101

0-315-28296-7

NAME OF AUTHOR/*NOM DE L'AUTEUR* Kam Nang Hareton Leung

TITLE OF THESIS/*TITRE DE LA THÈSE* Spline Collocation for Solving Time Discretized Parabolic Problems in One Space Variable

UNIVERSITY/*UNIVERSITÉ* Simon Fraser Unversity

DEGREE FOR WHICH THESIS WAS PRESENTED/ *GRADE POUR LEQUEL CETTE THÈSE FUT PRÉSENTÉE* Master of Science

YEAR THIS DEGREE CONFERRED/*ANNÉE D'OBTENTION DE CE GRADE* 1983

NAME OF SUPERVISOR/*NOM DU DIRECTEUR DE THÈSE* Robert D. Russell

DATED/*DATÉ* Aug 15 83 SIGNED/*SIGNÉ*

PERMANENT ADDRESS/*RÉSIDENCE FIXÉ* [redacted]

SPLINE COLLOCATION FOR SOLVING TIME DISCRETIZED PARABOLIC PROBLEMS IN ONE SPACE VARIABLE

by

Kam Nang Hareton Leung

B.Sc. (Hon.) University of British Columbia, 1980

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in the Department

of

Computing Science

SIMON FRASER UNIVERSITY

August 1983

Title of Thesis/Project/Extended Essay

Spline Collocation for Solving Time

Discretized Parabolic Problems in

One Space Variable

Author:

(signature)

Kam Nang Hareton Leung

(name)

Aug 15, 83

(date)

APPROVAL

Name: Hareton Leung (Kam Nang Hareton Leung)

Degree: Master of Science

Title of thesis: Spline Collocation for Solving Time Discretized Parabolic Problems in One Space Variable

Examining Committee:

Chairperson: Wo Shun Luk

Robert D. Russell
Senior Supervisor

Richard F. Hobson

Binay K. Bhattacharya

(in absentia - report dated 3 August 1983)

John W. Paine
External Examiner
Visiting Assistant Professor
Department of Mathematics
Simon Fraser University

Date Approved: Aug. 10, 1983

ABSTRACT

The numerical solution of the time varying system of partial differential equations in one space dimension of the form

$$E(U,x,t)\,U_t + (D(U,x,t)\,U_x)_x + F(U,U_x) + C = 0$$

is considered. The solution technique used is a combination of (1) discretization of the temporal variable using different orders of implicit formulae, and (2) solution in space by a varying spatial mesh, spline collocation method, using a Hermite basis.

A preliminary implementation of the algorithm in FORTRAN, called HERPDE (HERmite collocation for Partial Differential Equation), was able to satisfactorily solve all the test problems. The results strongly suggested that this approach merits further investigation.

An algorithm for solving the linear system arising from spline collocation with a monomial basis is presented and shown to require fewer operations than the existing methods.

ACKNOWLEDGMENTS

I would like to thank professor Binay Bhattacharya, professor Rick Hobson and professor John Paine for their advice and encouragement.

My deepest gratitude goes to my thesis supervisor, professor Bob Russell, whose continuous support and guardian have made the completion of this thesis possible.

I would also like to thank the Computing Science department for providing an excellent research environment for pursuing one's interest.

I dedicate this thesis to my parents who taught me that learning can be both fun and rewarding.

TABLE OF CONTENTS

Approval ii

Abstract iii

Acknowledgments iv

List of Tables vii

List of Figures viii

1. INTRODUCTION 1

2. COLLOCATION METHOD FOR ORDINARY DIFFERENTIAL EQUATIONS 5

2.1 Collocation with Piecewise Polynomials (Splines) 6

2.2 Local Representation 12

2.2.1 Monomial Basis 12

2.2.2 Condensation of Parameters 17

2.2.3 ASIS 18

2.2.4 Operation Counts 24

2.2.5 Hermite Basis 28

2.3 Solution of Nonlinear Boundary-Value Problem 32

3. DESIGN OF HERPDE 35

3.1 Motivation 35

3.2 Description of Problem 37

3.3 Temporal Discretization 38

3.4 Spatial Mesh Selection 39

3.5 Programming Considerations 45

3.5.1 Supplying Previous Time Step Solutions 45

3.5.2 Generation of Gaussian Points 48

3.6 Preliminary Experiments 50

3.6.1 Effects of the Degree of Basis Function 51

3.6.2 Performance of the Mesh Selection Algorithm ..56

4. NUMERICAL EXAMPLES 61

4.1 Method of Lines 61

4.1.1 PDECOL 61

4.1.2 DPDES 62

4.2 Test Problems and Results 64

5. CONCLUSIONS 74

5.1 Suggestions for Further Study 75

5.2 Summary 76

List of References 77

LIST OF TABLES

TABLE		PAGE
1	Operation counts (N factor omitted)	27
2	Storage and computation counts for the six discretization schemes	47
3	Variation of execution time with degree of basis function	54
4	Variation of absolute error with degree of basis function	54
5	Computational results for Problem 8	66
6	Computational results for Problem 9	69
7	Computational results for Problem 11	73

LIST OF FIGURES

FIGURE		PAGE
1	The structure of MONO, the almost block diagonal collocation matrix using local monomials	16
2	The almost block diagonal structure of a standard multiple shooting matrix	20
3	The structure of S after decomposition	23
4	The block structure of HER7, a Hermite basis collocation matrix	32
5	Mesh selected for Problem 3	57
6	Mesh selected for Problem 4	58
7	Mesh selected for Problem 5	59
8	Mesh selected for Problem 7	60
9	Computational results at t=1 for Problem 10	71

1. INTRODUCTION

It is well known that the ideas and techniques for solving ordinary boundary value problems (BVP's) can be used to solve partial differential equations (PDE's). There are at least two approaches in the solution of parabolic problems in one space variable - both reduce to ordinary differential equations (ODE's).

One approach involves discretizing the spatial variable and solving the resulting initial value problems (IVP's). This is the standard method of lines approach, which has been used widely for some time. The finite difference formulae are traditionally used for the spatial discretization. In the present investigation, we refer to any scheme which first discretizes the space variable and then solves a set of IVP's as the method of lines. The discretization can be finite element or finite difference approximation. Recently, Madsen and Sincovec [20] have developed a nonlinear PDE solver which uses collocation with B-splines for the discretization of the spatial variable. This approach has shown to be applicable in solving some problems such as those arising in petroleum engineering [29]. Another method belonging to this approach is that of Schryer [28], who uses Galerkin's method in space, using B-splines, and a variable order, variable time-step backward difference procedure in time. His code, POST, can solve PDE's with terms including u_{xt} and u_{xxt}: Hopkins and Wait [15] have done a performance comparison of using Galerkin, collocation and finite difference for the spatial discretization. One attraction of the method of lines is that one can use the robust IVP software. However, it has at least three drawbacks:

1. This technique increases the complexity of determining the solution of the problem because the number of ODE's is increased substantially. The number of ODE's (O) is determined by

   $$O = P * D$$

   where P is the number of PDE's, and D is the number of discretization points.

   For some practical problems, P = 100 and D = 250.
2. To avoid storage overhead, most software reevaluates the Jacobian whenever the stepsize Δt is changed, thus, increasing the amount of recomputation. This is undesirable if the Jacobian is reasonably well behaved but Δt changes frequently.
3. If a solution has transient regions, fixed space discretization causes problems because it cannot follow the evolving solution.

The other alternative is to discretize the temporal variable, reducing the PDE to a system of ordinary boundary value problems. Due to the competitive BV software recently developed (e.g. COLSYS [2], a spline collocation solver with mesh selection capability), it is natural to try the boundary value approach. Since spline collocation has been proven effective in solving ODE's, it is reasonable to apply it in solving the reduced problem. Corless [11] has done a feasibility study on this approach by using COLSYS as the BVP solver. His results indicate that this approach gives fairly accurate solutions.

Since the class of problem being considered includes some whose solutions have shocks or boundary layers, it is essential that the solution scheme provides the capability to adapt the spatial mesh to the solution profile. Various attempts have been made to construct reliable and efficient adaptive mesh algorithm. Anizor [1] developed an adaptive mesh selection scheme for boundary-layer type differential equations, using the box scheme. His results showed that the solutions obtained from the adaptive mesh are more accurate than those from the uniform mesh. Using a different approach, White [30] implemented a mesh selection scheme which involves the arc-length transformation of the problem to a new coordinate system. More recently, Davis and Flaherty [12] have designed a spatial mesh selection scheme which seems to work well with their Galerkin method.

It is not obvious which approach to the solution of parabolic problems in one space variable is superior. One might expect their efficiency to be problem-dependent.

In this study, we describe an implementation of a spline collocation method with time discretization for solving a class of parabolic problems. Because recent research, Ascher et al.

[4], on spline basis selection for the collocation method indicates that the popular B-splines basis representation is harder to implement and somewhat less efficient than the monomial and Hermite-type bases, we have chosen the Hermite basis. While not in polished form, our code HERPDE (HERmite basis collocation for PDE) has been sufficiently successful at solving some difficult problems that we believe such a code may well be competitive with such codes as PDECOL [20] if more time and programming effort are invested.

In the sequel, we review the collocation method for solving ODE in chapter 2. In section 2.2 we present a new result for the solution of the collocation system arising from using the monomial basis functions. In chapter 3, we describe the numerical methods used in HERPDE. The problem types which can be solved by HERPDE are described in section 3.2, while the temporal discretization and mesh selection algorithm are treated in sections 3.3 and 3.4. Finally, numerical results are presented in chapter 4, and comparisons are made with other methods.

Conclusions drawn from the research and recommendations for future work are given in chapter 5.

2. COLLOCATION METHOD FOR ORDINARY DIFFERENTIAL EQUATIONS

Collocation is the process of finding an approximate solution to a differential equation by requiring that the approximate solution satisfy the differential equation at some discrete set of points, together with the boundary conditions.

The basic ideas involved in collocation are as follows: Consider the BVP

$$y''(x) = f(x,y(x)) \tag{2.1}$$

$$y(a) = y_a$$

$$y(b) = y_b .$$

If y(x) is smooth, then there exists a polynomial

$$p(x) = a_0 + a_1 x + \ldots + a_n x^n$$

that is a good approximation to the solution y(x) of (2.1). In order to determine such a p(x), consider the expression

$$e(x) = p''(x) - f(x,p(x)) \tag{2.2}$$

with the objective of finding the coefficients $a_0, a_1, \ldots a_n$ of p(x) such that the error function, e(x), is as small as possible. These coefficients can be chosen in several ways. The collocation method determines p(x) by requiring $e(x_i) = 0$ at certain selected "collocation points," $x_0, x_1, \ldots, x_n$. Thus, collocation leads to the system of n+1 equations in n+1 unknowns,

$$e(x_i) = 0, \; i=0,1,\ldots,n. \tag{2.3}$$

If f(x,y) is a nonlinear function of y, then (2.3) is a nonlinear system of n+1 equations in n+1 unknowns. One can use Newton's method for solving this system.

A main feature of collocation is that its output is a function (usually a piecewise polynomial) that approximates the solution of a differential equation throughout some interval. That is, collocation can be used to estimate the behavior of the solution in the entire interval in question. This is in contrast to other methods such as finite difference and shooting, which produce a discrete set of solution values.

2.1 Collocation with Piecewise Polynomials (Splines)

For spline collocation, one chooses $p(x)=\sum_{i=0}^{n} a_i \phi_i(x)$ as the basic approximating form where the functions $\{\phi_i(x)\}_{i=0}^{n}$ form a basis of piecewise polynomials, viz B-splines or some other type of basis. Further, by differentiating the equation $y''=f(x,y)$ with respect to x, one could insist that both

$$e(x_i) = p''(x_i) - f(x_i, p(x_i)) = 0$$

and

$$e'(x_i) = p'''(x_i) - f'(x_i, p(x_i)) = 0$$

at the collocation points (this would be a form of Hermite interpolation). For a good exposition on the convergence and stability properties of spline collocation method, the readers are advised to consult [8,26]. Among the advantages of using spline collocation are:

1. Computations involve sparse and banded matrices which are easy to form and solve,
2. Piecewise polynomial functions are readily adaptable to special problems. For example one can place collocation points so as to adapt to known properties of the solution such as boundary layers.
3. The method is stable and mesh selection is generally successful.

We will illustrate the collocation method using the following m th order ODE

$$Lu(x) := D^m u(x) - \sum_{l=1}^{m} c_l(x) D^{l-1} u(x) = f(x), \quad a \leq x \leq b \qquad (2.4)$$

with boundary conditions (BC)

$$B_a z(u(a)) = b_a ,$$

$$B_b z(u(b)) = b_b , \qquad (2.5)$$

where B_a is $m_a \times m$,

B_b is $m_b \times m$,

$m = m_b + m_a$

and $z(u(x)) := (u(x), Du(x), \ldots, D^{m-1}u(x))^T$. (2.6)

It is assumed that a sufficiently smooth, unique solution $u(x)$ to (2.4), (2.5) exists and satisfies

$$\|D^j u\|_\infty := \max_{a \le x \le b} |D^j u(x)|$$

$$\le c \max\{\|f\|, \|b_a\|_\infty, \|b_b\|_\infty\}, \quad 0 \le j \le m-1 \qquad (2.7)$$

for some constant c.

