numer. Math. 10, 313-323 (1967).
25. Pereyra, V., "High order finite difference solution of differential equations," STAN-CS-73-348, Comp. Sc. Dep., Stanford Univ., Cal. (1973).
26. Pereyra, V. and Sewell, G., "Mesh selection for discrete solution of boundary problems in ordinary differential equations," Numer. Math. 23, 261-268 (1975).
27. Pereyra, V. and Lee, W.H.K., "Numerical solution of nonlinear two-point boundary value problems and

its application to seismic ray tracing, Part I: The general numerical method and its implementation." Manuscript (1977).

28. Roberts, S.M. and Shipman, J.S., Two Point Boundary Value Problems: Shooting Methods, American Elsevier, New York (1972).
29. Russell, R.D., "Collocation for systems of boundary value problems," Numer. Math. 23, 119-133 (1974).
30. Scott, M.R. and Watts, H.A., "SUPORT - A computer code for two-point boundary-value problems via orthonormalization," SAND 75-0198, Sandia Labs., Albuquerque, N.M. (1975).
31. Varah, J.M., "On the solution of block-tridiagonal systems arising from certain finite-difference equations," Math. Comp. 26, 859-868 (1972).
32. White, A.B. Jr., "On selection of equidistributing meshes for two-point boundary-value problems." Rep. 112, Center Num. Anal., Univ. of Texas at Austin (1976).
33. Björck, A. and Pereyra, V., "Solution of Vandermonde systems of equations." Math. Comp. 24, 893-903 (1970).
34. Bolstad, J., "Revised version of PASVA2," Comp. Sci. Dept., Stanford Univ. (1977).
35. Scott, M.R. and Watts, H.A., "Computational solution of nonlinear two-point boundary-value problems." SAND 77-0091, Sandia Labs., Albuquerque, N.M. (1977).

## ACKNOWLEDGMENT

This work was supported under Contract No. AT-04-3-767, Project Agreement 12 with Energy Research and Development Administration.