Our collocation procedure is defined as follows. Select a partition

$$\Delta : a = x_1 < x_2 < \ldots < x_N < x_{N+1} = b, \qquad (2.8)$$

and let

$$h_i := x_{i+1} - x_i, \quad 1 \le i \le N, \tag{2.9}$$

$$h_0 := h_1,$$

$$h_{N+1} := h_N,$$

$$h := \max_{1 \le i \le N} h_i .$$

The collocation procedure for solving (2.4),(2.5) determines a piecewise polynomial approximation $u_\Delta(x)$ to the exact solution $u(x)$. $u_\Delta(x)$ is uniquely defined by the following conditions:

(A) $u_\Delta(x)$ is a polynomial of order $k+m$ (degree $k+m-1$) on each subinterval (x_i, x_{i+1}), $1 \le i \le N$,

(B) $u_\Delta(x) \in C^{(m-1)}(a,b)$,

(C) $u_\Delta(x)$ satisfies (2.4) at the kN collocation points

$$\xi_{ij} := x_i + h_i \rho_j, \quad 1 \le j \le k, \quad 1 \le i \le N, \tag{2.10}$$

with ρ_j belong to the partition

$$0 \le \rho_1 < \ldots < \rho_k \le 1, \quad k \ge m, \tag{2.11}$$

(D) $u_\Delta(x)$ satisfies the boundary conditions.

If the k points in (2.11) satisfy the orthogonality relation

$$\int_0^1 P(t) \prod_{j=1}^{k} (t-\rho_j)dt = 0 \qquad (2.12)$$

for all polynomials P(t) of order n $(m \leq n \leq k)$ then

$$\left\| D^j (u-u_\Delta) \right\| = O(h^{k+m-j}), \quad 0 \leq j \leq m \qquad (2.13)$$

and at the mesh points, one gets superconvergence,

$$\left| D^j (u-u_\Delta)(x_i) \right| = O(h^{k+n}), \; 1 \leq i \leq N+1, \quad 0 \leq j \leq m-1, \qquad (2.14)$$

where n is the order of the approximating polynomial (cf. de Boor and Swartz [8].)

If Gaussian points are used, then n=k.

Any representation for $u_\Delta(x)$ which satisfies (A) contains a vector of parameters with $(k+m)N$ components, α, which can be determined by applying conditions (B), (C) and (D). These conditions yield the system of equations

$$A\alpha = f \qquad (2.15)$$

where A is an almost block diagonal matrix:

$$A = \begin{bmatrix} B_a & & & & \\ V_1 & & & & \\ & V_2 & & & \\ & & \ddots & & \\ & & & & V_N \\ & & & & B_b \end{bmatrix} \qquad (2.16)$$

with B_a, B_b defined in (2.5). The N blocks V_i all have the same size and offset, independent of N. The block V_i relating to the i th interval is formed from the collocation equations (C) and/or the continuity conditions (B).

Due to the local nature of the representation, some components of α can be eliminated locally (within each block V_i). This so called condensation of parameters process can effectively reduce the size of A, and in certain cases, reduce the amount of work in computing the components of α.

Notice that the collocation method, unlike the traditional shooting method, does not explicitly convert the m th order ODE into a first order system.

2.2 Local Representation

To compute an approximate solution $u_{\Delta}(x)$ of the boundary value problem, a convenient representation for this function should be chosen. The merits of different local bases have been extensively analyzed in [4], [5]. We will briefly review the local monomial basis and Hermite-type basis representations below. We note here that B-splines are piecewise polynomial functions with minimal support and have continuity conditions implicitly satisfied. A thorough exposition on B-splines and their application in solving (2.4) by collocation can be found in [7].

2.2.1 Monomial Basis

The monomial basis has many favourable properties over B-spline and Hermite bases. Because of the local nature of this basis representation, it allows efficient formation of continuity and discretization equations. Although continuity is not built-in, it is easier to implement than is a B-spline basis using the scheme devised by Ascher, Pruess and Russell [4]. They have also shown that the work required to assemble the collocation matrix A is roughly the same as that for the Hermite-type representation and significantly less than that for the B-spline representation. Unfortunately, the monomial basis requires more storage than the other two. But, this is often a small price to pay for better conditioned matrices. Local condensation of the discretization equations yields multiple-shooting-type matrices which are better conditioned than those for the other two bases.

For efficiency in the implementation, we have adopted the monomial basis representation suggested by Ascher et al. [4]. We choose the monomials

$$\frac{t^{j-1}}{(j-1)!}, \qquad 1\le j\le m \tag{2.17}$$

as the first m canonical polynomials on [0,1].

Thus for $x_i \le x \le x_{i+1}$, $1\le i\le N$,

$$u_\Delta(x) = \sum_{j=1}^{m} \frac{z_{ij}(x-x_i)^{j-1}}{(j-1)!} + h_i \sum_{j=1}^{k} w_{ij}\,\phi_j((x-x_i)/h_i) \tag{2.18}$$

where $\{\phi_j\}_{j=1}^{k}$ are the remaining canonical polynomials of order j+m on [0,1], satisfying

$$\phi_j(t) = t^{m+j-1}/(m+j-1)!, \quad 1\le j\le k, \tag{2.19}$$

Note

$$D^{l-1}\phi_j(0) = 0, \quad l=1,\ldots,m, \quad j=1,\ldots,k. \tag{2.20}$$

Also, let

$$z_{ij} := D^{j-1} u_{\Delta}(x_i), \quad 1 \le i \le N+1, \quad 1 \le j \le m. \tag{2.21}$$

The scale factor h_i^m for w_{ij} in (2.18) is introduced for notational convenience. With this choice, let

$$w_{ij} := h_i^{j-1} D^{m+j-1} u_{\Delta}(x_i), \quad 1 \le j \le k. \tag{2.22}$$

The continuity conditions become

$$z_{i+1} = C_i z_i + D_i w_i, \quad 1 \le i \le N, \tag{2.23}$$

where $w_i = (w_{i1}, \ldots, w_{ik})^T, \quad 1 \le i \le N,$ (2.24)

$$z_i = (z_{i1}, \ldots, z_{im})^T, \quad 1 \le i \le N+1. \tag{2.25}$$

$C_i = C^i_{rj}$ is an mxm upper triangular matrix with entries

$$C^i_{rj} = h_i^{j-r}/(j-r)!, \quad j \ge r, \tag{2.26}$$

and $D_i = D_{rj}^i$ is an mxk matrix with entries

$$D_{rj}^i = h_i^{m-l-r} D^{r-1} \phi_j(1). \tag{2.27}$$

From $$Lu_\Delta(x) = h_i^m \sum_{j=1}^{k} w_{ij} L(\phi_j((x-x_i)/h_i))$$

$$- \sum_{l=1}^{m} c_l(x) \sum_{j=1}^{m} z_{ij} (x-x_i)^{j-l}/(j-l)! \tag{2.28}$$

the collocation conditions give

$$H_i z_i + G_i w_i = f_i, \quad 1 \le i \le N, \tag{2.29}$$

where $H_i = H_{rj}^i$ is a kxm matrix with entries

$$H_{rj}^i = -\sum_{l=1}^{j} c_l(\xi_{ir})(h_i \rho_r)^{j-l}/(j-l)!, \tag{2.30}$$

$G_i = G_{rj}^i$ is a kxk matrix with entries

$$G_{rj}^i = D^m \phi_j(\rho_r) - \sum_{l=1}^{m} c_l(\xi_{ir}) h_i^{m+l-l} D^{l-1} \phi_j(\rho_r), \tag{2.31}$$

and $$f_i = (f(\xi_{i1}),\ldots,f(\xi_{ik}))^T. \tag{2.32}$$

The resulting blocks V_i $(1\leq i\leq N)$ of A have the structure

$$V_i = \begin{array}{c|c} \begin{bmatrix} H_i & G_i & 0 \\ -C_i & -D_i & I \end{bmatrix} & \begin{array}{c} k \\ m \end{array} \\ \begin{array}{ccc} m & k & m \end{array} & \end{array} \qquad (2.33)$$

where I is an mxm identity matrix and V_i are (k+m)x(k+2m).

Figure 1 shows the detailed structure of MONO, a collocation matrix for m=4, k=4, m_a =2, m_b =2 and N=2, where x denotes the potential nonzero entry.

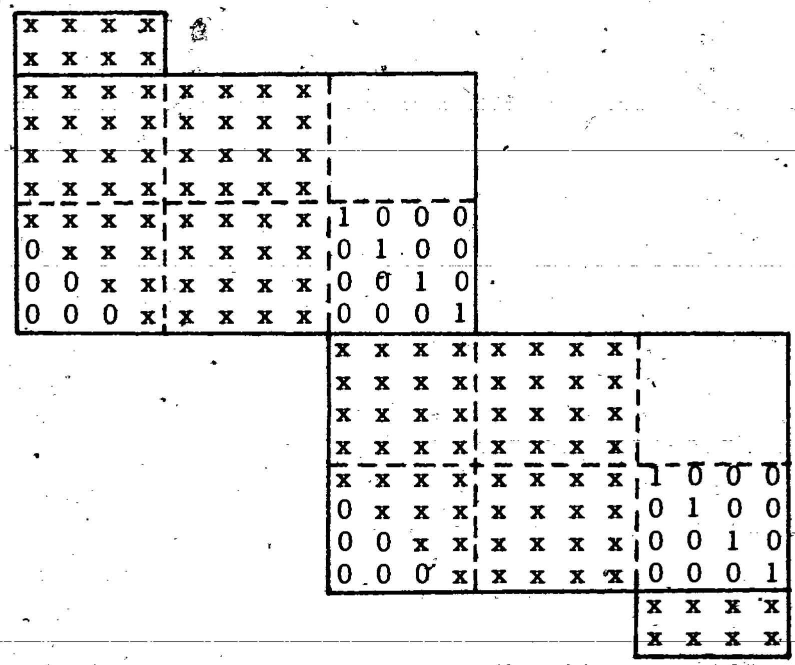


Figure 1 The structure of MONO, the almost block diagonal collocation matrix using local monomials

Because of the size of A, the suggestion in [4] is to perform condensation of parameters to reduce its size and then to solve the resulting linear system. However, we will describe a scheme which, by taking advantage of the sparsity and the block structure of matrix A, requires less computing effort.

2.2.2 Condensation of Parameters

Condensation of parameters can be used to locally eliminate some unknowns, thus reducing storage and the effort required in computing the nodal values because work for other unknowns ignored. In particular, since the unknowns w_{ij}, $1 \leq j \leq k$, which correspond to columns m+1 to m+k of block V_i, appear only in this block, Gaussian elimination (with column pivoting) can be applied to these columns to reduce V_i to the form

$$\overline{V}_i = \begin{bmatrix} \overline{H}_i & \overline{G}_i & 0 \\ -\overline{C}_i & 0 & I \end{bmatrix}, \tag{2.34}$$

where $\overline{G}_i$ is upper triangular. By discarding the top k rows and the middle k columns of each block $\overline{V}_i$, one obtains a new mx2m block $\hat{V}_i$ with the following structure

$$\hat{V}_i = \begin{bmatrix} -\bar{C}_i & I \end{bmatrix}. \tag{2.35}$$

Each block $\hat{V}_i$ offsets m columns from the previous one. Since the reduced collocation matrix A is like a multiple shooting matrix, any linear system solution technique for eliminating a multiple shooting matrix is applicable for computing the nodal values. This implies that LENBLOCK (to be described in the next section) can be applied on the condensed matrix. Note that the new linear system involves only z_{ij}, $1 \leq i \leq N+1$, $1 \leq j \leq m$.

There are a number of ways of obtaining a global solution after computing the unknowns z_{ij}. One can recover the w_{ij} by applying back substitution if the matrices $\bar{H}_i$, $\bar{G}_i$ are saved. An alternative is to use Hermite interpolation.

2.2.3 ASIS

By making use of the structure of A, we can also solve the system "AS IS", that is, without condensation, using alternate row and column elimination. Our scheme, ASIS, is a modified version of LENBLOCK, an alternate row and column elimination scheme, first discussed by Keller and Lentini [16] in the context of applying the box scheme for the numerical solution of a general linear two-point BVP. Russell [25] points out that their scheme can be applied to multiple shooting matrices with less work requirement than most of the existing schemes. After correcting an error in Keller and Lentini's scheme, we have implemented their idea and tested it for a number of problems using the multiple shooting technique (see [17] for a more detailed exposition and some numerical examples.) It is found that LENBLOCK performs no worse than LAMPAK and COLROW [14], two state-of-the-art codes for eliminating matrices with almost block diagonal structure. The main advantage of LENBLOCK is that it can be implemented fairly easily and the resulting code requires slightly less execution time than LAMPAK and COLROW. In addition, it preserves the zero structure of the matrix and introduces no fill-in. We will illustrate the method using a multiple shooting matrix and then show how the same idea can be extended to our collocation system.

A standard multiple shooting matrix has the almost block diagonal structure shown in Figure 2. The matrix has N blocks labelled

$$\begin{bmatrix} M_i & I \end{bmatrix}, \qquad i=1,\ldots,N,$$

each having p rows and 2p columns. N is the number of shooting intervals in the region of interest. The pxp block I represents the identity matrix.

Each block arises from the continuity conditions at a nodal point. The q left boundary conditions determine block BL which has q rows and p columns, while the p-q right hand boundary conditions are represented by the (p-q)xp block BR. An important characteristic of the matrix is that each block "overlaps" the previous one in p columns. Consequently the matrix has dimension px(N+1).

$$\begin{bmatrix} BL & & & & & \\ M_1 & I & & & & \\ & M_2 & I & & & \\ & & \ddots & \ddots & & \\ & & & \ddots & \ddots & \\ & & & & M_N & I \\ & & & & & BR \end{bmatrix}$$

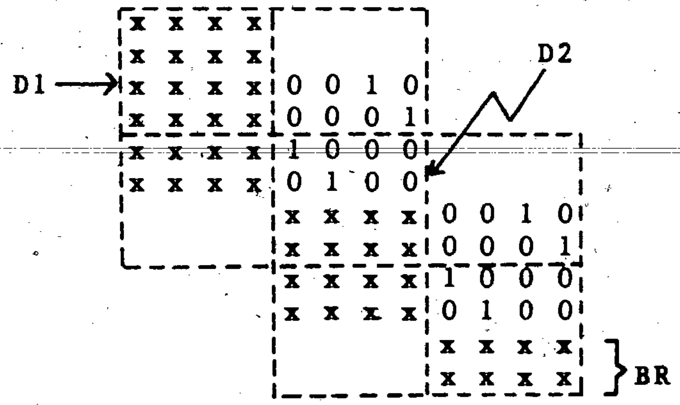
Figure 2 The almost block diagonal structure of a standard multiple shooting matrix

For ease of presentation, a multiple shooting matrix, S, with p=4, q=2 and N=2 will be used to show the results of applying each step of the LENBLOCK scheme. S has the following structure:

$$\begin{array}{|cccc|cccc|cccc|} \hline x & x & x & x & & & & & & & & \\ x & x & x & x & & & & & & & & \\ \hline x & x & x & x & 1 & 0 & 0 & 0 & & & & \\ x & x & x & x & 0 & 1 & 0 & 0 & & & & \\ x & x & x & x & 0 & 0 & 1 & 0 & & & & \\ x & x & x & x & 0 & 0 & 0 & 1 & & & & \\ \hline & & & & x & x & x & x & 1 & 0 & 0 & 0 \\ & & & & x & x & x & x & 0 & 1 & 0 & 0 \\ & & & & x & x & x & x & 0 & 0 & 1 & 0 \\ & & & & x & x & x & x & 0 & 0 & 0 & 1 \\ \hline & & & & & & & & x & x & x & x \\ & & & & & & & & x & x & x & x \\ \hline \end{array}$$

where x represents a potential nonzero element of S.

The LENBLOCK scheme proceeds in two stages, the first being the decomposition of the coefficient matrix S, and the second being the solution of the resulting system. Before the beginning of the first stage, the last p-q rows of each block are moved to the beginning of their block. The resulting structure of S is:



The LENBLOCK scheme treats S as a block tridiagonal system with pxp diagonal blocks (outlined by dotted lines above). In the decomposition phase, the first diagonal block (D1) is subjected to q column eliminations with column interchanges, followed by p-q column eliminations with row interchanges. Then a sequence of column interchanges are performed on the next p columns in order to preserve the zero structure of the coefficient matrix. This pattern is repeated for the remaining diagonal blocks (in this case, D2) except that no column interchanges are performed. Finally, elimination takes place in block BR, at which point p-q column eliminations with row interchanges are performed. Note that the q column eliminations without interchange in any but the first block is designed to preserve the favourable 0-1 structure of A. This allows some saving in computing the multipliers and performing eliminations because some pivotal elements have value 1. Small saving is also gained in the solution stage. In matrix form, this decomposition can be written as

$$S = PLAQ$$

where

1. P is a permutation matrix recording the row pivotal strategy,
2. L is a unit lower triangular matrix containing the row elimination multipliers,
3. A is an upper triangular matrix containing the resulting decomposition,
4. Q is a permutation matrix recording the column pivotal strategy.

In the factorization, the only storage requirements beyond that for S are for 2 vectors used for recording pivotal information. Moreover, the non-zero elements of L can be stored in the original positions in S. For definiteness, we have shown the matrix S after factorization in Figure 3. The elements of L are stored in S, where b represents the potential nonzero element of L and a that of A. The unit diagonal elements of L are not stored. Note that most 0, 1 elements of S are preserved in the decomposition.

With minor modification, the LENBLOCK process can be applied to our collocation system. We will use the previously

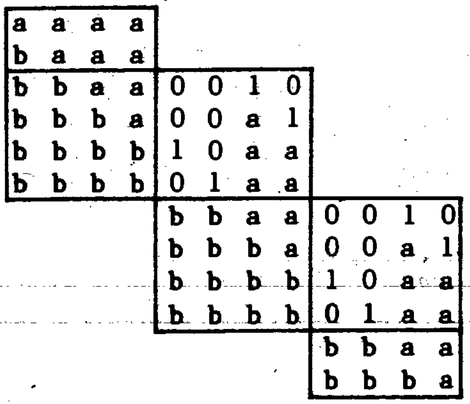
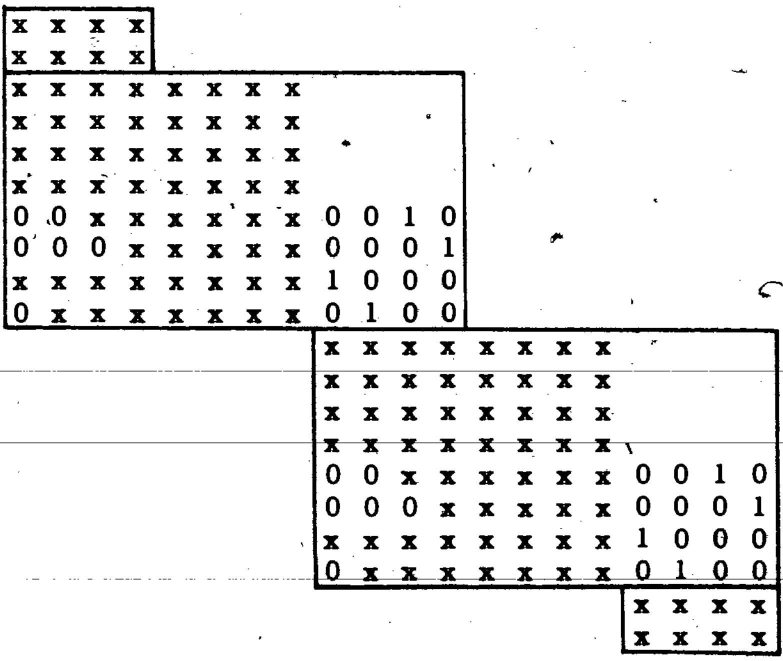


Figure 3 The structure of S after decomposition

introduced collocation matrix MONO to illustrate the process. Our ASIS process begins by transferring rows k+1 to $k+m_a$ of each block V_i, to the bottom of that block. The structure of MONO becomes:



In the decomposition phase, $k+m_a$ column eliminations with column interchanges are performed, followed by m_b column eliminations with row interchanges, and the next m columns are interchanged to preserve the structure of 0 and 1's. This pattern repeats for all the other blocks except no column interchanges are done for the first m_a rows.

It follows that the previous analysis is also applicable in this case. The advantages of ASIS are:

1. No extra storage is required besides that for storing permutation information,
2. Programming this scheme is relatively simple,
3. It requires less work and thus less time to compute the solution than using condensation of parameters (see next section.)

2.2.4 Operation Counts

We will compare the relative efficiency of the condensation of parameters and the ASIS schemes with respect to operation counts. The work estimates count only multiplications and divisions. The total operation count for condensation of parameters plus linear system solution using the LENBLOCK scheme is

$$W_{CP} = (k^3/3 + (m+1)k^2 + (m^2+2m-1/3)k + m^3/2 +$$

$$(m_a^2/2 + 3/2)m^2 + (m_a^2 - 3m_a/2 - 1)m + m_a^3/2$$

$$-3m_a^2/2 + m_a)N. \qquad (2.36)$$

Note that the above operation count includes the work for computing the unknowns w_{ij}.

The operation count for the solution using the ASIS scheme is

$$W_{AS} = (k^3/3 + (m+1-m_a/2)k^2 + (m^2+2m-1/3-m_a^2+m_a/2)k +$$

$$m^3/2 + (m_a/2 + 3/2)m^2 + (m_a^2 - 3m_a/2-1)m + m_a^3/2$$

$$- 3m_a^2/2 + m_a)N. \qquad (2.37)$$

One can see that W_{AS} is less than W_{CP} in all cases. It is instructive to compare our operation counts to the one given by Ascher et al. [4] for collocation with monomial basis. Using Gaussian elimination with scaled partial pivoting and taking no advantage of the special structure of the reduced matrix A, they obtain the following work estimate:

$$W_1 = (k^3/3 + (m+1)k^2 + (m^2+m-1/3)k + 5m^3/6 +$$

$$3(m_a+1)m^2/2 + (3m_a/2+5/3)m)N. \qquad (2.38)$$

Table 1 tabulates the typical operation counts for different values of k, m and m_a for the three schemes presented above. Both W_{AS} and W_{CP} are significantly less than W_1 in all cases, showing that big savings can be made by tailoring the solution scheme to the structure of the collocation matrix. W_{AS} is slightly smaller than W_{CP} in most cases, with larger saving when m_a is close to m.

Table 1

Operation counts (N factor omitted)

k	m	m_a	W_{AS}	W_{CP}	W_1
2	2	1	36	39	51
1	3	1	45	46	75
1	3	2	36	40	93
3	2	1	62	68	78
2	3	1	72	75	101
2	3	2	59	69	119
1	4	1	86	87	139
1	4	2	77	81	169
1	4	3	57	66	199
4	2	1	99	109	117
3	3	1	110	116	139
3	3	2	92	110	157
2	4	1	125	128	176
2	4	2	112	122	206
2	4	3	86	107	236
1	5	1	145	146	231
1	5	2	137	141	276
1	5	3	115	124	321
1	5	4	82	98	366
5	2	1	149	164	170
4	3	1	161	171	191
4	3	2	137	165	209
3	4	1	177	183	227
3	4	2	159	177	257
3	4	3	126	162	287
2	5	1	198	201	281
2	5	2	186	196	326
2	5	3	158	179	371
2	5	4	117	153	416
1	6	1	225	226	356
1	6	2	219	223	419
1	6	3	196	205	482
1	6	4	159	175	545
1	6	5	111	136	608

2.2.5 Hermite Basis

The Hermite-type basis is a local representation which allows the use of the same information (with some scaling) in each subinterval of a mesh. The local representations for neighboring subintervals are matched to allow continuity conditions to be implicitly satisfied. Thus, only the discretization equations need be explicitly satisfied. The following treatment is similar to that of Ascher et al. [4]. We define a canonical set of k+m polynomials $\phi_j(t)$ of order k+m over the interval [0,1]. Each $\phi_j(t)$ has the property

$$\lambda_l \phi_j = \delta_{lj}, \quad 1 \le l, j \le k+m, \tag{2.39}$$

where δ_{lj} is the Kronecker's constant and the linear functionals λ_l are defined with respect to the points

$$\eta_0 < 0 \le \eta_1 \le \eta_2 \le \dots \le \eta_{k+m} \le 1 \tag{2.40}$$

as follows: If

$$\eta_{l-q-1} < \eta_{l-q} = \eta_{l-q+1} = \dots = \eta_l,$$

then

$$\lambda_l \phi_j := D^q \phi_j(\eta_l), \quad 1 \le j \le k+m. \tag{2.41}$$

In order to obtain a nodal method, we choose

$$\eta_1 = \eta_2 = \ldots = \eta_m = 0. \tag{2.42}$$

A nodal method computes some or all of the components of α in such a way that they are proportional to the superconvergent values

$$z_{ij} := D^{j-1} u_\Delta(x_i), \quad 1 \le i \le N+1, \quad 1 \le j \le m. \tag{2.43}$$

To map the k+m polynomials into each subinterval of the mesh Δ and to have continuity implicitly satisfied, we express u_Δ as

$$u_\Delta(x) = \sum_{j=1}^{k+m} \alpha_{(i-1)k+j} \, S_{ij} \, \phi_j((x-x_i)/h_i), \quad x_i \le x \le x_{i+1}, \tag{2.44}$$

where

$$S_{ij} \, \alpha_{(i-1)k+j} = h_i^{j-1} z_{ij}, \quad 1 \le j \le m, \quad 1 \le i \le N \tag{2.45}$$

and S_{ij} are scaling factors. The choice of the S_{ij} is equivalent to the column scaling of A. This is required because the representation (2.44) can be unbalanced as $h \to 0$. Note that the standard Hermite basis local representation has k=m.

After setting $\eta_{k+1}=\eta_{k+2}=\ldots=\eta_{k+m}=1$, (2.39) and (2.41) give

$$D^q u_\Delta(x_i+\eta_1 h_i) = \frac{\alpha_{(i-1)k+j}\, S_{ij}}{h_i^q}, \quad 1\le j\le k+m, \quad 1\le i\le N. \qquad (2.46)$$

Thus, the continuity conditions provide a constraint on the scale factors S_{ij}:

$$\frac{S_{ij}}{h_i^{j-1}} = \frac{S_{i-1,k+j}}{h_{i-1}^{j-1}}, \quad 1\le j\le m, \quad 1\le i\le N. \qquad (2.47)$$

There are many possible choices of S_{ij} compatible with (2.41). For practical reasons, we use the following formulas:

$$S_{ij} := \begin{cases} \left(\dfrac{h_i}{h_{i-1}}\right)^{j-1}, & 1\le j\le m, \quad h_i<h_{i-1} \\ \left(\dfrac{h_i}{h_{i+1}}\right)^{j-k-1}, & k\le j\le k+m, \quad h_i<h_{i+1} \\ 1, & \text{otherwise.} \end{cases} \qquad (2.48)$$

This choice produces $S_{ij}\le 1$ and $S_{ij}=1$ whenever possible and it gives nearly balanced scaling for highly nonuniform meshes. At each collocation point, (2.44) gives

$$D^{l} u_{\Delta}(\xi_{ir}) = \sum_{j=1}^{k+m} \alpha_{(i-1)k+j} \frac{S_{ij} D^{l} \phi_{j}(\rho_{r})}{h_{i}^{l}},$$

$$1 \leq r \leq k, \quad 1 \leq i \leq N, \quad 0 \leq l \leq m. \tag{2.49}$$

Since the constants $D^{l}\phi_{j}(\rho_{r})$ are mesh independent, they can be evaluated and stored at the beginning. We use the divided difference table with respect to the points η_{l} to evaluate $\phi_{j}(t)$ and its derivatives at each point ρ_{r} using the Newton form of the interpolating polynomial and nested multiplication. Additional details can be found in [10].

From (2.4) and (2.49), the entries of $V_{i} = V^{i}_{rj}$ are the coefficients of $\alpha_{(i-1)k+j}$ in $Lu_{\Delta}(\xi_{ir})$,

$$V^{i}_{rj} = S_{ij} \{h_{i}^{-m} D^{m} \phi_{j}(\rho_{r}) - \sum_{l=1}^{m} c_{l}(\xi_{ir}) h_{i}^{l-1} D^{l-1} \phi_{j}(\rho_{r})\}$$

$$1 \leq j \leq k+m, \quad 1 \leq r \leq k, \quad 1 \leq i \leq N. \tag{2.50}$$

The block diagonal structure of HER7, a collocation matrix using the Hermite basis of degree 7 ($k+m=8$, $k=4$, $m=4$, $m_{a}=2$, $N=2$) is shown in Figure 4.

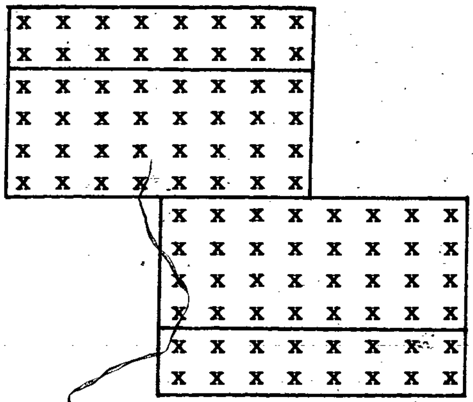


Figure 4 The block structure of HER7, a Hermite basis collocation matrix

2.3 Solution of Nonlinear Boundary-Value Problem

The basic existence and uniqueness theory of nonlinear BVP is not as developed as for IVP or linear BVP. A common method is to discretize the linear differential equation with a finite difference method and solve the resulting nonlinear algebraic or transcendental equations by Newton's method or its variations [18]. Another possibility is to first linearize by quasilinearization [6], and then solve a sequence of linear BVP's. The method of quasilinearization is actually an application of Newton's method to the nonlinear differential operator.

Consider the nonlinear BVP

$$U^{m} = F(x,U,U^{1},U^{2},\dots,U^{m-1})$$

$$g(x,U,U^{1},\ldots,U^{m-1}) = 0. \qquad (2.51)$$

Applying the method of quasilinearization to both the differential equation and the boundary conditions, (2.51) becomes

$$U^{m}_{j+1} - \sum_{i=0}^{m-1}\left(\frac{\partial F}{\partial U^{i}_{j}}\right)U^{i}_{j+1} = F(x,U_{j},\ldots,U^{m-1}_{j}) - \sum_{i=0}^{m-1}\left(\frac{\partial F}{\partial U^{i}_{j}}\right)U^{i}_{j} \qquad (2.52)$$

$$\sum_{i=0}^{m-1}\left(\frac{\partial g}{\partial U^{i}_{j}}\right)U^{i}_{j+1} = \sum_{i=0}^{m-1}\left(\frac{\partial g}{\partial U^{i}_{j}}\right)U^{i}_{j} - g(x,U_{j},U^{1}_{j},\ldots,U^{m-1}_{j}).$$

Here, $\partial F/\partial U^{i}_{j}$ and $\partial g/\partial U^{i}_{j}$ represent partial derivatives with respect to U^{i} calculated at the jth iteration.

We have chosen this method for our investigation because it is natural and requires minimal additional work. By adding an iteration control routine and a linearization routine to a linear problem solver, one obtains a program which can solve nonlinear problems.

Usually, the sequence of functions $\{U_{j}(x)\}$ converges to the solution of (2.51) if an appropriate initial approximation $U_{0}(x)$ is chosen. Unfortunately, the problem of finding $U_{0}(x)$ which will ensure convergence can be difficult. In selecting $U_{0}(x)$, one generally incorporates some partial knowledge of the solution of (2.51) or some of its properties, if possible.

Let $\underset{\sim}{v}^i$ be the ith Newton iterate and assume that the exact solution satisfies $\underset{\sim}{u} = \lim_{i\to\infty} \underset{\sim}{v}^i$. One property of a superlinearly convergent method is

$$\lim_{i\to\infty} \frac{\|\underset{\sim}{v}^{i+1} - \underset{\sim}{v}^{i}\|_\infty}{\|\underset{\sim}{v}^{i} - \underset{\sim}{u}\|_\infty} = 1. \qquad [13]$$

Thus, we assume that in the limit $\underset{\sim}{v}^{i+1} - \underset{\sim}{v}^{i}$ is a good measure for $\underset{\sim}{v}^{i} - \underset{\sim}{u}$. Therefore, the reasonable convergence criterion used for our nonlinear iteration is

$$\sum_{j=1}^{N} \left| D^l v_j^{i+1} - D^l v_j^{i} \right| \le TOL_l . N \qquad l=0,1,\ldots,m-1$$

$$v_j^i = v^i(x_j)$$

where N is the number of subintervals and TOL_l is the tolerance for the lth derivative of u. We have chosen this criterion for ease of use.

3. DESIGN OF HERPDE

3.1 Motivation

The method known as the method of lines (MOL) has been used for some fifty years in the Soviet Union. Apparently this process was first applied by E. Rothe [24] in 1930 to equations of parabolic type. But, the MOL has been successfully applied to other types of equation such as elliptic and hyperbolic. The basic feature of the method is that one of the independent variables is discretized and its derivatives are replaced by finite difference approximations, while the other variables and their corresponding derivatives remain continuous. The original differential problem is thus reduced to a new problem of lower dimension. If a two-dimensional problem is given in the rectangle R $\{0 \leq x \leq X,\ 0 \leq y \leq Y\}$, then one can seek an approximate solution on some lines parallel to one of the axes (hence the name). At each line the derivatives normal to that line would be replaced by finite differences and the other variable left continuous. The original system of partial differential equations is now transformed into a system of ordinary differential equations.

There are two possible treatments of the "reduced" problem. One can view it as a set of IVP's, the other as a sequence of two-point BVP's. The former method has been the popular choice [20], [28] because reliable and well-established techniques for solving IVP's are readily available. However, due to the recent advance in boundary-value techniques such as the finite difference, shooting and finite-element methods, we feel that the second approach merits some investigation.

Our approach involves the discretization of the PDE in time to yield a sequence of two point BVP's at successive time levels. This should permit greater flexibility in the spatial discretization. It also organizes the computations so that the linear algebra is tied much more closly to the spatial discretization and is not separated by an ODE integrator.

To our knowledge, there has been no attempt, prior to this work, in solving the "reduced" problem numerically using collocation with a Hermite basis. However, a feasibility study of the use of COLSYS in the solution of systems of partial differential equation has been done by Corless [11]. COLSYS is a robust general purpose boundary value problem solver using collocation with B-splines. It was developed by Uri Ascher of the University of British Columbia and Jan Christiansen and Robert D. Russell of Simon Fraser University [3].

The reader may wonder why we have chosen the Hermite basis in preference to the monomial basis after we had presented many favourable features of the monomial basis. According to our discretization scheme (to be discussed in the following sections), applying spline collocation with the monomial basis to the resulting BVP's no longer yields multiple-shooting like matrices when condensation of parameters is used. Because the main advantage of using a monomial basis, namely, the better conditioned matrix than those of Hermite and B-spline, is lost in our formulation of the problem, we decided to use the Hermite basis instead. Recall that the Hermite collocation matrix requires less storage than that of the monomial basis.

3.2 Description of Problem

The class of problem which can be solved by HERPDE may be represented by the system of partial differential equations:

$$E(U)U_t + (D(U)U_x)_x + F(U,U_x) + C = 0 \qquad (3.1)$$

with initial-boundary conditions on the strip $[a,b] \times [0,\infty)$ given by

$$B(U(a,t),U(b,t),t) = 0, \qquad (3.2)$$

$$U(x,0) = g(x).$$

Here, $U:=U(x,t)$ is a p-dimensional vector, E and D are positive definite diagonal matrices, F is a vector which may depend on x and t, C is a constant vector, B is a vector function giving the separated boundary conditions and g is the initial function.

Problems of the above form arise in many applications such as modeling heat conduction, bacterial motion, combustion, chemical reactions, population dynamics and convecting flows. Thus, a robust code to solve (3.1)-(3.2) efficiently would be highly desirable.

3.3 Temporal Discretization

Once the time step size is chosen, the time derivative $U_t(x,t)$ is replaced by a linear combination of $U(x,t)$ values at previous time steps. In the present investigation, the 1-step, 2-step and 3-step backward difference formulae, Crank-Nicolson and Adams-Moulton 2-step and 3-step formulae are used. These can be expressed as

$$U_t(x,t_i) = \frac{a}{h_i} U(x,t_i) - \frac{1}{h_i} \sum_{j=1}^{k} b_j U(x,t_{i-j}) \qquad (3.3)$$

where h_i is the step size at time t_i; a, b_j are some constants. For k=1, $a=b_1=1$, we have the familiar 1-step backward difference formula.

The original system of partial differential equations (3.1) is transformed into the following sequence of boundary value problems on [a,b],

$$E(U(x,t_i))(\frac{a}{h_i} U(x,t_i) + \frac{1}{h_i} \sum_{j=1}^{k} b_j U(x,t_{i-j})) + (D(U(x,t_i))U_x(x,t_i))_x$$

$$+ F(U(x,t_i), U_x(x,t_i)) + C = 0. \tag{3.4}$$

If the $U(x,t_{i-j})$, $j=1,\ldots,k$ are known and the diagonal matrix D is nonsingular, (3.4) is an ordinary system of two-point boundary value problems for $U(x,t)$ at a constant time. That is, the solution of the original problem can be found by solving the sequence of boundary value problems on the chosen temporal mesh (the lines $t=t_k$, $k=1,2,\ldots$).

The solution process for the resulting boundary value problems involves using the collocation technique outlined in chapter 2.

3.4 Spatial Mesh Selection

The adaptive meshes (both spatial and temporal) are most useful in problems which have solution components with widely differing time scales, or a moving subregion (such as a shock or contact discontinuity). With the adaptive capability, the mesh points can be placed near the area of rapid change and thus minimize the necessary computational effort.

Most problems of the form (3.1) have solutions which contain sharp transitions such as shock layers, boundary layers or wave fronts. It is desirable for the spatial mesh to adapt itself to the solution profile at each time step. This mesh should also be able to follow the sharp transition as the solution evolves.

Adaptive mesh selection strategies usually require some recomputation of the solution. Because of the expense involved in recomputing the solution at possibly every time step, we have used an algorithm which uses a fixed number of mesh points. Our algorithm initially places an equally space meshed point and then attempts to move them so that their locations become close to optimal. Algorithm of this type have been used by Davis and Flaherty [12].

The following well-known result (cf. eg. Pereyra and Sewell [22]) indicates how one should select a proper mesh.

Lemma. Let $\Delta_N := \{a=x_1<x_2<\dots<x_{N+1}=b\}$ be a partition of $[a,b]$ into N subintervals, and let $u(x)\in C^{l+1}[a,b]$. The piecewise polynomial of degree l on (x_i, x_{i+1}), $i=1,2,\dots,N$, that interpolates to u on Δ_N has minimal L_2-error when the knots x_i, $i=2,3,\dots,N$, are chosen such that

$$h_i^{l+1}\left|u^{(l+1)}(\xi_i)\right| = E, \quad i=1,2,\dots,N, \tag{3.5}$$

where $u^{(l)}$ is the lth derivative of u with respect to x, $\xi_i\in(x_i, x_{i+1})$, E is a constant and

$$h_i = x_{i+1} - x_i . \tag{3.6}$$

Thus, the interpolation error is minimized if the partition is chosen in such a way that the quantity $h_i^{l+1}\left|u^{(l+1)}(\xi_i)\right|$ is equidistributed. Lentini and Pereyra [19] and Ascher, Christiansen and Russell [3] have applied this result to implement adaptive grid algorithms for two-point boundary value problems with great success.

Rather than work with (3.5) directly, we use a modified version of the adaptive algorithm developed by Davis and Flaherty [12].

Let

$$p_i := h_{i+1}/h_i = \left(\left|u^{(l+1)}(\xi_i)/u^{(l+1)}(\xi_{i+1})\right|\right)^{1/(l+1)},$$

$$i=1,2,\ldots,N-1, \tag{3.7}$$

where l=3 for piecewise cubic approximations, and let

$$h_1+h_2+\ldots+h_N = x_{N+1} - x_1 . \tag{3.8}$$

Define

$$z := 1+(p_1)+(p_1 p_2)+\ldots+(p_1 p_2 p_3 \ldots p_{N-1})$$

$$= (h_1+h_2+\ldots+h_N)/h_1$$

$$= (x_{N+1}-x_1)/h_1 . \tag{3.9}$$

Equations (3.6),(3.7),(3.9) allow one to determine h_i, $i=1,2,\ldots N$, and x_i, $i=1,2,\ldots,N+1$, in terms of $u^{(l+1)}$ without knowing E. But, $u^{(l+1)}$ is unknown and must be approximated by $u^{(l)}$. The manner we approximate $u^{(l+1)}$ distinguishes our scheme from that of Davis and Flaherty.

Due to the proven success of the adaptive mesh selection scheme of COLSYS, we have used its approximation to compute $u^{(l+1)}$. In particular, given a mesh $\Delta:=\{a=x_1<x_2<\ldots<x_{N+1}=b\}$ and an approximate collocation solution v, an accurate approximation for $u^{(l+1)}$ can be constructed as follows [2]:

Let $$\hat{u}(x_i) := 2\left|v^{(l)}(x_i)-v^{(l)}(x_{i-1})\right|/(x_{i+1}-x_{i-1}) \qquad (3.10)$$

and define $\hat{u}(x)$ by

$$\hat{u}(x) = \begin{cases} \hat{u}(x_i) & x\in[x_i,x_{i+1}],\ i=2,3,\ldots,N \\ \hat{u}(x_2) & x\in[x_1,x_2]. \end{cases} \qquad (3.11)$$

For the ODE one can show that

$$\left|u^{(l+1)}(x)\right| = \hat{u}(x) + O(h) \qquad (3.12)$$

where $h = \max_{1\le i\le N} h_i$.

In solving for the optimal mesh, one must work on the nonlinear equations (3.6),(3.7),(3.9). Some iterative method is most appropriate. We use a slightly modified relaxation scheme of Davis and Flaherty. Although their scheme was used with the Galerkin method, we believe that it should work well with collocation. Further, the coding of this solution technique appears to be straightforward. We briefly give the algorithm for computing h_i, x_i below

```
1.  $R=1$ (R is a relaxation parameter)

2.  $x_i^1 = x_i^0$, $i=1,2,...,N+1$ (initial guess for the optimal
                                     mesh)

3.  $z^0 = (x_{N+1}^0 - x_1^0)/(x_2^0 - x_1^0)$

4.  $z^1 = z^0 + 2e$  (e is a convergence tolerance)

5.  COMPUTE $u^{(l+1)}(x_i^0)$   $i=1,2,...,N+1$

6.  FOR $j=2,3,...,J$ DO:

     6a.  IF $|z^{j-1} - z^{j-2}| \leq e$ RETURN

     6b.  CALCULATE $u^{(l+1)}(x_i^{j-1})$ by linear interpolation of

                $u^{(l+1)}(x_i^0)$,  $i=1,2,...,N+1$
```

```
6c. CALCULATE $p_i^j = \left| u^{(l+1)}(x_{i+1}^{j-1})/u^{(l+1)}(x_{i+2}^{j-1}) \right|^{1/(l+1)}$

        $i=1,2,\ldots,N-1$

6d. $\hat{z}^j = 1+p_1^j+(p_1^j p_2^j)+\ldots+(p_1^j p_2^j \ldots p_{N-1}^j)$

6e. IF $\left|\hat{z}^j - z^{j-1}\right| > \left|z^{j-1} - z^{j-2}\right|$ THEN $R=R/2$

6f. $h_1^j = (x_{N+1}^{j-1} - x_1^{j-1})/\hat{z}^j$

6g. $x_1^j = \hat{x}_1^j = x_1^{j-1}$

6h. $x_{N+1}^j = \hat{x}_{N+1}^j = x_{N+1}^{j-1}$

6i. FOR $i=1,2,\ldots,N-1$ DO:

        $\hat{x}_{i+1}^j = \hat{x}_i^j + h_i^j$

        $h_{i+1}^j = h_i^j p_i^j$

        $x_{i+1}^j = R\hat{x}_{i+1}^j + (1-R)x_{i+1}^{j-1}$

6j. $z^j = (x_{N+1}^j - x_1^j)/(\bar{x}_2^j - x_1^j)$
```

For vector systems, we define

$$p_i^q := \sum_{j=1}^{M} \left| u_j^{(l+1)}(x_{i+1}^{q-1}) \right|^{1/(l+1)} \Big/ \sum_{j=1}^{M} \left| u_j^{(l+1)}(x_{i+2}^{q-1}) \right|^{1/(l+1)} \qquad (3.13)$$

where u_j is the jth component of u.

Since this new p_i^q satisfies (3.9), the above procedure can be directly applied to vector systems.

3.5 Programming Considerations

3.5.1 Supplying Previous Time Step Solutions

In converting the PDE to an ODE, our approach requires that the solution values at previous time steps are known. That is, HERPDE must store the previous time step solutions. The number of time step solutions stored depends on the discretization formulae being used. In the present investigation, the following formulae are used.

Let $u_t(x,t) = f(x,t,u,u_x,u_{xx})$ and write

$$f(x,t) = f(x,t,u,u_x,u_{xx})$$

then

(BD1) 1-step backward difference formula of order Δt

$$u(x,t) - u(x,t-\Delta t) \doteq \Delta t\ f(x,t)$$

(BD2) 2-step backward difference formula of order 2 in Δt

$$1.5u(x,t) - 2u(x,t-\Delta t) + 0.5u(x,t-2\Delta t) \doteq \Delta t\ f(x,t)$$

(BD3) 3-step backward difference formula of order 3 in Δt

$$11/6u(x,t) - 3u(x,t-\Delta t) + 3/2u(x,t-2\Delta t) - 1/3u(x,t-3\Delta t) \doteq \Delta t\ f(x,t)$$

(CN) Crank-Nicolson scheme of order 2 in Δt

$$u(x,t) - u(x,t-\Delta t) \doteq (f(x,t) + f(x,t-\Delta t))\Delta t/2$$

(AM2) Adams-Moulton 2-step, order 3 in Δt

$$u(x,t) - u(x,t-\Delta t) \doteq (5/12\ f(x,t) + 8/12\ f(x,t-\Delta t) - 1/12\ f(x,t-2\Delta t))\Delta t$$

(AM3) Adams-Moulton 3-step, order 4 in Δt

$$u(x,t) - u(x,t-\Delta t) \doteq (9/24\ f(x,t) + 19/24\ f(x,t-\Delta t) - 5/12\ f(x,t-2\Delta t) + 1/24\ f(x,t-3\Delta t))\Delta t$$

One can immediately see that formulae (CN), (AM2), (AM3) require additional function evaluations. The higher the order of the method, the more function evaluations and storage needed. Table 2 gives the computation and storage counts for the six schemes listed above. s represents the number of Gaussian points in the whole interval which is kN, where k is the number of Gaussian points per subinterval and N is the number of subintervals. c stands for the total number of coefficients of the basis functions which is (k+m)N, where m is the order of the ODE. c is roughly twice the size of s in our scheme. For the adaptive mesh, it is not sufficient to store just the solution values calculated on the previous meshes because a new mesh may be used in the current time step. Thus, one must save the coefficients of the basis functions and compute the solution values or the

Table 2

Storage and computation counts for the six discretization schemes

		BD1	BD2	BD3	CN	AM2	AM3
Fixed Spatial Mesh	previous time step solution stored	s	2s	3s	s	s	s
	previous time step function stored				s	2s	3s
	total storage	s	2s	3s	2s	3s	4s
	solution computed	s	s	s	s	s	s
	function evaluation				s	s	s
	total amount of computation	s	s	s	2s	2s	2s
Adaptive Spatial Mesh	previous time step basis coeff. stored	c	2c	3c	c	2c	3c
	solution computed	s	2s	3s	s	s	s
	function evaluation				s	2s	3s
	total amount of computation	s	2s	3s	2s	3s	4s

function values when needed.

In both the fixed and adpative spatial mesh cases, AM3 requires the most storage and is the most expensive. In general, if one wants to have a more accurate solution, one must be willing to use more storage, do more computation and possibly take up more execution time to compute a solution. In terms of storage requirements, the amount of computation and the accuracy of the scheme, CN seems to be the optimal choice if high accuracy is not required. Furthermore, its storage and computation counts are roughly the same for both cases.

3.5.2 Generation of Gaussian Points

The Gaussian points are the zeros of the Legendre polynomials. Given the first two Legendre polynomials, $P_1(x)$ and $P_2(x)$, the successive Legendre polynomials are related according to the following recursion:

$$(n+1)P_{n+1}(x) = (2n+1)xP_n(x) - nP_{n-1}(x), \quad n=2,3,\ldots \qquad (3.14)$$

Our objective is to have our code generate the Gaussian points automatically. One advantage is that the Gaussian points can be determined to the machine precision, independent of the machine used. Although the following scheme is not the most efficient method, it is fairly easy to program. The Legendre polynomial of degree N yields N Gaussian points in (-1,1). If N is odd, $\rho_0=0$ will be a Gaussian point while the other Gaussian points are symmetric with respect to 0. If N is even, all Gaussian points are symmetric with respect to 0.

We use the Illinois root finding method to iteratively compute the Gaussian points. The method computes the successive approximations to a root of f(x) as follows:

$$x_{i+1} = x_i - f(x_i)(x_i - x_{i-1})/(f(x_i) - f(x_{i-1})) \qquad (3.15)$$

If $f(x_{i+1})f(x_i)<0$, replace $(x_{i-1}, f(x_{i-1}))$ by $(x_i, f(x_i))$, else replace $(x_{i-1}, f(x_{i-1}))$ by $(x_{i-1}, f(x_{i-1})/2)$.

In order to ensure computing all the Gaussian points, we let $f(t) = P_n(t)$

and choose

$$x_0 = ih \text{ and } x_1 = (i+1/2)h \quad (i=1,2,\ldots,N),$$

where $h = 1/N$. (3.16)

The iteration cycle starts with the knowns x_{i-1}, x_i, $P_0(x_{i-1})$, $P_0(x_i)$, $P_1(x_{i-1})$ and $P_1(x_i)$. (3.14) is recursively applied until $P_n(x_i)$ is computed. Formula (3.15) is then used to compute a better approximation to a Gaussian point. The iteration is repeated with the new x_{i-1}, x_i, $f(x_{i-1})$, $f(x_i)$ until convergence is obtained.

The convergence criterion used is $P_n(\rho_i)<1x10^{-9}$. This process would give N Gaussian points with some of them having identical values in [0,1). An elimination step is then used to get rid of duplications, resulting in [N/2] distinct nonzero Gaussian points. (Here, "[ ]" denotes the largest integer less than or equal to N/2.) The Gaussian points located in (-1,0] are obtained by negating the nonzero Gaussian points in [0,1).

We have found that this process works well for N up to 50.

3.6 Preliminary Experiments

In this section, we describe some preliminary experiments in our investigation of the usefulness of Hermite collocation in solving the parabolic equations. We examine the effect of the order of piecewise polynomial on the execution time and the accuracy of the solution in section 3.6.1. Results from five linear and two nonlinear ordinary BVP's are presented. Section 3.6.2 is devoted to the testing of our mesh selection algorithm which seems to work well.

3.6.1 Effects of the Degree of Basis Function

The question of what order piecewise polynomials to use has not received much attention. The higher the order of the piecewise polynomial, the more the number of unknown coefficients in the basis representation; thus, for a fixed mesh the execution time of the program will increase since a larger linear system is being solved. However, a higher order polynomial can better approximate the exact solution on a given mesh.

We have designed a spline collocation solver with Hermite basis (HCOL) for solving ordinary boundary value problem. The details of implementation are given in chapter 2. HCOL is similar to COLSYS except that it only solves the problem once. HCOL computes a solution using an uniform mesh and then uses the spatial mesh selection scheme outlined in section 3.4 to compute an optimal mesh. It does not recompute the solution on the new mesh found. The results given below and in the next chapter were obtained on an IBM 4341 in double precision arithmetic (16 decimal digits). The computer times are in seconds and were obtained using the internal clock. Due to the multiprogramming environment of the IBM 4341, the measured time may vary by as much as 10 percent for different executions. The error listed here is the largest error measured over the eleven mesh points used in each region of interest. The almost block diagonal linear system solver used was COLROW [14].

The seven test problems are given below.

Problem 1:

$$y'' = y$$
$$y(0) = 1 \qquad y'(b) = 1$$

Solution: $y(x) = (e^{-b}+1)e^{(x-b)}/(1+e^{-2b}) + (e^{b}-1)e^{-(x+b)}/(1+e^{-2b})$

b = 1 was used.

Problem 2: Osborne [21]

$$y'' = (1 + x^2)y$$
$$y(0) = 1 \qquad y(b) = 0$$

Solution: $y(x) = e^{x^2/2}(1-\mathrm{erf}(x)/\mathrm{erf}(b))$

b = 1 was used.

Problem 3: Bulirsch-Stoer [9]

$$y'' = 400y + 400\cos^2(\pi x) + 2\pi^2 \cos(2\pi x)$$
$$y(0) = y(1) = 0$$

Solution: $y(x) = (e^{(20x-20)} + e^{-20x})/(1+e^{-20}) - \cos^2(\pi x)$

Problem 4:

$$y''' = 20y'' + y' - 20y$$
$$y(0) = 0.1 + 0.1e^{-b} + e^{-20b}$$
$$y(b) = 1.1 + 0.1e^{-b}$$
$$y'(b) = 20.1 - 0.1e^{-b}$$

Solution: $y(x) = 0.1e^{(x-b)} + e^{20(x-b)} + 0.1e^{-x}$

$b = 1$ was used.

<u>Problem</u> <u>5</u>: Russell-Shampine [26]

$$y'' = e^{y}$$
$$y(0) = y(1) = 0$$

Solution: $y(x) = 2 \ln(c \sec(c(x-0.5)/2)) - \ln 2$

$c = 1.336056$

<u>Problem</u> <u>6</u>:

$$y'' = -yy'/\cos(x)$$
$$y(0) = 0 \qquad y(\pi/4) = 0.7071$$

Solution: $y(x) = \sin(x)$

<u>Problem</u> <u>7</u>:

$$u'' = u + 2v'$$
$$v'' = v' + u - v$$
$$u(0) = 0 \qquad u(\pi/2) = 1$$
$$v(0) = 1 \qquad v(\pi/2) = 0$$

Solution: $u(x) = \sin(x)$
$v(x) = \cos(x)$

Table 3

Variation of execution time with degree of basis function

DEGREE OF POLYNOMIAL	PROBLEM					
	1	2	3	4*	5	6
3	.24	.32	.32		.42	.33
4	.30	.32	.31		.46	.48
5	.41	.40	.37	.34	.64	.70
6	.46	.44	.46	.40	.93	.97
7	.50	.50	.53	.49	1.27	1.32
8	.64	.62	.64	.61	1.73	1.80
9	.87	.75	.77	.75	2.32	2.34

Table 4

Variation of absolute error with degree of basis function

DEGREE OF POLYNOMIAL	PROBLEM			
	1	2	3	4*
3	.19-8	.13-8	.13-3	
4	.15-12	.11-11	.26-5	
5	.12-14	.17-15	.31-7	.20-4
6	.10-14	.12-15	.26-9	.31-6
7	.10-14	.13-15	.15-11	.30-8
8	.11-15	.11-15	.18-14	.21-10
9	.11-15	.11-15	.11-15	.11-12

* For a differential equation of order m, we have decided to use $k \geq m$. Since m=3 in this case, the order of basis function used is at least 6.

In Tables 3 and 4, we tabulated the variation of execution time and absolute error with the degree of basis function used. The notation .2-5 for $.2\times10^{-5}$ is used. These results indicate that increasing the degree of basis function increases the execution time and decreases the absolute error of the method. One can see that the solutions of problem 1 and 2 are within the roundoff error when an approximating polynomial of degree 5 is used. Thus, using an approximating polynomial of degree higher than 5 will not improve the solution, but it will surely increase the execution time, as shown in Table 3. These results strongly suggest that using a higher degree polynomial does not necessary improve the accuracy of the solution because ultimately the roundoff error will block any possible improvement to the solution. It is not clear what degree of polynomial one should use for a particular problem since its effect seems to be problem dependent. Besides, other factors such as the smoothness of the solution, the linearity of the problem and the chosen mesh would also affect the numerical results. For the sake of simplicity, we will use cubic polynomials in the subsequent testings, unless otherwise stated, but certainly this matter deserves further study.

3.6.2 Performance of the Mesh Selection Algorithm

We have incorporated the mesh selection scheme into HCOL so that it will compute an optimal mesh based on the solution on an initial uniform mesh. The mesh selection scheme performs satisfactorily for all seven problems. Some of the results are given in Figure 5, 6, 7, 8 which show the new grid selected by our mesh selection scheme for problem 3, 4, 5, 7.

Figure 5. Mesh Selected for Problem 3

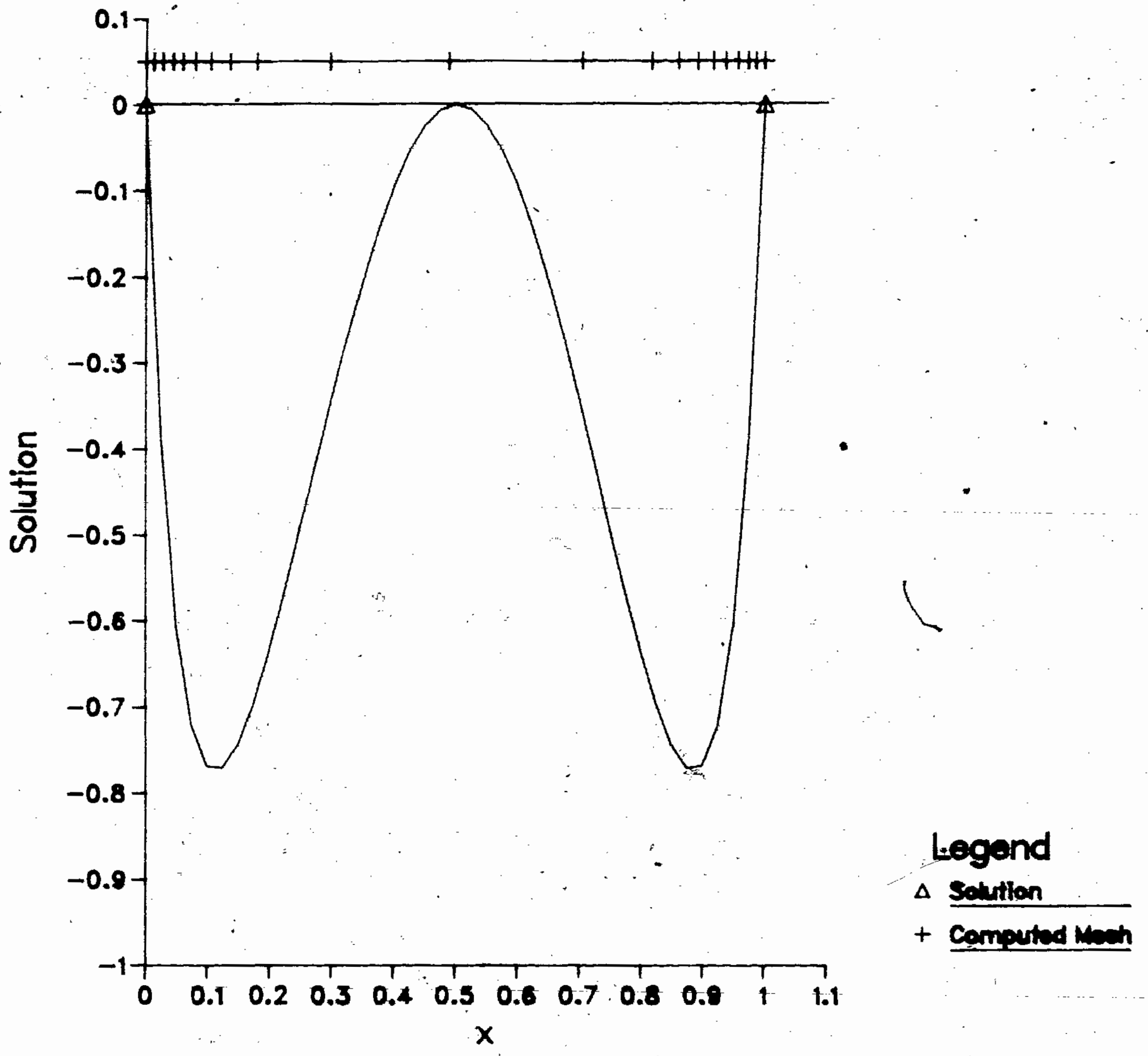


Figure 6. Mesh Selected for Problem 4

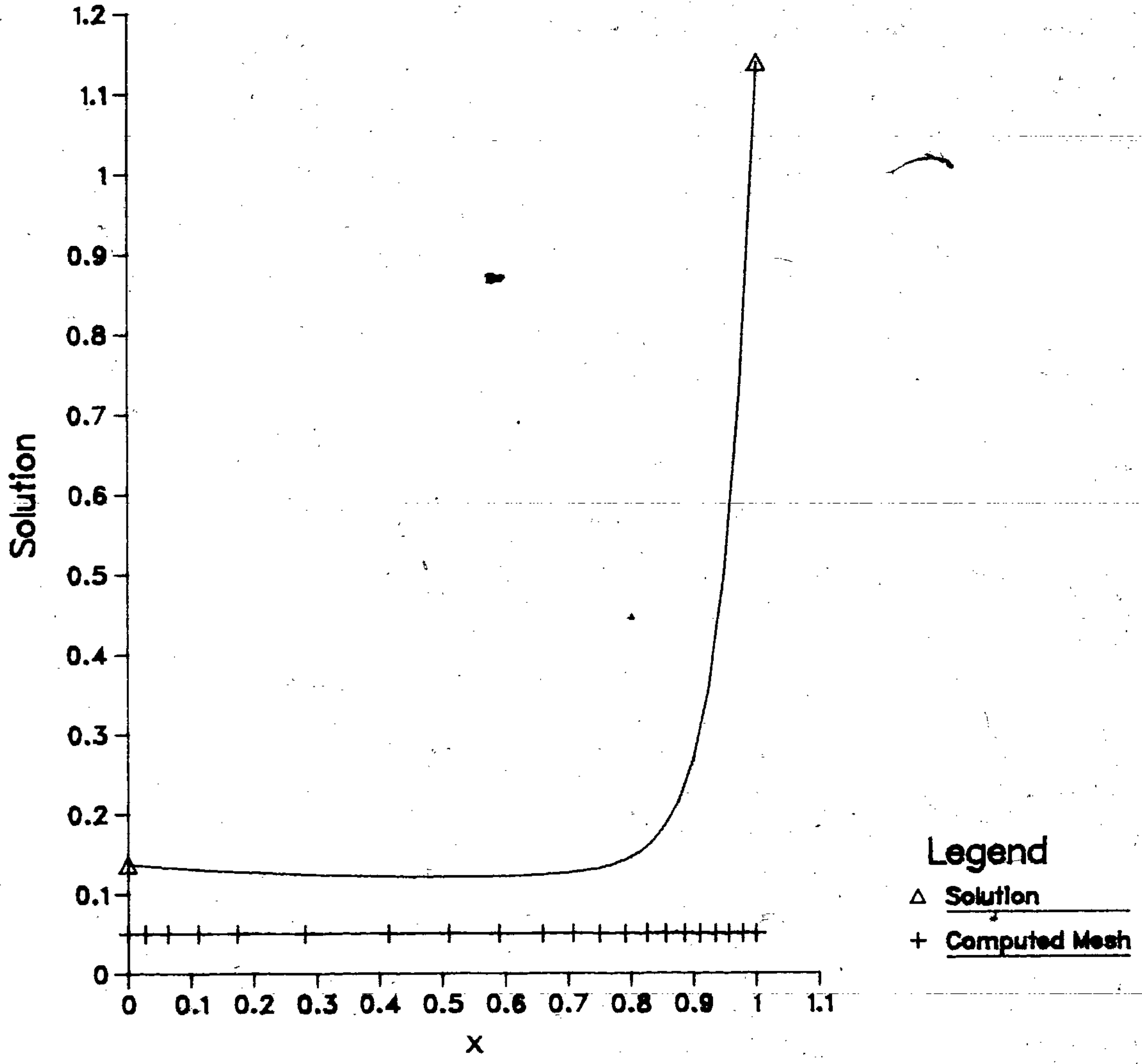


Figure 7. Mesh Selected for Problem 5

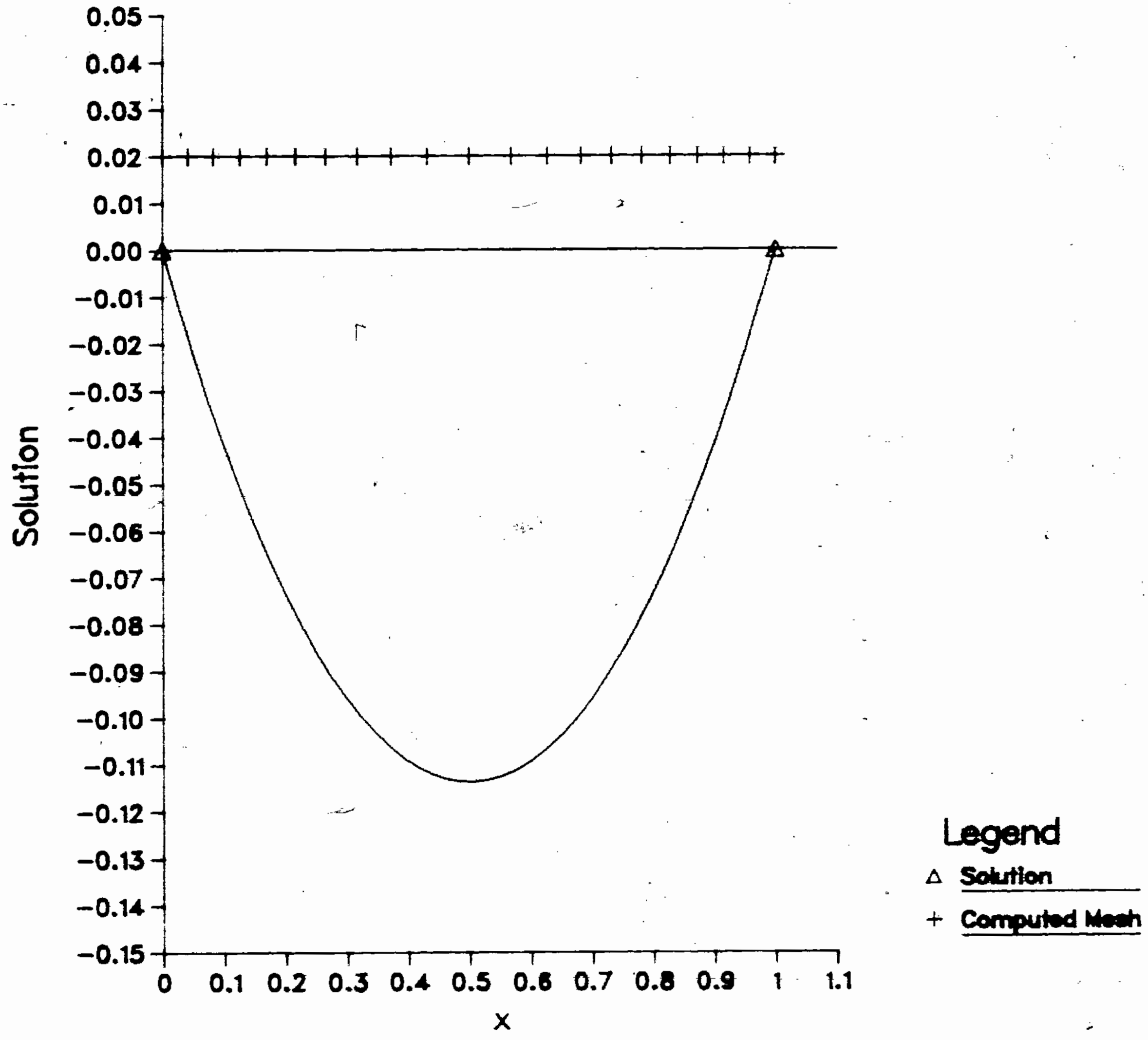
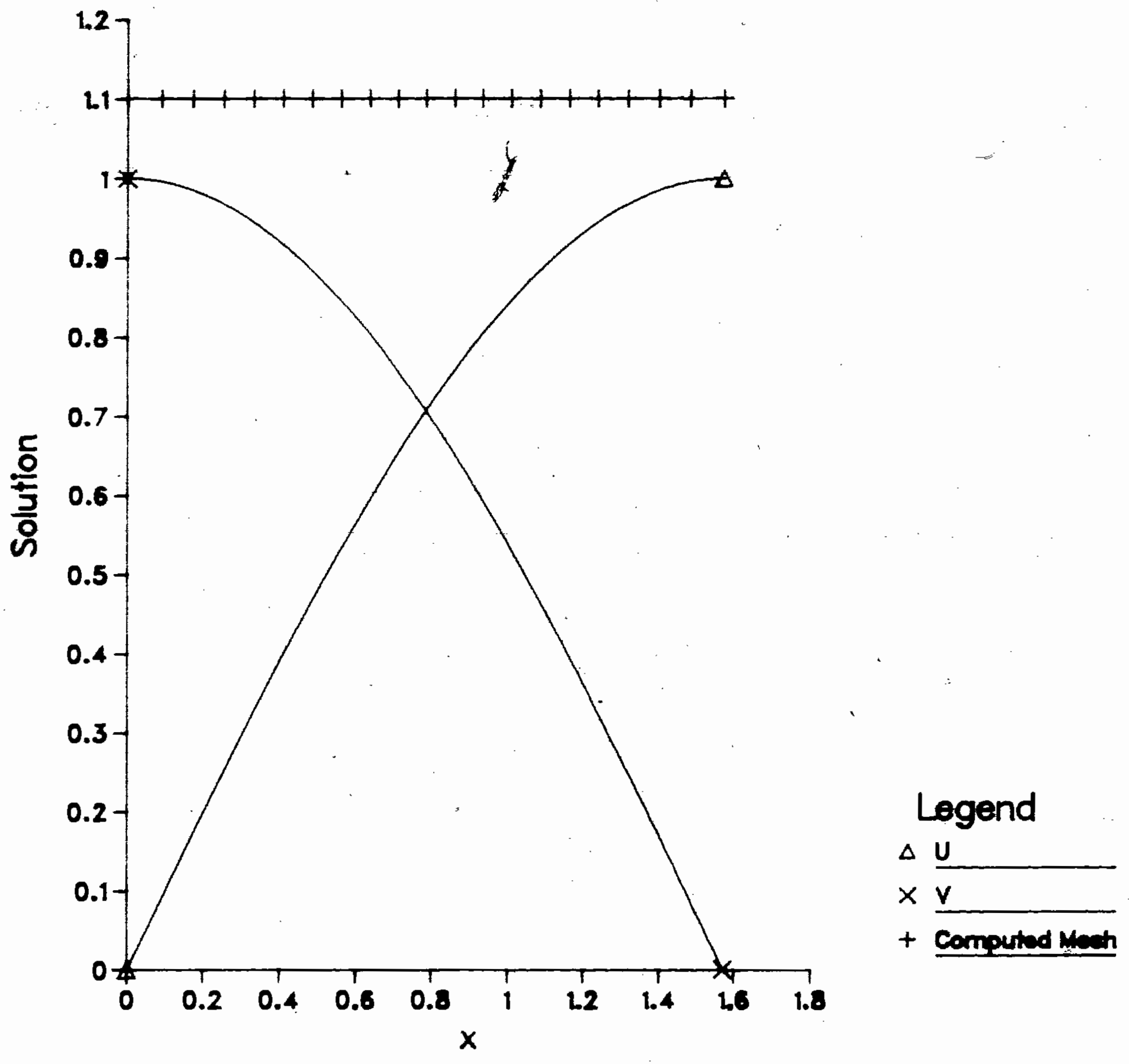


Figure 8. Mesh Selected for Problem 7



4. NUMERICAL EXAMPLES

4.1 Method of Lines

Most of the state-of-the-art numerical routines for solving partial differential equations use the method of lines to reduce the original problem to a set of IVP's. Two such routines are PDECOL [20] and DPDES, described in the following sections.

4.1.1 PDECOL

PDECOL is a versatile computer software package for solving a coupled system of nonlinear partial differential equations of at most second order in one space variable and one time dimension. The package uses spline collocation with B-splines for the initial spatial discretization technique. Then time integration methods which feature dynamic and automatic time step size are used to solve the resulting IVP's. Its implementors [20] claim that PDECOL can solve broad classes of difficult systems of partial differential equations that describe physical processes.

In particular, PDECOL solves the partial differential equation system

$$U^k_t = f_k(x,t,U^1,\ldots,U^N,U^1_x,\ldots,U^N_x,U^1_{xx},\ldots,U^N_{xx})$$

with initial condition

$$U^k(x,t_0) = U^k_0(x)$$

and boundary conditions

$$b^k_i(U^1,\ldots,U^N,U^1_x,\ldots,U^N_x) = z^k_i(t)$$

for $k=1,\ldots,N$, $i=1,2$.

The initial condition functions must be consistent with the boundary conditions and all functions should be continuous in t and at least piecewise continuous in x.

4.1.2 DPDES

This routine is similar to PDECOL and was recently incorporated into IMSL. It uses the method of lines to solve the partial differential equation system,

$$U^k_t = f_k(x,t,U^1,\ldots,U^N,U^1_x,\ldots,U^N_x,U^1_{xx},\ldots,U^N_{xx})$$

with initial conditions

$$U^k = U^k_0(x) \qquad \text{at } t=t_0$$

and boundary conditions

$$\alpha^k_1 U^k + \beta^k_1 U^k_x = \gamma^k_1(t) \qquad \text{at } x=x_a$$

$$\alpha^k_M U^k + \beta^k_M U^k_x = \gamma^k_M(t) \qquad \text{at } x=x_b$$

for $k=1,\ldots,N$.

Cubic Hermite polynomials are used in the spatial approximation. A constraint on the type of problems solvable by DPDES is that $U^k_0(x)$ must satisfy the boundary conditions and γ^k_1, γ^k_M must be continuous, or the boundary conditions will not be properly imposed for $t>t_0$. The time integrator used is the same as that used for PDECOL.

HERPDE takes a complementary approach. The temporal variable is discretized first and the resulting system of ordinary boundary value problems is then solved by spline collocation with a Hermite basis. Because of this difference, it is not clear how these approaches compare overall. In the next section, we compare the performance of HERPDE, PDECOL and DPDES on a few test problems. We caution the reader that these comparisons are by no means the final word. Since the basic intentions of the three codes are quite different, run times do not necessary mean anything. Still, we would hope that favorable performance by HERPDE would imply the usefulness of a more careful future implementation of this approach. Furthermore, a larger set of test problems would be needed to fully compare even the three current routines.

HERPDE starts the time discretization with the 1-step backward difference scheme since only the solutions at the initial time are known. As more time steps are taken, higher order discretization can then be used. Since the approximations at the first and second time steps are roughly order 1 and 2 in Δt, using a higher order discretization for the subsequent time steps may not give the expected higher order result.

4.2 Test Problems and Results

In this section, we examine the performance of HERPDE, PDECOL and DPDES on four problems of different degrees of difficulty. In the tests reported below, the error listed is the largest error measured at t=1. The results tabulated under X30 and X60 are obtained using thirty and sixty spatial subintervals respectively. For HERPDE, equal time steps are taken and their numbers are shown in the last column. The errors listed under SS (Special Start) are obtained by beginning with 10 smaller same sized time steps instead of the first regular time step used in

X30 and X60, followed by taking 29 regular time steps. This method has the advantage of giving a better approximated solution at $t+\Delta t$, thus increasing the accuracy of the solutions on the succeeding time levels because the computations at one time level require the solutions at the previous time steps. TOL is a parameter used by DPDES and PDECOL to control the time discretization error. They try to keep this below the space discretization error. Results from PDECOL are obtained using cubic B-splines.

Problem 8 Madsen-Sincovec [20]

Consider the equation

$$u_t = u_{xx} + \pi^2 \sin(\pi x)$$

with initial condition $u(0,x) = 1$

and boundary conditions $u(t,0) = u(t,1) = 1.$

The analytic solution of this simple diffusion type problem is easily seen to be

$$u(x,t) = 1 + \sin\pi x(1 - e^{-\pi^2 t}).$$

Table 5 shows the results obtained for this problem at t=1 using DPDES, PDECOL and HERPDE. Both DPDES and PDECOL give very accurate solutions. The number of time steps taken for DPDES is

Table 5

Computational results for Problem 8

Code	Tol	Maximum Error			Time (second)	Number of Time Steps
		X30	X60	SS		
DPDES	0.1-7	0.18-6			1.65	unknown
	0.1-11	0.83-7			7.23	unknown
PDECOL	0.1-7	0.43-6			3.46	64
	0.1-11	0.83-7			12.54	237
HERPDE	(CN)	0.32-4	0.32-4	0.30-4	1.61	10
		0.45-5	0.45-5	0.44-5	3.48	30
		0.12-5	0.11-5	0.12-5	5.06	60
	(AM2)	0.24-4	0.24-4	0.24-4	1.63	10
		0.42-4	0.31-4	0.23-4	3.61	30
		0.60-4	0.63-4		4.83	40
	(AM3)	0.19-4	0.19-4	0.19-4	1.68	10
		0.42-2	0.23-2	0.22-2	5.76	30
	(BD2)	0.26-3	0.26-3	0.48-4	1.54	10
		0.18-4	0.18-4	0.21-4	4.27	30
		0.41-5	0.40-5	0.50-5	9.37	60
	(BD3)	0.16-3	0.16-3	0.24-4	1.75	10
		0.85-5	0.85-5	0.63-5	4.44	30
		0.13-5	0.13-5	0.59-6	10.22	60

not available. For the same number of time steps, PDECOL always takes less time than HERPDE. The times given for HERPDE are for X30. The corresponding time for X60 is roughly twice as much.

One interesting result is that as the number of time steps increases, the accuracy of the Adams-Moulton methods decrease. This phenomenon requires further study. Our results also show that the size of the errors from X60 is almost the same as that from X30. This suggests that the time discretization is the main source of error. When we switched from the fixed time step to the special start SS, we found that HERPDE with BD3 gave more accurate solutions.

Problem 9

This example is given by IMSL as the model problem for its routine DPDES. The problem is

$$u_t = u_{xx} - 6xe^t + v$$

$$v_t = 0.5xv_{xx} - u_x + v$$

$$u(0,t) = 0 \qquad v(0,t) = 0$$

$$u(1,t) = e^t \qquad v(1,t) = e^t$$

$$u(x,0) = v(x,0) = x^3$$

with exact solutions: $u(x,t) = v(x,t) = x^3 e^t$.

The numerical results are shown in Table 6. Again, DPDES and PDECOL perform a bit better than HERPDE. The similar results obtained from X30 and X60 confirm that the time discretization again contributes most of the error observed. The increase in accuracy for BD3 with SS shows that the SS scheme can reduce the effect of the initial low order discretization. AM2 and AM3 fail to produce accurate solution.

Problem 10

This example is the popular Burger's equation:

$$u_t = -uu_x + \varepsilon u_{xx}$$

$$u(x,0) = \sin \pi x$$

$$u(0,t) = u(1,t) = 0.$$

The solution of this well known problem is a wave that steepens and moves to the right until a shock layer is formed at x=1. The wave dissipates after a time of $O(1/\varepsilon)$ and the solution decays to zero. See Whitham [31] for a complete exposition on the characteristics of Burger's equation.

Using a similar approach as ours and COLSYS as the BVP solver, Corless [11] has solved the same problem with a different initial boundary conditions for $\varepsilon = 0.2\text{-}1$. His solution is accurate to 0.2-1 when Gear's backward difference of order 4

Table 6

Computational results for Problem 9

CODE	Tol	Maximum Error			Time (second)	Number of Time Steps
		X30	X60	SS		
DPDES	0.1-7	0.11-4			3.29	unknown
	0.1-11	0.10-7			4.31	unknown
PDECOL	0.1-7	0.53-4			3.42	27
	0.1-11	0.98-8			14.10	77
HERPDE	(CN)	0.78-3	0.78-3	0.73-3	2.11	10
		0.87-4	0.87-4	0.84-4	8.80	30
		0.22-4	0.22-4	0.21-4	14.24	60
	(AM2)	0.11-3	0.13-3	0.36-4	2.60	10
	(AM3)	0.16-2	0.19-2	0.16-2	5.21	10
	(BD2)	0.71-2	0.71-2	0.27-2	2.20	10
		0.81-3	0.81-3	0.34-3	6.82	30
		0.20-3	0.20-3	0.86-4	14.82	60
	(BD3)	0.42-2	0.42-2	0.20-3	4.64	10
		0.43-3	0.43-3	0.11-4	7.76	30
		0.10-3	0.10-3	0.17-5	15.01	60

is used. Davis and Flaherty [12] have solved the same problem with their adaptive Galerkin method with good accuracy.

The numerical solution for ϵ=0.5-2 at t=1 obtained from HERPDE is shown in Figure 9. CN, BD2 and BD3 give similar results with 60 time steps. These results are consistent to 3 significance digits with those from DPDES and PDECOL. The execution times from BD3, DPDES and PDECOL using 30 spatial subintervals are 20.63, 25.99 and 8.65 seconds respectively. Other values of ϵ have also been tested. The solutions also agree with those from DPDES and PDECOL, which give almost identical solutions. We have difficulty in obtaining the expected solution profiles from both AM2 and AM3.

Problem 11

The following problem admits exact traveling wave solutions to nonlinear reaction-diffusion equations. This type of problem is of interest in biological applications and in the theory of flame propagation. The Fisher equation

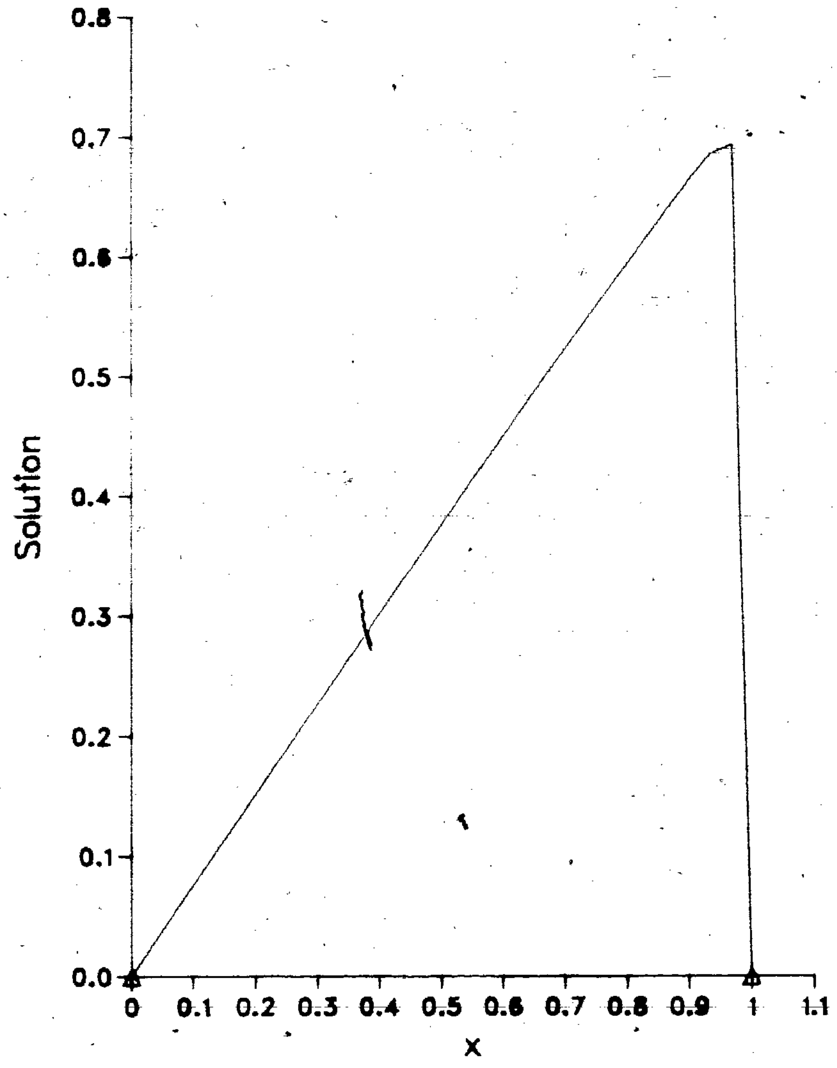
$$u_t = u_{xx} + u(1-u)$$

with initial condition $\quad u(x,0) = (1 + e^{x/\sqrt{6}})^{-2}$

and boundary conditions $\quad u(0,t) = (1 + e^{-5t/6})^{-2}$

$$u(6,t) = (1 + e^{(\sqrt{6}-5t/6)})^{-2}$$

Figure 9. Computational results at t=1 for Problem 10



has exact solution $u(x-5/\sqrt{6}t) = (1 + e^{(x-5/\sqrt{6}t)/\sqrt{6}})^{-2}$.

The solution to this problem is a wave that propagates to the right with its structure preserved up to t=8.

Reitz [23] has used this problem to study twelve numerical schemes which use variations of the difference method introduced by Saul'yev [27] for the diffusion terms. He found ~~that~~ the diffusion effects can reduce the accuracy of a numerical method significantly. This difference method is able to follow the wave for t up to 30.

Our results are indicated in Table 8. Although DPDES seems to have difficulty in solving this problem, PDECOL performs well. HERPDE gives a fairly accurate solution with BD3. The accuracy of HERPDE increases sharply using SS. In fact, the solution from BD3 is better than that of DPDES and almost as accurate as that of PDECOL. These results suggest that HERPDE should be able to perform as well as the other two codes if the proper time discretization strategy is used.

Table 7

Computational results for Problem 11

CODE	Tol	Maximum Error		Time (second)	Number of Time Steps
		X30	SS		
DPDES	0.1-7	0.26-4		8.24	unknown
	0.1-11	0.26-4		38.31	unknown
PDECOL	0.1-7	0.54-5		0.90	24
	0.1-11	0.12-7		2.56	76
HERPDE	(CN)	0.20-1	0.95-3	2.92	10
		0.41-2	0.29-3	4.55	30
		0.97-3	0.14-3	10.93	60
	(AM2)	0.19-1	0.50-2	4.53	10
	(AM3)	0.37-1	0.33-1	4.83	10
	(BD2)	0.12-2	0.19-3	2.25	10
		0.13-3	0.24-4	4.28	30
		0.33-4	0.62-5	9.05	60
	(BD3)	0.96-3	0.17-4	2.15	10
		0.10-3	0.42-6	4.94	30
		0.26-4	0.98-7	9.35	60

5. CONCLUSIONS

The results presented in the previous chapter indicate that our method is comparable to the library code DPDES and the production code PDECOL. In fact, with the special start procedure, HERPDE using BD3 performs as well as the other codes. The execution time of our method is a bit longer than the other two codes. However, we expect a more careful implementation of HERPDE to be faster than the current version.

The backward difference discretization formulae seem to work better than the Adams-Moulton approximations. The accuracy of Adams-Moulton discretization deteriorates when the number of time steps increases. This phenomenon requires further research. Our results also show that the second order CN scheme performs as well as the third order BD3 in most cases. However, BD3 is superior than CN for the nonlinear problem 11.

All things considered, our approach merits further study. It seems that our method has the potential to solve a large class of problems. For instance, one may be able to solve PDE with higher order terms like U_{xxx}, U_{xt} and U_{xxt}.

5.1 Suggestions for Further Study

There are a number of features of our method which warrant further study.

In the present implementation, there is one spatial mesh $a=x_1<x_2<\ldots<x_{N+1}=b$ for all the dependent $u^1, u^2, \ldots, u^{NPDE}$, where NPDE is the number of PDE´s. However, for many applications, such as those involving concentrations of chemical species, the regions of rapid variation are different for each dependent variable. If a single spatial mesh is used, it might get refined throughout the region. Thus the adaptive mesh would lose some of its advantage, although it would still serve to control the error. The ultimate code should have a different mesh for each dependent variable. The design of such a code would be quite challenging.

A way to improve the efficiency of this approach is to have variable time steps. This would allow the program to take as large a time step as required for a given level of accuracy. Further study is needed in designing a suitable adaptive time step scheme.

Further research is needed to determine which type and order of approximation for $U_t(x,t)$ should be used for a given order of spatial approximation. Our results indicate that the Adams-Moulton´s approximation has difficulty with a large number of time steps. Is this approximation unstable?

Our results show that both approaches to solving the parabolic problems give fairly accurate solutions. More work is needed in studying the type of problems which can best be handled by one scheme or the other.

5.2 Summary

In this project, an alternate row and column elimination scheme, ASIS, for solving the linear system arising from spline collocation with a monomial basis is presented and shown to require fewer operations than the existing methods.

A spatial mesh selection algorithm which uses an iterative relaxation method is implemented and found to perform satisfactorily.

We have also implemented an algorithm to solve parabolic problems in one space variable by the method of time discretization with spline collocation in space, using a Hermite basis. The algorithm has been tested on four problems. The scheme is shown to provide good accuracy, although it is less accurate and somewhat more costly in time when compared to DPDES and PDECOL. However, we expect a more careful implementation of our scheme will be as competitive as these production codes.

We hope the work presented here will stimulate more profound analyses of this method. Clearly it would be useful to have guidelines as to situations in which our method should be used rather than the method of lines or vice versa.

LIST OF REFERENCES

1. O.S. ANIZOR, An Adaptive Mesh-Generation Technique for Boundary-Layer Type Differential Equations, Using the Box Scheme. Research Report CS-75-08, Department of Computer Science, Univ. of Waterloo, 1975.

2. U. ASCHER, J. CHRISTIANSEN, and R.D. RUSSELL, A Collocation Solver for Mixed Order Systems of Boundary Value Problems, Tech. Report 77-13, Computer Science Department, Univ. of British Columbia, 1977.

3. U. ASCHER, J. CHRISTIANSEN and R.D. RUSSELL, Collocation Software for Boundary Value ODEs, ACM Trans. Math. Soft., 7 (1981), 209-222.

4. U. ASCHER, S. PRUESS and R.D. RUSSELL, On Spline Basis Selection for Solving Differential Equations, Tech. Report 81-4, Computer Science Department, Univ. of British Columbia, 1981 (to appear in SIAM).

5. U. ASCHER, and R.D. RUSSELL, Evaluation of B-splines for Solving Systems of Boundary Value Problems, Comp. Sci. Tech. Rep. 77-14, Univ. of British Columbia, 1977.

6. R.E. BELLMAN, and R.E. KALABA, Quasilinearization and Nonlinear Boundary-Value Problems, American Elsevier, New York (1965).

7. C.DE BOOR, A Practical Guide to Splines, Applied Math. Sciences Vol. 27, Springer-Verlag, New York (1978).

8. C.DE BOOR and B. SWARTZ, Collocation at Gaussian Points, SIAM J. Numer. Anal., 10(1973), 582-606.

9. R. BULIRSCH, and J. STOER, Introduction to Numerical Analysis, Springer-Verlag (1980).

10. S.D. CONTE and C.DE BOOR, Elementary Numerical Analysis: An Algorithmic Approach, 3rd ed., McGraw-Hill, New York (1980).

11. R.M. CORLESS, A Feasibility Study of the use of COLSYS in the Solution of Systems of Partial Differential Equations, Re. Report CS-82-32, Dept. of Mathematics, Univ. of Waterloo, 1982.

12. S.F. DAVIS, and J.E. FLAHERTY, An Adaptive Finite Element Method for Initial-Boundary Value Problems for Partial Differential Equations, SIAM J. Sci. Stat. Comput., 3(1982), 6-27.

13. J. DENNIS, and J. MORE, Quasi-Newton Methods, Motivation and Theory, SIAM Review 19(1977), 46-89.

14. J.C. DIAZ, G. FAIRWEATHER and P. KEAST, FORTRAN Packages for Solving Almost Block Diagonal Linear Systems by Modified Alternate Row and Column Elimination, Tech. Report 148/81, Dept. of Computer Science, Univ. of Toronto, 1981.

15. T.R. HOPKINS, and R. WAIT, A Comparison of Galerkin, Collocation and The Method of Lines for Partial Differential Equations, Int. J. Num. Meth. Engng, 12(1978), 1081-1107.

16. H.B. KELLER and M. LENTINI, Invariant Imbedding, the Box Scheme and an Equivalence Between Them (to appear).

17. M. LENTINI, M. OSBORNE, and R.D. RUSSELL, The Close Relationships between Methods for Solving Two-Point Boundary Value Problems (to appear).

18. M. LENTINI, and V. PEREYRA, A Variable Order, Variable Step, Finite Difference Method for Nonlinear Multipoint Boundary-Value Problem, Math. of Comp., 28(1974), 981-1003.

19. M. LENTINI, and V. PEREYRA, An Adaptive Finite Difference Solver for Nonlinear Two-point Boundary Problems with Mild Boundary Layers, SIAM J. Numer. Anal., 14(1977), 91-111.

20. N.K. MADSEN and R.F. SINCOVEC, PDECOL: General Collocation Software for Partial Differential Equations, ACM Trans. Math. Soft., 5(1979), 326-351.

21. M.R. OSBORNE, On Shooting Methods for Boundary Value Problems, J. Math. Anal. Appl. 27(1969), 417-433.

22. V. PEREYRA, and E.G. SEWELL, Mesh Selection for Discrete Solution of Boundary Problems in Ordinary Differential Equations, Numer. Math., 23(1975), 261-268.

23. R.D. REITZ, A Study of Numerical Methods for Reaction-Diffusion Equations, SIAM J. Sci. Stat. Comput., 2(1981), 95-105.

24. E. ROTHE, Zweidimensionale parabolische Randwertaufgaben als Grenzfall eindimensionaler Randwertaufgaben, Math. Ann., 102(1930), 650-670.

25. R.D. RUSSELL, private communication.

26. R.D. RUSSELL, and L.F. SHAMPINE, A Collocation Method for Boundary Value Problems, Numer. Math., 19(1972), 1-28.

27. V.K. SAUL'YEV, Integration of Equations of Parabolic Type by the Method of Nets, Pergamon, New York (1964).

28. N.L. SCHRYER, Numerical Solution of Time-Varying Partial Differential Equations in One Space Variable, Bell Laboratories Computing Science Technical Report #53, 1977.

29. R.F. SINCOVEC, Generalized Collocation Methods for Time-Dependent, Nonlinear Boundary-Value Problems, Soc. Pet. Eng. J., (Oct. 1977), 345-352.

30. A.B. WHITE Jr., On the Numerical Solution of Initial/Boundary-Value Problems in One Space Dimension, Report CNA-156, Center for Numerical Analysis, UT Austin, 1980.

31. G.B. WHITHAM, Linear and Nonlinear Waves, Wiley-Interscience, New York (1974).